SAPIENZA
UNIVERSITÀ DI ROMA

FACOLTÀ DI SCIENZE MATEMATICHE FISICHE E NATURALI
Dottorato in Matematica Applicata XXVI ciclo

Model Reduction for a Dynamic Programming Approach to optimal control problems with PDE constraints

Advisor:
Prof. MAURIZIO FALCONE

Candidate:
ALESSANDRO ALLA
matricola 1067357

Anno Accademico 2012-2013
Dipartimento di Matematica 'Guido Castelnuovo'

"The research is the easiest.
The outline is the most fun.
The first draft is the hardest,
because every word of the
outline has to be fleshed out.
The rewrite is very satisfying."
(Ken Follet)

Acknowledgements

At the end of my thesis I would like to thank all those people who made these three years an unforgettable period of my life.

First of all, I would really like to express my deepest gratitude to my advisor Prof. Maurizio Falcone who supported me, motivated me, helped me with his patience, enthusiasm, and immense knowledge. His guidance helped me in through the time of research and writing of this thesis. The topics, he has introduced to me, were always very interesting, exciting and challenging.

Besides my advisor, I would like to thank Prof. E. Carlini, Prof. M. Hinze and Prof. G. Rozza for accepting to be member of my thesis committee and for their remarks, questions and suggestions on my work.

I want also to thank Prof. Volkwein for his hospitality at University of Constance, the chance to visit and collaborate with him and his research group. My colleagaus, in particluar I am grateful to Dr Kalise for many interesting discussions and collaborating with me.

I feel to thank Prof. Kunisch and Prof. Fernandez-Cara who pushed me, during the time I spent in Graz and Seville, to try to get the PhD.

"Se sei convinto di riuscire in qualcosa, non credo basti solo la convizione ma anche di qualcuno che ti stia vicino"... Quando ho letto questa frase ho subito pensato ai miei genitori e a mio fratello Gianluca che, sono stati, sono e saranno SEMPRE al mio fianco incoraggiando le mie scelte con enfasi indescrivibile! Anche se lontani, in cittá diverse, NON ci lasceremo mai...

Finally I would like to thank my close friends, although most of them are still wondering what I am doing. In particolare ringrazio Andrea e Matthias (ordine alfabetico!!!), compagni di risate, scherzi, avventure, uscite, vacanze, mare, chattate, telefonate and so on ...

Concludo ringraziando Veronica, a modo suo, che rallegra le mie giornate!

Contents

1. Introduction

1.1. Context and General background

The numerical solution of optimal control problems is a crucial issue for many industrial applications such as aerospace engineering, chemical processing, power systems, and resource economics. In some cases the original problem comes from a different setting, e.g. when one has to fit a given set of data or has to solve a shape optimization problem (see for instance [7]), but has been reformulated in terms of a control problems for an appropriate dynamic and cost functional. The typical goal is to compute an optimal trajectory for the controlled system and the optimal control corresponding to it. A classical finite horizon optimal control problem can be described as follow:

$$\begin{aligned} \inf_{u\in U} J_{x,t}(u(\cdot)) &:= \int_0^t L(y(s),u(s),s)e^{-\lambda s}\,ds + g(y(t)) \\ &\text{subject to } \dot{y}(s) = f(y(s),u(s),s), \quad y(0) = x. \end{aligned} \tag{1.1}$$

Here, y is the state trajectory, x the initial condition, u denotes the control, U is the control space and λ the discount factor. The pair (y,u) satisfy the system (1.1).
In the framework of open-loop controls the classical solution is a pair (y^*,u^*) where u^* minimizes the cost functional $J_{x,t}(u(\cdot))$ and y^* is the corresponding trajectory to the optimal control. One way to obtain the optimal pair is based on the Pontryagin's Maximum Principle (see [78]) which leads to the solution of a two-point boundary value problem for the coupled state/adjoint system. The numerical solution can be obtained via a shooting method (see [73] for details). Despite its simplicity and mathematical elegance, this approach is not always satisfactory because the initialization of the shooting method can be a difficult task, mainly for the adjoint variables. Moreover, this approach is typically based on necessary conditions for optimality and produces only open-loop controls.
Another way to solve optimal control problems which involves open-loop controls is called *direct method.* It consists in discretizing directly the optimal control problem, leading to nonlinear optimization problem, which can be solved by various numerical algorithms as shown in the book by Kelley [61] and by Gerdts [44].

An alternative way to solve optimal control problems was introduced by Bellman [18] which leads to deal with the value function $v(x,t)$ defined as the infimum of the cost functional $J_{x,t}(u(\cdot))$:

$$v(x,t) := \inf_{u\in U} J_{x,t}(u(\cdot)).$$

It is well known that the Bellman's Dynamic Programming (DP) produces optimal con-

trol in feedback form so it looks more appealing in terms of online implementations and robustness. However, the synthesis of feedback controls requires the previous knowledge of the value function and this is the major bottleneck for the application of DP. In fact the value of optimal control problems are known to be only Lipschitz continuous even when the data are regular and the characterization of the value function is obtained in terms of a first order nonlinear Hamilton-Jacobi-Bellman (HJB) partial differential equation:

$$-\frac{\partial v(x,t)}{\partial t} - \lambda v(x,t) + \max_{u \in U}\{-f(x,u) \cdot Dv(x,t) - L(x,u,t)\} = 0.$$

In the last thirty years, the DP approach has been pursued for all the classical control problems in the framework of viscosity solution introduced by Crandall and Lions in the 80's (see [15] for a comprehensive illustration of this approach). Viscosity solution allows us to characterize the value function as the unique solution of HJB equation [16].
Due to the analitycal complexity of the solution of HJB, several approximation schemes have been proposed for this class of equations, ranging from finite differences [30] to semi-Lagrangian [27, 37, 39] and finite volume methods [62]. These algorithms compute the solution on the iteration of the value space looking for a fixed point of the equation. They converge to the value function but their convergence is slow (see [38] for error estimates on Semi-Lagrangian schemes).
We must recall that algorithms based on the iteration in the space of controls (or policies) for the solution of HJB equations have a rather long history, starting more or less at the same time of dynamic programming. The Policy Iteration (PI) method, also known as Howard's algorithm [57], has been investigated by Kalaba [59] and Pollatschek and Avi-Itzhak [77] who proved that it corresponds to the Newton method applied to the functional equation of dynamic programming. Later, Puterman and Brumelle [79] have given sufficient conditions for the rate of convergence to be either superlinear or quadratic. More recent contributions on the policy iteration method can be found in Santos and Rust [90] and Bokanowski et al. [21]. Results on its numerical implementation and diverse hybrid algorithms related to the proposed scheme have been reported in Capuzzo-Dolcetta and Falcone [26], González and Sagastizábal [48], and Grüne [50].

Finally, we should mention that an acceleration method based on the the set of subsolutions has been studied in Falcone [37]. More in general, dealing with domain decomposition methods for Hamilton-Jacobi-Bellman equations, we should also mention approaches based on domain decomposition algorithms as in Falcone et al. [42] and more recently by Cacace et al. [24], on geometric considerations as in Botkin, et al. [23], and those focusing on the localization of numerical schemes which leads to Fast Marching Methods. This approach has shown to be very effective for level-set equations related to front propagation problems (see e.g. the book by Sethian [92]), i.e. eikonal type equations. At every iteration, the scheme is applied only on a subset of nodes (localization) which are the nodes close to the front, the so-called *narrow band*. The remaining part of the grid is divided into two parts: the accepted region, where the solution has been already computed, and the far region where the solution will be computed little by little

in the following iterations. At every iteration, one node is accepted and moved from the narrow band to the accepted region; the narrow band is then updated adding the first neighbors of that node (which before where in the far region). For eikonal type equations these methods converge in finite number of iterations to the correct viscosity solution and have a very low complexity (typically $\mathcal{O}(N \ln(N))$ where N is the cardinality of the grid). More recently several efforts have been made to extend these methods to more complex problems where the front propagation is anisotropic [93] and/or to more general Hamilton-Jacobi equations as in [11]. However, their implementation is rather delicate and their convergence to the correct viscosity solution for general Hamilton-Jacobi equations is still an open problem; we refer to [25] for a an extensive discussion and several examples of these limitations.
However, these methods suffer the so-called *curse of the dimensionality*, namely, the fact that the dimension of the partial differential equation characterizing the value function increases as the dimension of the state space does, constitutes a major computational challenge towards a practical implementation of numerical algorithms for optimal control design based on viscosity solutions of HJB equations.

In recent years, new tools have been developed to deal with optimal control problems in high dimension (see for instance the book by Lions [71] or by Hinze et al. [54]). In particular, new techniques emerged to reduce the number of dimensions in the description of the dynamical system or, more in general, of the solution of the problem that one is trying to optimize. These methods are generally called *reduced-order methods* and include for example the Proper Orthogonal Decomposition (POD, see [56, 95, 100]) method, the reduced basis approximation (see [76]) and Balance Truncation method ([12]). POD is also known as Princial Component Analysis for data analysis (see [70]) or Karhunen-Loeve transform in discrete signal processing (see [72]).
The general idea for all these methods is that, when the solutions are sufficiently regular, one can represent them via Galerkin expansion so that the number of variables involved in this discretization will be strongly reduced. In some particular cases, as for the heat equation, even 5 basis functions will suffice to have a rather accurate POD representation of the solution (see e.g. [1] and, more in general, [64]). Having this in mind, it is reasonable to start thinking to follow a different approach based on DP and HJB equations. In this new approach we will first develop a basis functions representation of the solution along a reference trajectory and then use this basis to set-up a control problem in the new space of coordinates. The corresponding HJB equation will just need 3-5 variables to represent the state of the system. Moreover, by this method one can obtain optimal control in feedback form looking at the gradient of the value function.
As far as we know, the first tentative to obtain feedback control with POD is due to Atwell and King in [13] where they control heat equation with a quadratic cost functional. This problem leads to an ordinary differential Riccati's equation whose solution provides the optimal control in feedback form. Then, we should mention that a first tentative to couple POD and HJB equations was made by Kunisch and Xie in [68, 69] where the dynamics was given by a diffusion dominated equations. They have presented a third order TVD Runge Kutta scheme to approximate an evolutive HJB equation for the control of viscous Burger equation solving the problem with 4 POD basis functions. The

viscous term allows to keep the system parabolic and to work with a low number of POD basis functions. Then, in the paper by Kunisch, Volkwein and Xie [67] one can see this feedback control approach applied to the viscous Burgers equation with a semi-lagrangian scheme for the infinite horizon problem. Since they implemented a parallel code for HJB they were able to solve the problem up to 10 basis functions.

As we said, in many control problems it is desired to design a stabilizing feedback control but often the closed-loop solution can not be found analytically, even for the unconstrained case since it involves the solution of the corresponding HJB equations. One approach to circumvent this problem is the repeated solution of an open-loop optimal control problem for a given state. The first part of the resulting open-loop input signal is implemented and the whole process is repeated. Control approaches using this strategy are referred to as Model Predictive Control (MPC), Moving Horizon Control or Receding Horizon Control (for more informations, the interested reader can see the books [52, 82]). In general one distinguishes between linear and Nonlinear Model Predictive Control (NMPC). Linear MPC refers to a family of MPC schems in which linear models are used to predict the system dynamics and considers linear constraints on the states and inputs. Note that even if the system is linear, the closed loop dynamics are nonlinear due to the presence of the constraints. NMPC refers to MPC schemes that are based on nonlinear models and/or consider a non quadratic cost-functional and general nonlinear constraints. Although linear MPC has become an increasingly popular control technique used in industry, in many applications linear models are not sufficient to describe the process dynamics adquately and nonlinear models must be applied. This inadequacy of linear models is one of the motivations for the increasing interest in nonlinear MPC (please refer to [35, 9] for an introduction to NMPC).
The prediction horizon has a crucial role in Model Predictive Control, for instance the Quasi Infinite horizon NMPC allows a efficient formulation of NMPC while guaranteeing stability and the performances of the closed-loop as shown in [8, 36] under appropriate assumptions.
Since the computational complexity of MPC schemes grows rapidly with the length of the optimization horizon, esimates for minimal stabilizing horizons are of particular interest to ensure stability. Stability and suboptimality analysis for NMPC schemes without stabilizing constraints is presented in Chapter 6 of the book by Grüne and Panneck ([52]) where they proved conditions to get asymptotic stability with minimal horizon. Note that the stabilization of the problem and the computation of the minimal horizon involve the (Relaxed) Dynamic Programming Principle (see also [51, 75]). This approach allows estimates of the horizon based on controllability properties of the system.
Since several optimization problems are performed in the context of MPC it is reasonable to couple the problem with POD. As far as we know, the unique approach of coupling MPC with POD is due to Ghiglieri and Ulbrich in [45].

1.2. Contributions

Chapter 2 is an overview of several model reduction methods used in the solution of optimal control problems with PDE constraints. We present in particular Proper Orthogonal Decomposition (POD) which will turn out to be our key tool of this thesis. Then, we briefly introduce Balance Truncation method and Reduced Basis approach making a short comparison between the methods.

We illustrate a new accelerated algorithm which can produce an accurate approximation of the value function with a reduced CPU time compared to the classical DP methods. The proposed scheme can be used in a large variety of problems connected to static HJB equations, such as infinite horizon optimal control, minimum time control.
Our new method couples two techniques: the value iteration method (VI) and the policy iteration method (PI) for the solution of Bellman equations. The first is known to be slow but convergent for any initial guess, whereas the second is known to be fast when it converges (but if not initialized correctly, convergence might be as slow as for the value iteration). The approach that we consider relates to multigrid methods (we refer to Santos [89] for a brief introduction to the subject in this context), as the coupling that we introduce features an unidirectional, two-level mesh. However, as far as we know, the efficient coupling between the two methods has not been investigated as we have done here in Chapter 3.

The thesis deals also with the approximation of a *finite horizon* optimal control problem for an evolutive linear and nonlinear partial differential equation, e.g. the advection–diffusion equation. Although the coupling between HJB and POD already exists in ([68, 67]), our contribution involves a new adaptive POD basis representation of the solution applied to a Dynamic Programming scheme for the evolutive Hamilton-Jacobi equation characterizing the value function.
Due to the curse of dimensionality, we need to restrict the dimension to a rather low number of basis functions (typically 4 or 5) and this limitation naturally affects the accuracy of the POD approximation. The difficulty clearly appears when dealing with advection–diffusion problems where with few basis functions the POD method does not have enough informations to follow correctly the solution. It is worth to note that one could also adopt a strategy based on *a-posteriori* error estimators as in the linear quadratic case presented by Tröltzsch and Volkwein [98] (see also the recent developments presented in [60]).
We circumvent this problem updating our POD basis functions during the evolution and splitting the original problem into subproblems. Every sub-problem is set in an interval $I_k = [T_k, T_{k+1}]$ where we recompute the POD basis. Behind the adaptive method and the choice of the T_k there are two important *a-posteriori* estimators: the first is related to the computation of the POD basis functions whereas the second takes into account the residual of the dynamics. The approach described in Chapter 4 is clearly different from the more classical approach based on the solution of the system of necessary conditions obtained by the Pontryagin maximum principle. The main advantage here is that we naturally obtain optimal controls in feedback form but the price we pay is related to the

well know curse of dimensionality of Dynamic Programming. Note that in this approach, in every time interval we will first develop a POD representation of the solution along a reference trajectory and then use this basis to set-up a control problem in the new space of coordinates. Then, in the POD space the state of the system will just need 3-5 variables to be represented in a rather accurate way, these variables will appear in the corresponding evolutive Hamilton-Jacobi equation.

Then, we present the analysis of the stability of NMPC algorithm without terminal constraints applied to a semilinear parabolic equation with advection term. A minimal finite horizon is determined to guarantee stabilization of the system. A similar approach for the wave equation is presented in [10]. The main difference here we have added an advection term in the state equation and control constraints which influences the conditions for the stability of the problem.
Note that the computation of the minimal horizon involves a relaxed form of the dynamic programming principle where we need to compute a parameter α. We give some condition in order to obtain α.
Since the minimal horizon can be large, the numerical approximation is very expensive. Our contribution is to apply POD model reduction to reduce the computational cost by computing suboptimal solutions. Therefore, we have investigated the asymptotic stability of the surrogate model giving conditions on the coefficient α in the POD model. New conditions on the parameter are clearly influenced by the computation of the POD basis functions. Thus, we provide a study of this influence of α subject to the number of POD basis function chosen. Asymptotic stability for reduced order models is presented in Chapter 5.

1.3. Organization

The Manuscript is divided into 6 chapters.

- **Chapter 2** recalls Proper Orthogonal Decomposition in Section 2.1, Balance Truncation method in Section 2.2 and finally the Reduce Basis approach in Section 2.3.

- **Chapter 3** is organized as follows. In Section 3.1, we introduce some basic notions for optimal control synthesis by the dynamic programming principle and its numerical solution via a value and policy iteration schemes. Section 3.2 contains the core of the proposed accelerated method and discuss practical implementation details. In Section 3.3 we propose a theoretical estimate for our accelerated algorithm. Finally, Section 3.4 shows our numerical results on a number of different examples concerning infinite horizon optimal control, minimum time control, and some further extensions towards the optimal control of partial differential equations.

- **Chapter 4** studies the possibility to solve optimal control problems of partial differential equations by HJB equations. In order to work with HJB equation we have to perform a model reduction method, e.g. POD and an adaptive-method to be sure to have a very low dimensional model. The chapter is organized as follows.

In Section 4.1 we will deal with optimal control problem by means of POD and HJB equation. In Section 4.2 we will show an estimate of the approximation of the value function with a POD surrogate model. In Section 4.3 we present the adaptive POD method, and finally, in Section 4.4 the numerical tests.

- **Chapter 5** focuses on the asysmptotic stability of an infinite horizon optimal control problem via Model Predictive Control (MPC). The problem is formulated in Section 5.1. Then we explain MPC algorithm and how to compute the prediction horizon which ensures the stability of the method in Section 5.2. In Section 5.3 we study the corresponding finite horizon problem and we apply POD model reduction to the problem. Section 5.3.3 is devoted to the main results of the chapter: to carry out conditions for asympotic stability of the reduced model. Finally we give several numerical tests in Section 5.4

- **Chapter 6** provides conclusions, future directions and perspectives.

Appendix A contains some well-known results on the convergence of the Newton's Method and recalls a result on the continuous dependence of the data of ordinary differential equations.

1.3.1. Original material for this thesis.

Let us briefly mention the original contributions which are behind this thesis.
Chapter 3 is based on the submitted paper [4] to SIAM J. of Scientific Computing. The accepted Proceeding [3] shortly summarizes the topic. The most recent paper [5], submitted to ENUMATH 2013, tests the algorithm for differential games.
Chapter 4 is based on the paper [1] published in K. Kunisch, K. Bredies, C. Clason, G. von Winckel, (eds) *Control and Optimization with PDE Constraints*, and the IFAC CPDE conference article [2] which has to appear.
Chapter 5 is based on the paper [6] submitted to Advances in Computational Mathematics.

2. Overview of results for model reduction methods

The aim of this chapter is to present an overview of some model reduction techniques applied to dynamical or parametrized systems in order to reduce significantly the complexity of the problem. In particular, we discuss the Proper Orthogonal Decomposition (POD) which will turn out to be our key ingredient in this thesis.

2.1. Proper Orthogonal Decomposition

In this section we explain the Proper Orthogonal Decomposition method. More details can be found, for instance, in the book by Holmes et al. [56] or in the lecture notes by Volkwein [100].

2.1.1. Singular Values Decomposition

In this section we briefly recall the main notions about the Singular Value Decomposition (SVD) of a matrix since it is strictly linked to the POD method as we will see in the next section. We refer to [20, 47] for further details on SVD.

Definition 2.1 (Singular values and vectors) *Let $Y \in \mathbb{R}^{m\times n}$ be a matrix of rank $d \leq n$ with $m > n$. Let $\Psi := \{\psi_i\}_{i=1}^m \subset \mathbb{R}^m$ and $V := \{v_i\}_{i=1}^n \subset \mathbb{R}^n$ be the set of orthonormal vectors such that:*

$$Yv_i = \sigma_i\psi_i \quad \text{and} \quad Y^T\psi_i = \sigma_i v_i \quad \text{for } i = 1,\dots,d. \tag{2.1}$$

Then, $\sigma_1,\dots,\sigma_d$ are called singular values, and the vectors $\psi \in \Psi, v \in V$ are called: right and left singular vectors respectively.

Theorem 2.1 (Existence of SVD) *Let $Y = [y_1,\dots,y_n]$ be a given matrix with real value $m \times n$ of rank $d \leq \min\{m,n\}$. Then, there exists a singular value decomposition of Y, with real numbers $\sigma_1 \geq \sigma_2 \geq \dots \geq \sigma_d > 0$ and orthogonal matrices $\Psi = [\psi_1,\dots,\psi_m] \in \mathbb{R}^{m\times m}$ and $V = [v_1,\dots,v_n] \in \mathbb{R}^{n\times n}$ such that:*

$$Y = \Psi\Sigma V^T, \qquad \Sigma = \begin{pmatrix} D & 0 \\ 0 & 0 \end{pmatrix} \in \mathbb{R}^{m\times n}, \tag{2.2}$$

where D is the diagonal matrix with singular values.

The following Lemma provides a uniqueness result for SVD.

Lemma 2.1 *For any matrix $Y \in \mathbb{R}^{m\times n}$ the singular values are uniquely defined. The singular vectors correspondent to the singular values greater than zero of multiplicity one are unique up to a change of the sign.*

Remark 2.1 *SVD may be interpreted as a generalization of eigenvalues problem. If we insert the first equation in (2.1) into the second and vice-versa, we see the right singular vectors $\{\psi_i\}_{i=1}^d$ are eigenvectors of YY^T with eigenvalues $\lambda_i = \sigma_i^2$ and the left singular vectors $\{v_i\}_{i=1}^d$ are eigenvectors of Y^TY with eigenvalues $\lambda_i = \sigma_i^2$, then the following equalities hold:*

$$YY^T\psi_i = \sigma_i^2\psi_i, \quad \text{and} \quad Y^TYv_i = \sigma_i^2 v_i, \ \text{for } i = 1,\dots,d$$

and for $i > d$ we have $YY^T\psi_i = 0 = Y^TYv_i$.

Another important result for SVD is about the optimal approximation in the Frobenius norm:

Theorem 2.2 *Let $Y \in \mathbb{R}^{m\times n}$ be a matrix of rank $d \le n$, with $m \ge n$, Let $Y = \Psi\Sigma V^T$ be the SVD with singular values $\sigma_1,\dots,\sigma_n$. We define Y^ℓ of rank ℓ, such that $\sigma_{\ell+1} = \sigma_{\ell+2} = \ldots = \sigma_n = 0$.*
Then, Y^ℓ is the best approximation with respect to the Frobenius-norm of Y among the matrices of rank ℓ:

$$\left|\left|Y - Y^\ell\right|\right|_F = \min_{\text{rank(B)}=\ell} ||Y - B||_F = \sqrt{\left(\sum_{i=\ell+1}^{d} \sigma_i^2\right)}.$$

This result holds even with the 2-norm, but Y^ℓ is not uniquely determined.

A useful algorithm to compute the SVD was introduced by Golub and Kahan in the 60s (see [47]).

2.1.2. Proper Orthongonal Decomposition for the uncontrolled dynamics

Let us consider a system of ordinary differential equations:

$$\begin{cases} \dot{y}(t) = Ay(t) + f(t, y(t)), \quad t \in (0,T] \\ \\ y(0) = y_0, \end{cases} \tag{2.3}$$

where $y_0 \in \mathbb{R}^m$ is a given initial data, $A \in \mathbb{R}^{m\times m}$ a given matrix, $f : [0,T] \times \mathbb{R}^m \to \mathbb{R}^m$ a continuous function in both arguments and locally Lipschitz-type with respect to the second variable. It is well–known that under these assumptions there exists an unique classical solution $y \in C^1(0;T;\mathbb{R}^m) \cap C([0,T];\mathbb{R}^m)$ to (2.3). The solution is given by:

$$y(t) = e^{tA}y_0 + \int_0^t e^{(t-s)A} f(s, y(s))\, ds.$$

We build an equispaced grid on time Δt. Let $t_0 := 0 < t_1 < t_2 < \ldots < t_n \leq T$ with $t_j = j\Delta t, \;\; j = 0, \ldots, n$. Let us suppose to know the exact solution of (2.3) on the time grid points t_j, $j \in \{1, \ldots, n\}$. Our aim is to determine a POD basis of rank $\ell < n$ to describe the set:

$$y_j = y(t_j) = e^{t_j A} y_0 + \int_0^{t_j} e^{(t_j - s)A} f(s, y(s))\, ds \quad j \in \{1, \ldots, n\},$$

solving the following minimization problem:

$$\min_{\psi_1, \ldots, \psi_\ell \in \mathbb{R}^m} \sum_{j=1}^{n} \alpha_j \left\| y_j - \sum_{i=1}^{\ell} \langle y_j, \psi_i \rangle \psi_i \right\|^2, \tag{2.4}$$

where the coefficients α_j are non-negative and y_j are the so called *snapshots*, e.g. the solution of (2.3) at a given time t_j. At the end of this section we will give more details about the α_j. The norm, here and in the sequel of the chapter, can be interpreted as the standard Euclidean norm.
Solving (2.4) we look for an orthonormal basis $\{\psi\}_{i=1}^{\ell}$ which minimizes the distance between the sequence y_j with respect to its projection onto this unknown basis. Moreover, it is rather useful to look for $\ell \ll \min\{m, n\}$ in order to reduce the dimension of the problem considered.
The solution of (2.4) is given by the following theorem where it shows the influence of the SVD in the POD method:

Theorem 2.3 *Let $Y = [y_1, \ldots, y_n] \in \mathbb{R}^{m \times n}$ be a given snapshots matrix of rank $d \leq \min\{m, n\}$. Further, let $Y = \Psi \Sigma V^T$ be the Singular Value Decomposition of Y, where $\Psi = [\psi_1, \ldots, \psi_m] \in \mathbb{R}^{m \times m}$, $V = [v_1, \ldots, v_n] \in \mathbb{R}^{n \times n}$ are orthogonal matrices, and the matrix $\Sigma \in \mathbb{R}^{m \times n}$ has the form given in (2.2). Then, for any $\ell \in \{1, \ldots, d\}$ the solution*

$$\min_{\psi_1, \ldots, \psi_\ell \in \mathbb{R}^m} \sum_{j=1}^{n} \alpha_j \left\| y_j - \sum_{i=1}^{\ell} \langle y_j, \psi_i \rangle \psi_i \right\|^2 \qquad \text{s.t. } \langle \psi_i, \psi_k \rangle = \delta_{ik} \text{ for } 1 \leq i, k \leq \ell \tag{2.5}$$

is given by the right singular vectors $\{\psi_i\}_{i=1}^{\ell}$, that is, the first ℓ columns of Ψ.

Motivated by the previous theorem we give the following definition:

Definition 2.2 *Assume that $\ell \in \{1, \ldots, d\}$, the vectors $\{\psi_i\}$ for $i = 1, \ldots, \ell$ are called* POD basis *of rank ℓ.*

A practical way to obtain the POD basis functions of rank ℓ is to compute the eigenvalues and eigenvectors, namely, if $n < m$ the POD basis will be the eigenvectors $v_1, \ldots, v_\ell \in \mathbb{R}^n$ of the $n \times n$

$$YY^T v_i = \lambda_i v_i \qquad \text{for } i = 1, \ldots, \ell$$

and then setting

$$\psi_i = \frac{1}{\sqrt{\lambda_i}} Y v_i. \tag{2.6}$$

If the dimension of the matrix is $m < n$, we can compute directly the POD basis, solving $m \times m$ eigenvalues problem: $YY^T\psi_i = \lambda_i\psi_i$. However, it is well-known the computation of the SVD of a matrix Y is more stable than the computation of the eigenvalues of YY^T as explained in [47] and the references therein. For this reason in this thesis, we will always deal with the computation of singular values decomposition.
To concretely apply the POD method, the choice of the parameter ℓ has a crucial role. There are no a priori estimates which guarantee to build a coherent reduced model, but one can focus on heuristic considerations, introduced by Sirovich [95], as to have the following ratio close to one:

$$\mathcal{E}(\ell) = \frac{\sum_{i=1}^{\ell} \lambda_i}{\sum_{i=1}^{d} \lambda_i}. \tag{2.7}$$

This indicator is motivated by the fact that:

$$\sum_{j=1}^{n} \alpha_j \left\| y_j - \sum_{i=1}^{\ell} \langle y_j, \psi_i \rangle \psi_i \right\|^2 = \sum_{i=\ell+1}^{d} \sigma_i^2$$

which tells us the importance of the POD basis functions we neglect, although this error is strictly related to the computation of the snapshots.

Let us discuss the non-negative weigths $\{\alpha_j\}_{j=1}^n$ in (2.4). For this purpose we introduce a continuous version of the POD method. Let $y : [0, T] \to \mathbb{R}^m$ be the unique solution of (2.3). Since we are interesing in finding a POD basis of rank ℓ which describes all the trajectories $y(t)$, we look for the following continuous minimization problem:

$$\min_{\widehat{\psi}_1,\dots,\widehat{\psi}_\ell \in \mathbb{R}^m} \int_0^T \left\| y(t) - \sum_{i=1}^{\ell} \langle y(t), \widehat{\psi}_i \rangle \widehat{\psi}_i \right\|^2 dt \text{ s.t. } \langle \widehat{\psi}_i, \widehat{\psi}_j \rangle = \delta_{ij}, \quad 1 \le i, j \le \ell. \tag{2.8}$$

Note that the following idea can be applied even in a discrete framework. We can solve (2.8) working with the Lagrange multipliers. If $\ell = 1$ we have the following problem:

$$\min_{\widehat{\psi} \in \mathbb{R}^m} \int_0^T \left\| y(t) - \langle y(t), \widehat{\psi} \rangle \widehat{\psi} \right\|^2 dt \text{ s.t. } ||\widehat{\psi}|| = 1, \tag{2.9}$$

where we suppose that $\{\widehat{\psi}_i\}_{i=2}^m$ are chosen such that $\{\widehat{\psi}, \widehat{\psi}_2, \dots, \widehat{\psi}_m\}$ is an orthonormal base $\mathbb{R}^m$ with respect to the standard inner product. Then it holds:

$$y(t) = \langle y(t), \widehat{\psi} \rangle \widehat{\psi} + \sum_{i=2}^{m} \langle y(t), \widehat{\psi}_i \rangle \widehat{\psi}_i \qquad \forall t \in [0, T].$$

Thus,

$$\int_0^T \left\| y(t) - \langle y(t), \widehat{\psi}\rangle \widehat{\psi} \right\|^2 \, dt = \int_0^T \left\| \sum_{i=2}^{m} \langle y(t), \widehat{\psi}_i\rangle \widehat{\psi}_i \right\| \, dt = \sum_{i=2}^{m} \int_0^T |\langle y(t), \widehat{\psi}_i\rangle| \, dt,$$

we conclude that (2.9) is equivalent to:

$$\max_{\widehat{u}\in\mathbb{R}^m} \int_0^T |\langle y(t), \widehat{\psi}\rangle|^2 \, dt \text{ s.t. } ||\widehat{\psi}|| = 1. \tag{2.10}$$

The Lagrangian functional: $\mathcal{L} : \mathbb{R}^m \times \mathbb{R} \to \mathbb{R}$ associated to (2.10) is given by

$$\mathcal{L}(\psi, \lambda) = \int_0^T |\langle y(t), \psi\rangle|^2 \, dt + \lambda(1 - ||\psi||^2) \text{ per } (\psi, \lambda) \in \mathbb{R}^m \times \mathbb{R}.$$

The first order optimality conditions are:

$$\nabla\mathcal{L}(\psi, \lambda) := 0 \text{ in } \mathbb{R}^m \times \mathbb{R}.$$

We compute the partial derivative of $\mathcal{L}$ with respect to ψ:

$$\nabla_\psi L(\psi, \lambda) = 2\left(\int_0^T \langle y(t), \psi\rangle y(t) - \lambda\psi \right) \equiv 0 \text{ in } \mathbb{R}^m$$

which provides

$$\int_0^T \langle y(t), \psi\rangle y(t) \, dt = \lambda\psi \text{ in } \mathbb{R}^m. \tag{2.11}$$

Therefore we define the following operator $\mathcal{R} : \mathbb{R}^m \to \mathbb{R}^m$ as follow:

$$\mathcal{R}\psi = \int_0^T \langle y(t), \psi\rangle y(t) \, dt \text{ with } \psi \in \mathbb{R}^m \tag{2.12}$$

Lemma 2.2 *The operator $\mathcal{R}$ is linear and bounded. Furthermore,*

1. *$\mathcal{R}$ is non-negative:*
$$\langle \mathcal{R}\psi, \psi\rangle \geq 0 \ \ \forall \psi \in \mathbb{R}^m$$
2. *$\mathcal{R}$ is self-adjoint (or symmetric):*
$$\langle \mathcal{R}\psi, \widehat{\psi}\rangle = \langle \psi, \mathcal{R}\widehat{\psi}\rangle \quad \textit{for each } \psi, \widehat{\psi} \in \mathbb{R}^m$$

Thanks to the operator $\mathcal{R}$ we can write (2.11) as the eigenvalue problem:

$$\mathcal{R}\psi = \lambda\psi \quad \text{in } \mathbb{R}^m.$$

It follows from Lemma 2.2 that the eigenvectors $\{\psi_i\}_{i=1}^m$ of $\mathcal{R}$ with associated eigenvalues $\{\lambda_i\}_{i=1}^m$ hold for:

$$\mathcal{R}\psi_i = \lambda_i\psi_i \quad \text{per } 1 \leq i \leq m, \qquad \lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_m \geq 0. \tag{2.13}$$

We remark that

$$\int_0^T |\langle y(t), \psi_i\rangle|^2 \, dt = \int_0^T \langle\langle y(t), \psi_i\rangle y(t), \psi_i\rangle \, dt = \langle \mathcal{R}\psi_i, \psi_i\rangle = \lambda_i ||\psi_i||^2 = \lambda_i$$

for $i \in \{1, \ldots, m\}$ such that ψ_1 solves (2.9). The continuous version of Theorem 2.3 is given by the following theorem:

Theorem 2.4 *Let $y \in C([0,T];\mathbb{R}^m)$ be the unique solution of (2.3). Then, the POD basis of rank ℓ, computed solving (2.8), is given by the eigenvectors $\{\psi_i\}_{i=1}^\ell$ of $\mathcal{R}$ corresponding to the greater eigenvalues $\lambda_1 \geq \ldots \geq \lambda_\ell$.*

Let us introduce the linear and bounded operator $\mathcal{Y} : L^2(0,T) \to \mathbb{R}^m$:

$$\mathcal{Y}v = \int_0^T v(t)y(t) \, dt \qquad v \in L^2(0,T).$$

The adjoint operator $\mathcal{Y}^* : \mathbb{R}^m \to L^2(0,T)$ such that:

$$\langle \mathcal{Y}^*\psi, v\rangle_{L^2(0,T)} = \langle \psi, \mathcal{Y}v\rangle \qquad \forall (\psi, v) \in \mathbb{R}^m \times L^2(0,T),$$

is defined as follows

$$(\mathcal{Y}^*\psi)(t) = \langle \psi, y(t)\rangle \quad u \in \mathbb{R}^m; \;\; \text{almost every } t \in [0,T].$$

Thus,

$$\mathcal{Y}\mathcal{Y}^*\psi = \int_0^T \langle \psi, y(t)\rangle y(t) \, dt = \int_0^T \langle y(t), u\rangle y(t) \, dt = \mathcal{R}u$$

for every $\psi \in \mathbb{R}^m$, moreover $\mathcal{Y}\mathcal{Y}^* = \mathcal{R}$. On the other hand the operator

$$\mathcal{K}v(t) := (\mathcal{Y}^*\mathcal{Y}v)(t) = \left\langle \int_0^T v(s)y(s) \, ds, y(t) \right\rangle = \int_0^T \langle y(s), y(t)\rangle v(s) \, ds,$$

for every $v \in L^2(0,T)$ and almost every $t \in [0,T]$. Then, $\mathcal{K} = \mathcal{Y}^*\mathcal{Y}$. One may prove that $\mathcal{K}$ is linear, bounded, non-negative and self-adjoint and compact (see [100]). As in the discrete case, the POD basis functions may be computed solving:

$$\mathcal{K}v_i = \lambda_i v_i \text{ for } i \leq i \leq \ell, \quad \lambda_1 \geq \ldots \geq \lambda_\ell \geq 0, \qquad \int_0^T v_i(t)v_j(t) \, dt = \delta_{ij}$$

and setting as in (2.6)

$$\psi_i = \frac{1}{\sqrt{\lambda_i}}\mathcal{Y}v_i = \frac{1}{\sqrt{\lambda_i}}\int_0^T v_i(t)y(t)\,dt \qquad i = 1,\dots,\ell.$$

Let us go back to the discrete version, we can define the discrete version of $\mathcal{R}$:

$$YDY^T\psi = \sum_{j=1}^{n}\alpha_j\langle y_j,\psi\rangle y_j := \mathcal{R}^n\psi,$$

where D is the diagonal matrix with α_j.
Note that the operator $\mathcal{R}^n : \mathbb{R}^m \to \mathbb{R}^m$ is linear and bounded. Moreover,

$$\langle \mathcal{R}^n\psi,\psi\rangle = \left\langle \sum_{j=1}^{n}\alpha_j\langle y_j,\psi\langle y_j,\psi \right\rangle = \sum_{j=1}^{n}\alpha_j|\langle y_j,\psi\rangle|^2 \geq 0,$$

which ensures the non-negativity of $\mathcal{R}^n$. Finally,

$$\begin{aligned}\langle \mathcal{R}^n\psi,\widehat{\psi}\rangle &= \left\langle \sum_{j=1}^{n}\alpha_j\langle y_j,\psi\rangle y_j,\widehat{\psi} \right\rangle = \sum_{j=1}^{n}\alpha_j\langle y_j,u\rangle\langle y_j,\widehat{\psi}\rangle \\ &= \left\langle \sum_{j=1}^{n}\alpha_j\langle y_j,\widehat{u}y_j\rangle,\psi \right\rangle = \langle \mathcal{R}^n\widehat{\psi},\psi\rangle = \langle \psi,\mathcal{R}^n\widehat{\psi}\rangle,\end{aligned}$$

for every $\psi,\widehat{\psi}\in\mathbb{R}^m$, $\mathcal{R}^n$ is self-adjoint. Therefore, $\mathcal{R}^n$ has the same properties of $\mathcal{R}$. To summarize we have:

$$\mathcal{R}^n\psi_i^n = \lambda_i^n\psi_i^n \qquad \lambda_1^n \geq \ldots \geq \lambda_\ell^n \geq \ldots \lambda_{d(n)}^n > \lambda_{d(n)+1}^n = \ldots = \lambda_m^n = 0, \tag{2.14}$$

$$\mathcal{R}\psi_i = \lambda_i\psi_i \qquad \lambda_1 \geq \ldots \geq \lambda_\ell \geq \ldots \lambda_d > \lambda_{d+1} = \ldots = \lambda_m = 0. \tag{2.15}$$

where $d(n)$ is the rank of the discrete problem, therefore we observe that

$$\int_0^T ||y(t)||^2\,dt = \sum_{i=1}^{d}\lambda_i = \sum_{i=1}^{m}\lambda_i. \tag{2.16}$$

Indeed,

$$\mathcal{R}\psi_i = \int_0^T \langle y(t),\psi_i\rangle y(t)\,dt \qquad \forall i\in\{1,\dots,m\}.$$

If we consider the inner product with ψ_i, the vector has unitary norm and summing over i we achieve the following:

$$\sum_{i=1}^{d}\int_0^T |\langle y(t),\psi_i\rangle|^2\,dt = \sum_{i=1}^{d}\langle \mathcal{R}\psi_i,\psi_i\rangle = \sum_{i=1}^{d}\lambda_i = \sum_{i=1}^{m}\lambda_i.$$

If we write $y(t) \in \mathbb{R}^m$ in terms of $\{\psi_i\}_{i=1}^m$ we will have

$$y(t) = \sum_{i=1}^{m} \langle y(t), \psi_i \rangle \psi_i,$$

hence we get again (2.16):

$$\int_0^T ||y(t)||^2 \, dt = \sum_{i=1}^{m} \int_0^T |\langle y(t), \psi_i \rangle|^2 \, dt = \sum_{i=1}^{m} \lambda_i.$$

In the same way, we have

$$\sum_{j=1}^{n} \alpha_j ||y(t_j)||^2 = \sum_{i=1}^{d(n)} \lambda_i^n \qquad \forall n \in \mathbb{N}. \tag{2.17}$$

Let $y \in C([0,T]; \mathbb{R}^m)$, to ensure convergence:

$$\sum_{j=1}^{n} \alpha_j ||y(t_j)||^2 \to \int_0^T ||y(t)||^2 \, dt \qquad \text{when } \Delta t \to 0 \tag{2.18}$$

we must properly chose the parameters α_j. For instance, we can take the trapezoidal rule:

$$\alpha_1 = \frac{\Delta t}{2}, \;\; \alpha_j = \Delta t \text{ per } 2 \le j \le n-1, \; \alpha_n = \frac{\Delta t}{2}. \tag{2.19}$$

Then, the following convergence theorem holds:

Theorem 2.5 *Let us assume that $y \in C^1([0,T]; \mathbb{R}^m)$ is the unique solution of (2.3). Let $\{(\psi_i^n, \lambda_i^n)\}_{i=1}^m$ and $\{(\psi_i, \lambda_i)\}_{i=1}^m$ be the eigenvalues-eigenfunction of (2.14) e (2.15). Suppose that $\ell \in \{1, \ldots, m\}$ is given such that*

$$\lim_{n \to +\infty} \sum_{i=1}^{m} \lambda_i^n = \sum_{i=1}^{m} \lambda_i. \tag{2.20}$$

and

$$\sum_{i=\ell+1}^{m} \lambda_i \neq 0, \qquad \sum_{i=\ell+1}^{m} |\langle y_0, \psi_i \rangle|^2 \neq 0.$$

Then,

$$\lim_{n \to +\infty} ||\mathcal{R}^n - \mathcal{R}||_{L(\mathbb{R}^m)} = 0. \tag{2.21}$$

This implies:

$$\lim_{n \to +\infty} |\lambda_i^n - \lambda_i| = \lim_{n \to +\infty} ||\psi_i^n - \psi_i|| = 0, \qquad \textit{for } 1 \le i \le \ell,$$

$$\lim_{n\to+\infty}\sum_{i=\ell+1}^{m}(\lambda_i^n-\lambda_i)=0 \ \textit{and}\ \lim_{n\to+\infty}\sum_{i=\ell+1}^{m}|\langle y_0,\psi_i^n\rangle|^2=\sum_{i=\ell+1}^{m}|\langle y_0,\psi_i\rangle|^2.$$

It is interesting to observe that the coefficients α_j are determined in order to approximate, for instance with a trapezoidal rule, the integral (2.18).

2.1.3. Reduced Order Modelling via POD

In the previous section, we have introduced the computation in $\mathbb{R}^m$ of the POD basis functions of rank ℓ and discussed some applications to problems with initial conditions. Once the POD basis is computed, one may obtain a reduced order model for the problem (2.3). The focus of this section is to introduce this important application.
Assume that we know the POD basis functions $\{\psi_j\}_{j=1}^{\ell}$ of rank $\ell\in\{1,\dots,m\}$ in $\mathbb{R}^m$, we make the following ansatz:

$$y^\ell(t)=\sum_{j=1}^{\ell}y_j^\ell(t)\psi_j=\sum_{j=1}^{\ell}\langle y^\ell(t),\psi_j\rangle\psi_j,\qquad\forall t\in[0,T] \tag{2.22}$$

where the Fourier coefficients $y_j^\ell(t)$, $1\le j\le\ell$ are functions from $[0,T]$ to $\mathbb{R}$.
Since

$$y(t)=\sum_{j=1}^{m}\langle y(t),\psi_j\rangle\psi_j\qquad\forall t\in[0,T],$$

$y^\ell(t)$ will turn out to be an approximation for $y(t)$ with $\ell<m$. Inserting (2.22) in (2.3) we have:

$$\begin{cases}\displaystyle\sum_{j=1}^{\ell}\dot{y}_j^\ell(t)\psi_j=\sum_{j=1}^{\ell}y_j(t)A\psi_j+f(t,y^\ell(t)),\qquad t\in(0,T]\\[2ex]\displaystyle\sum_{j=1}^{\ell}y_j^\ell(0)\psi_j=y_0.\end{cases} \tag{2.23}$$

Note that now (2.23) is a problem in $\mathbb{R}^m$ for the coefficients $y_j^\ell(t)$, $1\le j\le\ell$ and $t\in[0,T]$. Moreover, we assume that (2.23) holds after the projection into the subspace of dimension ℓ :

$$V^\ell=\operatorname{span}\{\psi_1,\dots,\psi_\ell\}. \tag{2.24}$$

From the first equation in (2.23) and $\langle\psi_i,\psi_j\rangle=\delta_{ij}$ we deduce that

$$\dot{y}_i^\ell(t)=\sum_{j=1}^{\ell}y_j^\ell(t)\langle A\psi_j,\psi_i\rangle+\langle f(t,y^\ell(t)),\psi_i\rangle \tag{2.25}$$

for $1\le i\le\ell$ and $t\in(0,T]$. Let us introduce compact notations for (2.25): the matrix

$$A^\ell=(a_{ij}^\ell)\in\mathbb{R}^{\ell\times\ell}\qquad\text{con } a_{ij}^\ell=\langle A\psi_i,\psi_j\rangle, \tag{2.26}$$

the vectors:

$$y^\ell = \begin{pmatrix} y_1^\ell \\ \vdots \\ y_\ell^\ell \end{pmatrix} : [0,T] \to \mathbb{R}^\ell \tag{2.27}$$

and the non-linear function $F = (F_1, \dots, F_\ell)^T : [0,T] \times \mathbb{R}^\ell \to \mathbb{R}^\ell$:

$$F_i(t,y) := \left\langle f\left(t, \sum_{j=1}^{\ell} y_j \psi_j\right), \psi_i \right\rangle \qquad \text{for } t \in [0,T] \;\; y = (y_1, \dots y_\ell) \in \mathbb{R}^\ell. \tag{2.28}$$

Then, (2.25) may be written as:

$$\begin{cases} \dot{y}^\ell(t) = A^\ell y^\ell(t) + F(t, y^\ell(t)) \\ y^\ell(0) = y_0^\ell \end{cases} \tag{2.29}$$

where

$$y_0^\ell = \begin{pmatrix} \langle y_0, \psi_1 \rangle \\ \vdots \\ \langle y_0, \psi_\ell \rangle \end{pmatrix} \in \mathbb{R}^\ell. \tag{2.30}$$

The system (2.29) is approximating following a Galerkin projection where the basis functions are computed by the POD method for (2.3). If the dimension of the system is $\ell \ll m$ we get an impressive reduction.

2.1.4. Discrete Empirical Interpolation Method

The Reduced Model introduced in (2.29) is a nonlinear system where the problem with the POD–Galerkin approach is the complexity of the evaluation of the non-linearity. To illustrate this problem we have a look at the non-linearity $F(t, y^\ell)$ in (2.28). Setting $\Psi^\ell = [\Psi_1, \dots, \Psi_\ell] \in R^{m \times \ell}$ we can write

$$F(t, y^\ell(t)) = \Psi^T f(t, \Psi y^\ell(t)) = \langle f(t, y(t)), \Psi \rangle.$$

This can be interpreted in the way that the variable $y^\ell(t) \in \mathbb{R}^\ell$ is first expanded to a vector $\Psi y^\ell(t) \in \mathbb{R}^m$ then the non-linearity $f(t, \Psi y^\ell(t))$ is evaluated and, at the end, we go back to the reduced-order model. This is computationally expensive since it implies the evaluation of the nonlinear term in the full dimensional model and therefore the reduced model is not independent of the full dimension m.
To avoid this computationally expensive evaluation the *Discrete Empirical Interpolation Method* (DEIM) was introduced. It is based on a POD approach combined with a greedy algorithm (see [29] for more details on DEIM and Section 2.3.2 for the presentation of the Greedy method). We define

$$b(t) = f(t, \Psi y^\ell(t)) \in \mathbb{R}^m \qquad \text{for } t \in [0,T].$$

The function $b(t)$ is approximated by a Galerkin ansatz utilizing $\mathcal{P}$ linearly independent functions $\Phi_1, \ldots, \Phi_{\mathcal{P}} \in \mathbb{R}^m$, i.e.

$$b(t) \approx \sum_{k=1}^{\mathcal{P}} \Phi_k c_k(t) = \mathbf{\Phi} c(t) \tag{2.31}$$

with $c(t) = [c_1(t), \ldots, c_{\mathcal{P}}(t)]^T \in \mathbb{R}^{\mathcal{P}}$ and $\mathbf{\Phi} = [\Phi_1, \ldots, \Phi_{\mathcal{P}}] \in \mathbb{R}^{m \times \mathcal{P}}$. Hence we can write the approximation of $F(t, \cdot)$ as

$$F(t, y^\ell(t)) = \Psi^T f(t, \Psi y^\ell(t)) = \Psi^T b(t) \approx \Psi^T \mathbf{\Phi} c(t).$$

The question which arises is how to compute the matrix $\mathbf{\Phi}$ and the vector $c(t)$. Let $\mathcal{I} \in \mathbb{R}^{\mathcal{P}}$, be an index vector and $B \in \mathbb{R}^{m \times \mathcal{P}}$ be a given matrix. Then by $B_{\mathcal{I}}$ we denote the submatrix consisting of the rows of B corresponding to the indices in $\mathcal{I}$.
Let us assume we have computed $\mathbf{\Phi}$ and $\mathcal{I}$ by an algorithm, then we proceed as follows. For simplicity we introduce here the matrix $P = (e_{\mathcal{I}_1}, \ldots, e_{\mathcal{I}_{\mathcal{P}}}) \in \mathbb{R}^{m \times \mathcal{P}}$, where $e_{\mathcal{I}_i} = (0, \ldots, 0, 1, 0, \ldots, 0)^T \in \mathbb{R}^m$ is a vector with all zeros and at the $\mathcal{I}_i-$th row a one. Note that $\mathbf{\Phi}_{\mathcal{I}} = P^T \mathbf{\Phi}$ holds. To evaluate the approximated non-linearity we need $c(t)$. Since we know $\mathbf{\Phi}$ and the index vector $\mathcal{I}$ we can compute

$$c(t) = (P^T \mathbf{\Phi})^{-1} P^T b(t) = (P^T \mathbf{\Phi})^{-1} P^T f(t, \Psi y^\ell(t)) \quad \text{for } t \in [0, T].$$

Suppose that the matrix P can be moved into the non-linearity. Then we obtain

$$P^T f(t, \Psi y^\ell(t)) = (f(t, \Psi y^\ell(t)))_{\mathcal{I}} = f(t, P^T \Psi y^\ell(t)).$$

Let us now have a look at the computational expenses. The matrices

$$P^T \Psi \in \mathbb{R}^{\mathcal{P} \times \ell}, \; (P^T \mathbf{\Phi})^{-1} \in \mathbb{R}^{\mathcal{P} \times \mathcal{P}} \text{ and } \Psi^T \mathbf{\Phi} \in \mathbb{R}^{\ell \times \mathcal{P}}$$

can be precomputed. All the precomputed quantities are independent from the full dimension m. Additionally, during the iterations the nonlinearity has only to be evaluated at the $\mathcal{P}$ interpolation points since $P^T \Psi y^\ell(t) \in \mathbb{R}^{\mathcal{P}}$. Typically the dimension $\mathcal{P}$ is much smaller than the full dimension. This allows the reduced-order model to be completely independent of the full dimension.
The DEIM algorithm generates the basis using the POD approach which is applied to the snapshots of the nonlinearity $b(t) = f(t, y(t))$ to compute $\mathbf{\Phi}$. The selection of the interpolation points $\mathcal{I}$ is based on a greedy algorithm. The idea is to select spatial points to limit the growth of an error bound which is the residual. The indices are constructed inductively from the input data (see Algorithm 1). We have to mention that DEIM is built upon the Empirical Interpolation Method which aims to deal with nonlinear terms in reduced model (see [17]). The two methods are the same, DEIM is the tensorial matricial version of EIM.

Algorithm 1: The discrete empirical interpolation method (**DEIM**)

Data: $\mathcal{P}$ and matrix $F = [f(t_1, y(t_1)), \ldots, f(t_n, y(t_n))] \in \mathbb{R}^{m\times n}$;

1. Compute POD basis $\mathbf{\Phi} = [\mathbf{\Phi_1}, \ldots, \mathbf{\Phi}_\mathcal{P}]$ for F.
2. idx$\leftarrow \arg\max_{j=1,\ldots,m} |(\Phi_1)_j|$;
3. $U = [\Phi_1]$ and $\vec{i}$
4. **for** $i = 2$ *to* $\mathcal{P}$ **do**
   5. $u \leftarrow \Phi_i$;
   6. Solve $U_{\vec{i}} = idx$;
   7. $r \leftarrow u - Uc$;
   8. $idx \leftarrow \arg\max_{j=1,\ldots,m} |(r)_{\{j\}})|$;
   9. $U \leftarrow [U, u]$ and $\vec{i} \leftarrow [\vec{i}, idx]$;

end

10. **return** $\mathbf{\Phi}$ **and** $\vec{i}$.

2.2. Balance Truncation

Another important tool, among model reduction methods, is the so-called Balance Truncation which helps to reduce the complexity of linear time-invariant system. Please refer to [12, 46] for an extensive presentation of the method and to [31] for infinite dimensional linear systems.
Let us consider the following linear time-invariant (LTI) system:

$$\begin{cases} \dot{x}(t) = Ax(t) + Bu(t),\ x(0) = x_0 \quad \text{for } t \in (0, +\infty) \\ y(t) = Cx(t) \end{cases} \tag{2.32}$$

where $x(t) \in \mathbb{R}^n$ is called the system state, $x_0 \in \mathbb{R}^n$ is the initial condition of the system, $u(t) \in \mathbb{R}^m$ is said to be the system input, or control in the sequel, and $y(t) \in \mathbb{R}^q$ is called the system output. The matrices A, B, C are assumed to have appropriate sizes.
Let us define observability and controllability for a linear time-invariant system. Then, we provide conditions to determine if the system (2.32) is controllable and observable (please refer to [31, 101]).

Definition 2.3 *The system* (2.32)*, or the pair* (A, B)*, is called controllable if for any* $x_0 \in \mathbb{R}^n$ *and final state* $x_T \in \mathbb{R}^n$*, there exists an input* $u(t)$ *such that the solution of* (2.32) *satisfies* $x(T) = x_T$.

Let us define the *controllability Gramian* $W_c(t)$ and the *observability Gramian* $W_\circ(t)$:

$$W_c(t) = \int_0^t e^{sA} B B^T e^{sA^T}\, ds \qquad W_\circ(t) = \int_0^t e^{sA} C^T C e^{sA^T}\, ds.$$

Controllability can be verified as explained in the following theorem.

Theorem 2.6 *The following statements are equivalent:*
1. The system (2.32) *is controllable;*

2. The controllability Gramian $W_c(t)$ is positive definite for every $t > 0$:
3. The controllability matrix $\mathcal{C}$ has full rank:

$$\mathcal{C} = [B\, AB\, A^2B\, \ldots\, A^{m-1}B] \in \mathbb{R}^{n\times nm}$$

Definition 2.4 *The uncontrolled system, e.g. $u \equiv 0$, is called stable, if the real part of the eigenvalues of A are negative. A matrix with this property is said to be stable.*

Definition 2.5 *The system is stabilizable if there exists a state-feedback $u(t) = -Kx(t)$ such that the matrix $A - BK$ is stable.*

Theorem 2.7 *The system (2.32) is stabilizable if and only if the matrix $[A - \lambda I\, B] \in \mathbb{R}^{n\times(n+m)}$ has full row rank for all $\lambda \in \mathbb{C}$ with a negative real part.*

Definition 2.6 *The system (2.32), or the pair (A, C), is called observable if for any $t_1 \in (0, T]$, the initial condition $x_0 \in \mathbb{R}^n$ can be determined from the time history of the input $u(t)$ and the output $y(t)$ in the interval $[0, t_1] \subset [0, T]$.*

Now, we give some conditions to check the observability of the system.

Theorem 2.8 *The following statements are equivalent:*
1. The system is observable;
2. The observability gramian $W_\circ(t)$ is positive definite for every $t > 0$;
3. The observability matrix $\mathcal{O}$ has full rank.

$$\mathcal{O} = [C\, CA\, CA^2\, \ldots CA^{m-1}] \in \mathbb{R}^{nq\times n}$$

We set for infinite horizon problem:

$$W_\circ := \int_0^\infty e^{sA^T} C^T C e^{sA}\, ds \text{ and } W_c := \int_0^\infty e^{sA} BB^T e^{sA^T}\, ds$$

It is proved that W_c and $W_\circ$ can be determined numerically by solving the algebraic *Lyapunov equations:*

$$AW_c + W_cA^T + BB^T = 0 \tag{2.33}$$
$$A^TW_\circ + W_\circ A + C^TC = 0 \tag{2.34}$$

The matrix W_cW_o has nonegative eigenvalues, and the square roots of these eigenvalues define the *Hankel singular values* of system (2.32). The Hankel singular values characterize the importance of the state variables. States of the balanced system corresponding to the small Hankel singular values can be neglected. Thus, the general idea of balance truncation method is to transform the system (2.32) into a *balanced* form and to *truncate* the states that correspond to the small Hankel singular values. In practice, balancing and truncation can be combined by projecting system (2.32) onto the dominant subspaces of the matrix W_cW_o.

Instead of (2.32) we only consider the system for the first $\ell \in \{1, \dots, n\}$ components of z :

$$\begin{cases} \dot{z}_\ell(t) = A^\ell z^\ell(t) + B^\ell u(t),\ z^\ell(0) = {z^\ell}_0 \quad \text{for } t \in (0, +\infty) \\ y^\ell(t) = C^\ell z_\ell(t) \end{cases} \tag{2.35}$$

as explained in Algorithm 2 One disadvantage of this method is that Lyapunov equa-

Algorithm 2: Balance Truncation method (**BT**).

Require: A, B, C, W_c, W_o.

1: Compute the Cholesky factorizazion of the Gramians $W_c = L_c L_c^T$ and $W_o = L_o L_o^T$.

2: Compute Hankel Singular values by SVD:

$$L_c^T L_o = (U_1, U_2) \begin{pmatrix} \Sigma_1 & 0 \\ 0 & \Sigma_2 \end{pmatrix} (V_1, V_2)^T,$$

where $\Sigma_1 = \text{diag}(\sigma_1, \dots, \sigma_\ell)$.

3: Compute the reduce system (2.35) with

$$A^\ell = \mathcal{W}^T A \mathcal{T}, \quad B^\ell = \mathcal{W}^T B, \quad C^\ell = C\mathcal{T},$$

where $\mathcal{W} = L_o V_1 \Sigma_1^{-1/2}, \quad \mathcal{T} = L_c U_1 \Sigma_1^{-1/2}$.

tions, such as (2.33) and (2.34), have to be solved. These matrices equations have to be thought in $\mathbb{R}^{n \times n}$. However, recent results on low rank approximations to the solutions of matrix equation (see [19] and the reference therein) make the balanced truncation model reduction approach attractive for large-scale problems. On the other hand, it is known that balance truncation preserve the structure and the properties of the full dimensional system (see [84]). This very interesting property does not hold in the POD method.
One big advantage of balanced truncation is that *a-priori* error bounds are known thanks to the Hankel singular values. Unfortunately, we do not have a-priori error bounds working with POD since the method is strictly related to the information we have from the system, e.g. snapshots. These bounds are formulated for the transfer function $G(s) := C(sI - A)^{-1}B \in \mathbb{R}^{q \times m}$ of the system (2.32) and the reduced transfer function $G^\ell(s) := C^\ell(sI - A^\ell)^{-1}B^\ell \in \mathbb{R}^{q \times m}$, of (2.35). Then we have

$$\|G - G^\ell\|_{\mathcal{H}_\infty} := \max \left\{ \|(G - G^\ell)u\|_{L^2(0,\infty;\mathbb{R}^q)} : \|u\|_{L^2(0,\infty;\mathbb{R}^m)} = 1 \right\} > \sigma_{\ell+1}.$$

and

$$\|G - G^\ell\| < 2 \sum_{i=\ell+1}^{n} \sigma_i.$$

Thanks to this error bounds we know a-priori the error we get in balance truncation, on the other hand this reduction method is rather restrictive to the class of problem which solves (2.32).
Finally one can see a comparison result between POD and BT methods in Section 4.4.3.

We solved an optimal control problem for heat equation with a quadratic cost functional (see Figure 4.4).

2.3. Reduced Basis Method for Parametrized Problems

Reduced Basis (RB) method is another important tool in the class of model reduction techiniques. The Reduced Basis approach computes efficiently numerical solutions for parametrized problems where it is desired to solve the same problem with different parameter configurations. This method is built upon a classical discretization method. The interested reader can find in [76] and the references therein more informations. In this section we keep the standard notations of RB theory, e.g. $\mathcal{N}$ is the dimension of the discrete model, and N is the dimension of the reduced problem.

2.3.1. Reduced Basis Method

In order to introduce the method we deal with an elliptic equation with arbitrary boundary conditions, considering $\mathcal{D} \subset \mathbb{R}^p$, $p \geq 1$ as the parameter space and $\Omega \in \mathbb{R}^d$ as a domain. The functional space X^e is such that $H_0^1(\Omega) \subset X^e \subset H^1(\Omega)$, with $H^1(\Omega)$ the Sobolev Space (please refer to [34] for more details on partial differential equations) defined as:

$$H^1(\Omega) := \left\{ f \in L^2(\Omega) : D^\alpha f \in L^2(\Omega), \alpha \leq 1 \right\}$$

where f is a measurable function, $D^\alpha f$ denotes the weak $\alpha^{th}-$ partial derivative of f, and

$$L^2(\Omega) := \left\{ f : \Omega \to \mathbb{R}, \int_\Omega f(x)^2 \, dx < \infty \right\}, \quad H_0^1(\Omega) := \left\{ f \in H^1(\Omega) : f \equiv 0 \text{ a.e. on } \partial\Omega \right\}.$$

We introduce $\forall \mu \in \mathcal{D}$ a bilinear and coercive for $a(\cdot, \cdot; \mu)$ and a linear and continuous functional $f(\cdot)$. Then, we consider the following model problem:

$$\begin{cases} \text{For } \mu \in \mathcal{D} \text{ find } u^e \in X^e \text{ s.t} \\ a(u^e, v; \mu) = f(v), \quad \forall v \in X^e. \end{cases} \tag{2.36}$$

The crucial hypothesis is that the bilinear form a can be expressed with an affine linear decomposition:

$$a(w, v; \mu) = \sum_{q=1}^{Q} \Theta^q(\mu) a^q(w, v) \tag{2.37}$$

such that for $\Theta^q : \mathbb{R} \to \mathbb{R}$ for $q = 1, \ldots, Q$ is depending on μ and $a^q : X^e \times X^e \to \mathbb{R}$ is parameter independent. This hypothesis on $a(\cdot, \cdot; \mu)$ allows us to improve the computational efficiency in the evaluation of $a(u, v; \mu)$: the components $a^q(u, v)$ can be computed once and then stored in the so called offline stage of the method. In other words we assume that the problem is affinely dependent on the parameter μ.

We introduce an high dimensional classic discretization in our model problem such that the space $X^{\mathcal{N}} \subset X^e$ and the problem is reformulated as

$$\begin{cases} \text{For } \mu \in \mathcal{D} \text{ find } u^{\mathcal{N}} \in X^{\mathcal{N}} \text{ s.t.} \\ a(u^{\mathcal{N}}, v; \mu) = f(v), \quad \forall v \in X^{\mathcal{N}}. \end{cases} \tag{2.38}$$

We introduce a sample set of parameters $S_N = \{\mu_1, \dots, \mu_N\} \subset \mathcal{D}$ to which we associate the reduced basis space, defined as $W_N^{\mathcal{N}} = \text{span}\{u^{\mathcal{N}}(\mu_n), 1 \leq n \leq N\}$.
To have a system from (2.38) which is computationally stable we can use a Gram-Schmidt orthonormalization (see [47]) procedure for the snapshots $u^{\mathcal{N}}(\mu_n)$, $1 \leq n \leq N$, with respect to the scalar product $\langle \cdot, \cdot \rangle_X$ to obtain $\psi_n^{\mathcal{N}}$ as basis functions, so that

$$X_N^{\mathcal{N}} = \text{span}\{\psi_n^{\mathcal{N}} \text{ s.t. } 1 \leq n \leq N\}. \tag{2.39}$$

Consider that the orthonormalization keeps the linear independence of the basis in fact:

$$\text{span}\{u_n^{\mathcal{N}} \text{ s.t. } 1 \leq n \leq N\} = \text{span}\{\psi_n^{\mathcal{N}} \text{ s.t. } 1 \leq n \leq N\}.$$

Note that usually one chooses $N \ll \mathcal{N}$ in order to reduce the complexity of the problem. As in the POD case (2.24), the reduced space is generated by orthogonal basis functions. The selection of the parameters is typically performed by the Greedy's algorithm (see [86] and more recently [58]).
By a Galerkin projection we can solve the reduced basis problem defined as

$$\begin{cases} \text{For } \mu \in \mathcal{D} \text{ find } u_N^{\mathcal{N}} \in X_N^{\mathcal{N}} \text{ s.t.} \\ a(u_N^{\mathcal{N}}, v; \mu) = f(v) \quad \forall v \in X_N^{\mathcal{N}} \end{cases} \tag{2.40}$$

Similar to (2.22), we can rewrite $u_N^{\mathcal{N}}$ as

$$u_N^{\mathcal{N}} = \sum_{m=1}^{N} u_{N_m}^{\mathcal{N}} \psi_m^{\mathcal{N}}. \tag{2.41}$$

where $\psi_m^{\mathcal{N}}$ is the reduced basis.
Posing $v = \psi_n^{\mathcal{N}}, 1 \leq n \leq N$ in (2.38) we get for every $\mu \in \mathcal{D}$ the numerical linear system:

$$\sum_{m=1}^{N} a(\psi_m^{\mathcal{N}}, \psi_n^{\mathcal{N}}; \mu) u_{N_m}^{\mathcal{N}}(\mu) = f(\psi_n^{\mathcal{N}}) \quad 1 \leq n \leq N \tag{2.42}$$

which by (2.37) can be written as

$$\sum_{m=1}^{N} \left(\sum_{q=1}^{Q} \Theta^q(\mu) a(\psi_m^{\mathcal{N}}, \psi_n^{\mathcal{N}}) \right) u_{N_m}^{\mathcal{N}}(\mu) = f(\psi_n^{\mathcal{N}}) \quad 1 \leq n \leq N. \tag{2.43}$$

Here the basis $\psi^{\mathcal{N}}$ are independent by the parameter μ therefore the quantities $f(\psi_n^{\mathcal{N}})$, $1 \leq$

$n \leq N$ and $a^q(\psi_m^{\mathcal{N}}, \psi_n^{\mathcal{N}})$, $1 \leq n \leq N, 1 \leq q \leq Q$ can be precomputed and stored to decouple the offline computational part from the online one which depends on the parameters. We note the dimension of the system is very small and the online computation for finding every solution of the problem is inexpensive.

2.3.2. Error bound for RB Method.

The main ingredient of the RB method is the error bound which allows either to select efficiently the parameter set either to give a prediction of the accuracy of the method. The error bound takes advantage from the online/ offline stage since it is fast and reliable. We introduce the a *posteriori* error bounds. We reconsider the high dimensional discrete problem (2.38) and the Galerkin projection to get the reduced problem (2.40). We define $e(\mu) \equiv u^{\mathcal{N}}(\mu) - u_N^{\mathcal{N}}(\mu) \in X^{\mathcal{N}}$. Thanks to the linearity of $a(\cdot, \cdot; \mu)$ we have

$$\begin{aligned} a(e(\mu), v; \mu) &= a(u^{\mathcal{N}}(\mu), v; \mu) - a(u_N^{\mathcal{N}}(\mu), v; \mu) \\ &= f(v) - a(u_N^{\mathcal{N}}(\mu), v; \mu) \quad \forall v \in X^{\mathcal{N}}. \end{aligned}$$

We denote

$$r(v; \mu) := f(v) - a(u_N^{\mathcal{N}}(\mu), v; \mu), \tag{2.44}$$

to get the equation of the residual

$$a(e(\mu), v; \mu) = r(v; \mu).$$

Here $r(v; \mu) \in X^{\mathcal{N}'}$, and thanks to Riesz representation theorem we can write $r(v; \mu)$ as

$$r(v; \mu) = \langle \hat{e}(\mu), v)_X \quad \forall v \in X^{\mathcal{N}},$$

in fact we have $\|r(\cdot; \mu)\|_{(X^{\mathcal{N}})'} \equiv \frac{r(v;\mu)}{\|v\|_X} = \|\hat{e}(\mu)\|_X$. By the coercivity of the bilinear form $a(u, v; \mu)$ we define the following error bound:

$$\Delta_N(\mu) = \frac{\|\hat{e}(\mu)\|}{\sqrt{\alpha_{LB}^{\mathcal{N}}(\mu)}}$$

where $\alpha_{LB}^{\mathcal{N}}$ is the coercivity lower bound of the bilinear form and may be computed with a Successive Constraint Method (SCM) algorithm (see [76, 74])).
In order to build the space $W_N^{\mathcal{N}}$ we start considering $S_1 = \{\mu_1\}$. Then we look for

$$\mu_N = \arg \max_{\mu \in \mathcal{D}} \Delta_{N-1}(\mu).$$

The reduced basis method has been developed for parametrized elliptic PDEs [76] and successfully applied to Stokes [85] and Navier-Stokes equations. This method has been used for time-dependent problems such as in [49].

3. An efficient Policy Iteration Algorithms for Dynamic Programming Equations

We present an accelerated algorithm for the solution of static Hamilton-Jacobi-Bellman equations related to optimal control problems. Our scheme is based on a classic policy iteration procedure, which is known to have superlinear convergence in many relevant cases provided the initial guess is sufficiently close to the solution. In many cases, this limitation degenerates into a behavior similar to a value iteration method, with an increased computations time. The new scheme circumvents this problem by combining the advantages of both algorithms with an efficient coupling. The method starts with a value iteration phase and then switches to a policy iteration procedure when a certain error threshold is reached. A delicate point is to determine this threshold in order to avoid cumbersome computation with the value iteration and, at the same time, to be reasonably sure that the policy iteration method will finally converge to the optimal solution. We analyze the methods and efficient coupling in a number of examples in dimension two, three and four illustrating their properties. The original elements of this chapter are presented in [4].

3.1. Dynamic programming in optimal control and the basic solution algorithms

In this section we will summarize the basic results for the two methods as they will constitute the building blocks for our new algorithm. The essential features will be briefly sketched, and more details can be found in the original papers and in some monographs, e.g. in the classical books by Bellman [18], Howard [57] and for a more recent setting in framework of the viscosity solutions in [26], [15] and [38].
Let us first present the method for the classical *infinite horizon problem*. Let the dynamics be given by

$$\begin{cases} \dot{y}(t) = f(y(t), u(t)), & t \geq 0 \\ y(0) = x \end{cases} \tag{3.1}$$

where $y \in \mathbb{R}^d$, $u \in \mathbb{R}^m$ and $u \in \mathcal{U} \equiv \{u : [0, +\infty) \to U, \text{ measurable}\}$. If f is Lipschitz continuous with respect to the state variable and continuous with respect to (y, u), the classical assumptions for the existence and uniqueness result for the Cauchy problem (3.1) are satisfied. To be more precise, the Carathéodory theorem (see [43] or [15]) implies that for any given control $u(\cdot) \in \mathcal{U}$, there exists a unique trajectory $y(\cdot; u)$ satisfying (3.1) almost everywhere. Changing the control policy the trajectory will change and we

will have a family of infinitely many solutions of the controlled system (3.1) parametrized with respect to u.
Let us introduce the *cost functional* $J : \mathcal{U} \to \mathbb{R}$ which will be used to select the "optimal trajectory". For infinite horizon problem the functional is

$$J_x(u(\cdot)) = \int_0^\infty L(y(s), u(s))e^{-\lambda s}ds\,, \tag{3.2}$$

where L is Lipschitz continuous in both arguments and $\lambda > 0$ is a given parameter. The function L represents the running cost and $\lambda > 0$ is the discount factor which allows to compare the costs at different times rescaling the costs at time 0. From the technical point of view, the presence of the discount factor guarantees that the integral is finite whenever L is bounded, i.e. $||L||_\infty \leq M_L$. Let us define the *value function* of the problem as

$$v(x) = \inf_{u(\cdot)\in\mathcal{U}} J_x(u(\cdot))\,. \tag{3.3}$$

It is well known that passing to the limit in the Dynamic Programming Principle one can obtain a characterization of the value function in terms of the following first order non-linear Bellman equation

$$\lambda v(x) + \max_{u\in U}\{-f(x,u)\cdot Dv(x) - L(x,u)\} = 0, \quad \text{for } x \in \mathbb{R}^d\,. \tag{3.4}$$

Several approximation schemes on a fixed grid G have been proposed for (3.4). Here we will use a semi-Lagrangian approximation based on a Discrete Time Dynamic Programming Principle. This leads to

$$v_{\Delta t}(x) = \min_{u\in U}\{e^{-\lambda\Delta t}v_{\Delta t}\left(x + \Delta t f\left(x,u\right)\right) + \Delta t L\left(x,u\right)\}\,, \tag{3.5}$$

where $v_{\Delta t}(x)$ converges to $v(x)$ when $\Delta t \to 0$. A natural way to solve (3.5) is to write it in fixed point form (see [38] for more details) as in the following algorithm:

Algorithm 3: Value Iteration for infinite horizon optimal control **(VI)**

Data: Mesh G, Δt, initial guess V^0, tolerance ϵ.
while $||V^{k+1} - V^k|| \geq \epsilon$ **do**
  forall the $x_i \in G$ **do**

$$V_i^{k+1} = \min_{u\in U}\{e^{-\lambda\Delta t}\mathcal{I}\left[V^k\right]\left(x_i + \Delta t f\left(x_i,u\right)\right) + \Delta t L\left(x_i,u\right)\} \tag{3.6}$$

  end
  $k = k + 1$
end

Here V_i^k represents the values at a node x_i of the grid at the k-th iteration and $\mathcal{I}$ is an interpolation operator acting on the values of the grid; without loss of generality, throughout this chapter we will assume that the numerical grid G is a regular equidistant array of points with mesh spacing denoted by Δx, and we consider a multilinear

interpolation operator. Extensions to nonuniform grids and high-order interpolants can be performed in a straightforward manner.
Algorithm 3 is referred in the literature as the *value iteration method* because, starting from an initial guess V^0, it modifies the values on the grid according to the nonlinear rule (3.6). It is well-known that the convergence of the value iteration can be very slow, since the contraction constant $e^{-\lambda\Delta t}$ is close to 1 when Δt is close to 0. This means that a higher accuracy will also require more iterations. Then, there is a need for an acceleration technique in order to cut the link between accuracy and complexity of the value iteration.

For sake of clarity, the above framework has been presented for the infinite horizon optimal control problem. However, similar ideas can be extended to other classical control problems with small changes. Let us mention how to deal with the minimum time problem which we will use in the final section on numerical tests.
In the minimum time problem one has to drive the controlled dynamical system (3.1) from its initial state to a given target $\mathcal{T}$. Let us assume that the target is a compact subset of $\mathbb{R}^d$ with non empty interior and piecewise smooth boundary. The major difficulty dealing with this problem is that the time of arrival to the target starting from the point x

$$t(x,u(\cdot)) := \begin{cases} \inf\limits_{u\in\mathcal{U}} \{t \in \mathbb{R}_+ : y(t,u(\cdot)) \in \mathcal{T}\} & \text{if } y(t,u(t)) \in \mathcal{T} \text{ for some } t, \\ +\infty & \text{otherwise,} \end{cases} \tag{3.7}$$

can be infinite at some points. As a consequence, the minimum time function defined as

$$T(x) = \inf_{u\in\mathcal{U}} t(x,u(\cdot)) \tag{3.8}$$

is not defined everywhere if some controllability assumptions are not introduced. In general, this is a free boundary problem where one has to determine at the same time, the couple (T,Ω), i.e. the minimum time function and its domain. Nevertheless, by applying the Dynamic Programming Principle and the so-called Kruzkhov transform

$$v(x) \equiv \begin{cases} 1-\exp(-T(x)) & \text{for } T(x) < +\infty \\ 1 & \text{for } T(x) = +\infty \end{cases} \tag{3.9}$$

the minimum time problem is characterized in terms of the unique viscosity solution of the BVP

$$\begin{cases} v(x) + \sup\limits_{u\in\mathcal{U}}\{-f(x,u)\cdot Dv(x)\} = 1 & \text{in } \mathcal{R}\backslash\mathcal{T} \\ v(x) = 0 & \text{on } \partial\mathcal{T}, \end{cases} \tag{3.10}$$

where $\mathcal{R}$ stands for the set of point in the state space where the time of arrival is finite. Then, the application of the semi-Lagrangian method presented for the infinite horizon optimal control problem together with a value iteration procedure leads to following iterative scheme:

The numerical implementation is closed with the boundary conditions $v(x) = 0$ at $\partial\mathcal{T}$

Algorithm 4: Value Iteration for minimum time optimal control **(VI)**

Data: Mesh G, Δt, initial guess V^0, tolerance ϵ.
while $||V^{k+1} - V^k|| \geq \epsilon$ **do**
  forall the $x_i \in G$ **do**

$$V_i^{k+1} = \min_{u \in U} \{ e^{-\Delta t} \mathcal{I}\left[V^k\right](x_i + \Delta t f(x_i, u)) + 1 - e^{-\Delta t} \} \tag{3.11}$$

  end
  $k = k + 1$
end

(and inside the target as well), and with $v(x) = 1$ at other points outside the computational domain (we refer the reader to [14] for more details on the approximation of minimum time problems). Next chapter will focus on finite horizon optimal control problems. Therefore, we will introduce an evolutive HJB and a value-iteration scheme for the approximation of that equation.

3.1.1. Two acceleration techniques

We will briefly describe two effective acceleration methods based on different ideas and resulting in a monotone convergence.

Policy iteration

The first acceleration technique is the *approximation in the policy space* (or policy iteration), and is based on a linearization of the Bellman equation. First, an initial guess for the control for every point in the state space is chosen. Once the control has been fixed, the Bellman equation becomes linear (no search for the minimum in the control space is performed), and it is solved as an advection equation. Then, an updated policy is computed and a new iteration starts. Let us sketch the procedure for the scheme related to the infinite horizon problem.

Algorithm 5: Policy Iteration for infinite horizon optimal control **(PI)**

Data: Mesh G, Δt, initial guess V^0 and u^0, tolerance ϵ.
while $||V^{k+1} - V^k|| \geq \epsilon$ **do**
  Policy evaluation step:
  forall the $x_i \in G$ **do**

$$V_i^{k+1} = \Delta t L\left(x_i, u_i^k\right) + e^{-\lambda \Delta t} \mathcal{I}\left[V^k\right]\left(x_i + \Delta t f\left(x_i, u_i^k\right)\right) \tag{3.12}$$

  end
  Policy improvement step:
  forall the $x_i \in G$ **do**

$$u_i^{k+1} = \arg\min_u \left\{\Delta t L(x_i, u) + e^{-\lambda \Delta t} \mathcal{I}\left[V^k\right](x_i + \Delta t f(x_i, u))\right\} \tag{3.13}$$

  end
  $k = k + 1$
end

Note that the solution of (3.12) can be obtained either by a linear system (assuming a linear interpolation operator) or as the limit

$$V^k = \lim_{m \to +\infty} V^{k,m}, \tag{3.14}$$

of the linear time-marching scheme

$$V_i^{k,m+1} = \Delta t L\left(x_i, u_i^k\right) + e^{-\lambda \Delta t} \mathcal{I}\left[V^{k,m}\right]\left(x_i + \Delta t f\left(x_i, u_i^k\right)\right). \tag{3.15}$$

Although this scheme is still iterative, the lack of a minimization phase makes it faster than the original value iteration.
The sequence $\{V^k\}$ turns out to be monotone decreasing at every node of the grid. In fact, by construction,

$$\begin{aligned} V_i^k &= \Delta t L\left(x_i, u_i^k\right) + e^{-\lambda \Delta t} \mathcal{I}\left[V^k\right]\left(x_i + \Delta t f\left(x_i, u_i^k\right)\right) \geq \\ &\geq \min_u \left\{\Delta t L(x_i, u) + e^{-\lambda \Delta t} \mathcal{I}\left[V^k\right](x_i + \Delta t f(x_i, u))\right\} = \\ &= \Delta t L\left(x_i, u_i^{k+1}\right) + e^{-\lambda \Delta t} \mathcal{I}\left[V^k\right]\left(x_i + \Delta t f\left(x_i, u_i^{k+1}\right)\right) = \\ &= V_i^{k+1} \end{aligned}$$

At a theoretical level, policy iteration can be shown to be equivalent to a Newton method, and therefore, under appropriate assumptions, it converges with quadratic speed. On the other hand, convergence is local and this may represent a drawback with respect to value iterations.

Iteration in the set of subsolutions

A different monotone acceleration technique can be constructed upon the idea of following the direction provided by the value iteration up to the boundary of the set of (numerical) subsolutions. This idea is justified by the property of the value function of being the maximal subsolution of the dynamic programming equation (We address to [38] for further details on the algorithm).
Consider the fixed point iteration associated to the infinite horizon problem, written in short as

$$V^{(k+1)} = S(\Delta t, V^{(k)}).$$

Since the operator S is a contraction mapping, we have convergence starting from any initial guess $V^{(0)}$. Moreover, if the initial guess is in the set of subsolutions Σ for the discrete problem, i.e., if

$$V^{(0)} \in \Sigma := \{W : W \leq S(\Delta t, W)\},$$

then convergence is monotone. In fact, by the monotonicity of S,

$$V^{(k)} \leq S(\Delta t, V^{(k)}) = V^{(k+1)}.$$

The set of subsolutions Σ can be proved to be a closed convex set with a maximal element, which coincides with the fixed point V. Then, we can use the operator S to obtain a search direction (along which the components of the numerical solution increase), until we reach the boundary of Σ.

Algorithm 6: Accelerated Monotone Value Iteration **(AMVI)**

Data: Mesh G with n nodes, Δt, initial guess $W^0 \in \Sigma$, tolerance ϵ.
while $||W^{k+1} - W^k|| \geq \epsilon$ **do**
  $W^{k+1/2} = S(\Delta t, W^k)$
  $d^k = W^{k+1/2} - W^k$
  $W^{k+1} = \max_{\mu}\{W^k + \mu d^k\}$ such that $W^k + \mu d^k \in \Sigma$
  $k = k + 1$
end

In this case, the accelerated sequence W^k converges monotonically and for any initial guess. However, the application of this acceleration technique is limited by the assumptions we made. In some cases it can be difficult to find an initial condition for the sequence. Another limitation is the fact that the set of subsolutions is not always convex as in the infinite horizon problem so the rate of acceleration decreases fast (as in the minimum time problem).

3.2. An accelerated policy iteration algorithm with smart initialization

In this section we present an accelerated iterative algorithm which is constructed upon the building blocks previously introduced. We aim to an efficient formulation exploiting the main computational features of both value and policy iteration algorithms. As it has been stated in [79], there exists a theoretical equivalence between both algorithms, which guarantees a rather wide convergence framework. However, from a computational perspective, there are significant differences between both implementations. A first key factor can be observed in Figure 3.1, which shows, for a two-dimensional minimum time problem (more details on the test can be found in Section 3.4), the typical situation arising with the evolution of the error measured with respect to the optimal solution, when comparing value and policy iteration algorithms. To achieve a similar error level, policy iteration requires considerable fewer iterations than the value iteration scheme, as quadratic convergent behavior is reached faster for any number of nodes in the state-space grid. Despite the observed computational evidence, a second issue is observed when

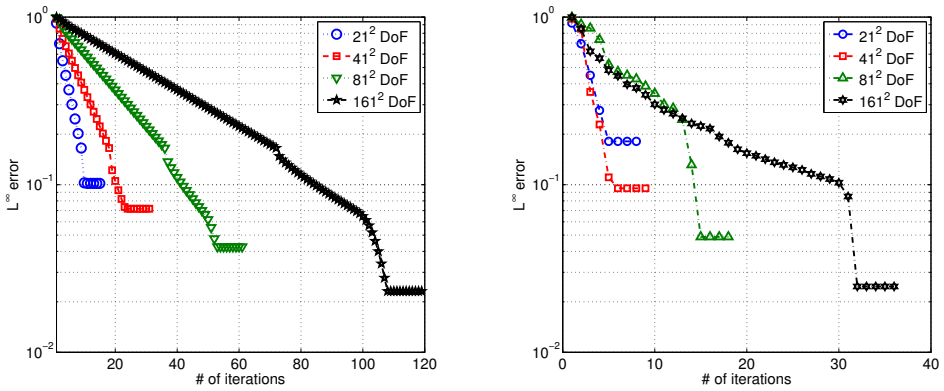

Figure 3.1.: Error evolution in a 2D problem: value iteration (left) and policy iteration (right).

examining the policy iteration algorithm in more detail. That is, as shown in Figure 3.2, the sensitivity of the method with respect to the choice of the initial guess of the control field. It can be seen that different initial admissible control fields can lead to radically different convergent behaviors. While some guesses will produce quadratic convergence from the beginning of the iterative procedure, others can lead to an underperformant value iteration-like evolution of the error. This latter is computationally costly, because it translates into a non-monotone evolution of the subiteration count of the solution of equation (3.12).

A final relevant remark goes back to Figure 3.1, where it can be observed that for coarse meshes, the value iteration algorithm generates a fast error decay up to a higher global error. Combining the fast error decay with the fact that value iteration algorithms are

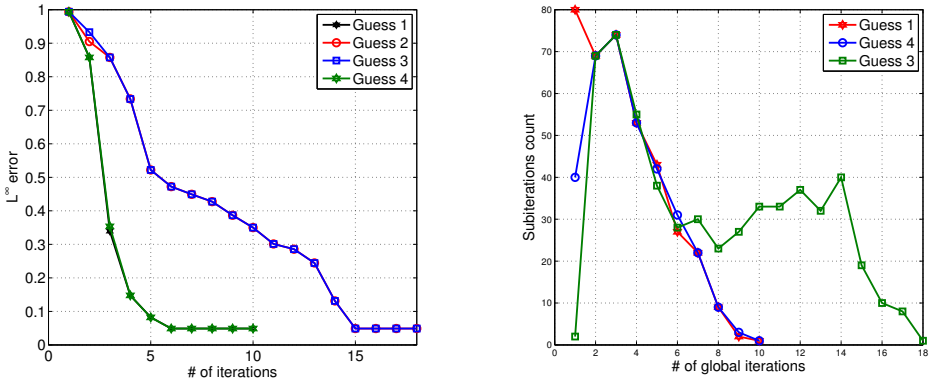

Figure 3.2.: Left: error evolution in a PI algorithm for different initial guesses. Right: evolution of the subiteration count in PI for different guesses.

rather insensitive to the choice of the initial guess for the value function (see [89] for a detailed error quantification), are crucial points for the construction of our accelerated algorithm. The accelerated policy iteration algorithm is based on a robust initialization of the policy iteration procedure via a coarse value iteration which will yield to a good guess of the initial control field.

Algorithm 7: Accelerated Policy Iteration **(API)**

Data: Coarse mesh G_c and Δt_c, fine mesh G_f and Δt_f, initial coarse guess V_c^0, initial control u^0, coarse-mesh tolerance ϵ_c, fine-mesh tolerance ϵ_f.

begin

Coarse-mesh value iteration step: perform Algorithm 3

Input: G_c, Δt_c, V_c^0, ϵ_c

Output: V_c^*

forall the $x_i \in G_f$ **do**

$V_f^0(x_i) = \mathcal{I}_1[V_c^*](x_i)$

$U_f^0(x_i) = \arg\min_{u \in U} \{e^{-\lambda \Delta t} \mathcal{I}_1[V_f^0](x_i + f(x_i, u)) + \Delta t L(x_i, u)\}$

end

Fine-mesh policy iteration step: perform Algorithm 5

Input: G_f, Δt_f, V_f^0, U_f^0, ϵ_f

Output: V_f^*

end

3.2.1. Practical details concerning the computational implementation of the algorithm

The above presented accelerated algorithm can lead to a considerably improved performance when compared to value iteration and naively initialized policy iteration algo-

rithms. However, it naturally contains trade-offs that need to be carefully handled in order to obtain a correct behavior. The extensive numerical tests performed in Section 3.4 suggest the following guidelines:

Coarse and fine meshes. The main trade-off of the accelerated algorithm is related to this point. For a good behavior of the PI part of the algorithm, a good initialization is required, but this should be obtained without deteriorating the overall performance. Too coarse VI will lead to poor initialization, while fine VI will increase the CPU time. We recall that for this chapter we assume regular equidistant meshes with mesh parameter Δx. If we denote by Δx_c and by Δx_f the mesh parameters associated to the coarse and fine grids respectively, numerical findings illustrated in Figure 3.3 suggest that for minimum time problems and infinite horizon optimal control, a good balance is achieved with $\Delta x_c = 2\Delta x_f$. In the case of minimum time problem, additionally, it is important that the coarse mesh is able to accurately represent the target. Van Der Pol example has a different behavior, but it is important to observe that, although the choice of the stepsize is suboptimal, it is still close to the optimal ratio.

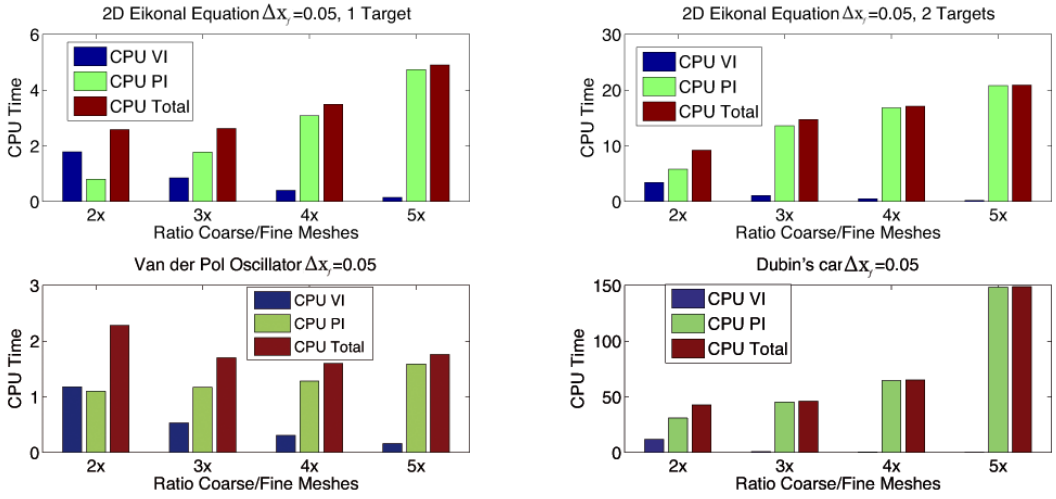

Figure 3.3.: Ratios $\Delta x_c/\Delta x_f$ and CPU time for different control problems. A good overall balance can be observed in most cases by considering $\Delta x_c = 2\Delta x_f$.

Accuracy. Both VI and PI algorithms require a stopping criteria for convergence. Following [90], the stopping criteria is given by

$$||V^{k+1} - V^k|| \leq C\Delta x^2\,,$$

which relates the error to the resolution of the state-space mesh. The constant C is set to $C = \frac{1}{5}$ for the fine mesh, and for values ranging from 1 to 10 in the coarse mesh, as we do not strive for additional accuracy that usually will not improve the initial guess of the control field. However, different options have been extensively discussed in the literature, as in [91] for instance, where the stopping criteria is related to a variability

threshold on the control space.

Policy evaluation. In every main cycle of the policy iteration algorithm, provided the interpolation operator is linear, as it is in our case, a solution of the linear system (3.12) is required. This can be performed in several ways, specially given the sparsity of the system. For sake of simplicity and in order to make numerical comparisons with the VI scheme, we use a fixed point iteration, i.e., the policy evaluation is implemented as

$$V_i^{k,j+1} = \Delta t L\left(x_i, u_i^k\right) + e^{-\lambda \Delta t}\mathcal{I}\left[V^{k,j}\right]\left(x_i + \Delta t f\left(x_i, u_i^k\right)\right) \tag{3.16}$$

with initial guess $V^{k,0} = V^{k-1,\infty}$. We use the same stopping criteria as for the global iteration.

Minimization. Although counterexamples can be constructed in order to show that it is not possible to establish error bounds of the PI algorithm independently of the number of controls [90], the algorithm does not change its performance when the control set is increased, and therefore the argmin computation required for the policy update can be performed by discretizing the set of controls and evaluating all the possible arrival points. Otherwise, minimizers can be computed via Brent's algorithm, as in [27].

A remark on parallelism. Although the numerical tests that we present were performed in a serial code, we note that the accelerated algorithm allows an easy parallel implementation. Whenever an iterative procedure is performed over the value function, parallelism can be implemented via a domain decomposition of the state space as in [42, 24]. If a control space discretization is also performed, the policy update (3.13) can also be parallelized with respect of the set of controls.

Error estimates. The Policy Iteration scheme may be considered as a Newton-like algorithm (as explained in [79]) which is shown to have local quadratic convergence, as shown in [90]. A delicate point is the initialization of the algorithm. The idea is to compute the value function V^{Δ} in a coarse mesh computed via the VI scheme where the stepsize is Δ. Then we pass to a finer mesh $\Delta/2$, and by linear interpolation we obtain $V^{\Delta/2}$. The grid is built as $x = (\ldots, x_{i-1}, x_{i-1/2}, x_i, x_{i+1/2}, x_{i+1}, \ldots)$ such that $x_i - x_{i-1} = \Delta$ and $x_i - x_{i-1/2} = \Delta/2$. Therefore, $V_i^{k,\Delta/2} = V_i^{k,\Delta}$ and $V_{i+1/2}^{k,\Delta/2} = (V_i^{k,\Delta} + V_{i+1}^{k,\Delta})/2$, where $V_i^{k,\Delta} \approx V(x_i)$. In this way, we can guarantee to have the same accuracy of the coarse grid. In fact, if we suppose that $\|V^{k+1,\Delta} - V^{k,\Delta}\| \leq \varepsilon$, then

$$\|V_{i+1/2}^{k+1,\Delta/2} - V_{i+1/2}^{k,\Delta/2}\| = \frac{1}{2}\left\|\left(V_{i+1}^{k+1,\Delta} - V_{i+1}^{k,\Delta}\right) + \left(V_i^{k+1,\Delta} - V_i^{k,\Delta}\right)\right\|$$

and by triangular inequality, the value function holds the desired tolerance ε. The controls are simply obtained computing the argmin of $V^{k+1,\Delta/2}$. Hence, we suppose that we are in the neighborhood of the solution V^*.

3.3. An error estimate for API

Let us try to obtain a theoretical estimate for the error related to the API algorithm. Since the method is based on the coupling of a VI approximation with a PI approximation, we need to couple the general error estimate for VI [38] and the local error estimates for the Newton method. The crucial point is to decide when to switch from one method to the other. First of all, let us note that we can rewrite relation (3.5) with compact notations:

$$V = T(V),$$

where the map $T : \mathbb{R}^M \to \mathbb{R}^M$ is defined componentwise as

$$(T(V))_i \equiv \min_{u \in U} [e^{-\lambda \Delta t} \Lambda(u) V + \Delta t L(u)]_i,$$

where $V_i = v(x_i), L_i(u) = L(x_i, u)$, and $\Lambda(u) \in \mathbb{R}^{M \times M}$ matrix and we have:

$$v(x_i + \Delta t f(x_i, u)) = \sum_{j=1}^{M} \lambda_{ij}(u) v(x_j), \quad i = 1, \ldots, M,$$

where $\lambda_{ij}(u)$ are the coefficients of the convex combination representing the point $x_i + \Delta t f(x_i, u)$.
The following theorem gives an estimate of the error of the value function, in the Value Iteration algorithm (proof and other details can be found in [38]):

Theorem 3.1 *Let V and $V_{\Delta t}$ be the solutions of (3.4) and (3.5). Assume that:*

$$f : \mathbb{R}^d \times U \to \mathbb{R}^d \text{ and } L : \mathbb{R}^d \times U \to \mathbb{R} \text{ are continuous,}$$

$$\|f(x, u) - f(y, u)\| \leq K_f \|x - y\| \text{ for any } u \in U \text{ and } \|f\|_\infty \leq M_f,$$

$$\|L(x, u) - L(y, u)\| \leq K_L \|x - y\| \text{ for any } u \in U \text{ and } \|L\|_\infty \leq M_L,$$

where $M_L, M_f, K_L > 0, \lambda > K_f$. Moreover, if we assume $x_i + \Delta t f(x_i, u) \in G$, the following inequality holds:

$$\|V - V_{\Delta t}\|_\infty \leq C(\Delta t)^{1/2} + \frac{K_f}{\lambda(\lambda - K_f)} \frac{\Delta x}{\Delta t}. \tag{3.17}$$

Note that $\|V(y) - V_{\Delta t}(y)\|_\infty := \max_{y \in G} |V(y) - V_{\Delta t}(y)|$. Since T is a contraction mapping in $\mathbb{R}^M$, the sequence for a given initial condition V^0

$$V_{\Delta t}^n = T(V_{\Delta t}^{n-1}), \quad n = 1, 2, \ldots$$

will converge to V^* for any initial condition $V_{\Delta t}^0 \in \mathbb{R}^d$. Moreover, under the assumptions of Theorem 3.1 we have the following relation between the current iteration $V_{\Delta t}^n$ and the

initial condition $V^0_{\Delta t}$:

$$\|V^n_{\Delta t} - V^*_{\Delta t}\|_\infty \leq (e^{-n\lambda\Delta t})\|V^0_{\Delta t} - V^*_{\Delta t}\|_\infty. \tag{3.18}$$

The Policy Iteration scheme may be considered as a Newton-like algorithm (as explained in [79]) which is shown to have quadratic convergence in the following theorem proved by Santos and Rust in [90].

Theorem 3.2 *Assume V^* is the fixed point of the discretized Bellman equation* (3.5)*. Assume that $V_{\Delta t}(x_i, u)$ is concave in x_i. Let $\{V^n_i\}_{n\geq 1}$ be a sequence of functions generated by* (3.12) *and* (3.13)*. Moreover if the initial guess is close to the solution V^*, we have:*

$$\|V^{n+1}_{\Delta t} - V^*_{\Delta t}\| \leq C\frac{\lambda}{\Delta t^2(1-\lambda)}\|V^n_{\Delta t} - V^*\|^2.$$

Combining Theorem 3.1 and Theorem 3.2 we get an estimate for our accelerated Policy Iteration algorithm. For simplicity we drop the subscript Δt.
We call V^n_{API} the current iteration of the accelerated scheme, V^0_{VI} is the initial condition and we suppose that the VI method converges to V^*_{VI}. Suppose we need γ iterations to get the desidered convergence in the VI method. Then, we have:

$$\|V^n_{API} - V^*\| \leq e^{\gamma\lambda\Delta t}\|V^0_{VI} - V^*_{VI}\| + \left(C\frac{\lambda}{\Delta t^2(1-\lambda)}\right)^{n-\gamma}\|V^*_{VI} - V^*\|^2. \tag{3.19}$$

Since we are working with a Newton-like method, the choice of the initial condition has a crucial role. Therefore we need to check the hypothesis given in Appendix A in order to guarantee convergence. Note that we can rewrite the problem in the following form:

$$F(V) := 0 \iff \text{V satisfies (3.5)}.$$

Hence, $F(V) := V - T(V)$ and we can easily write it as a Newton method, if we assume all the next computations make sense, we get:

$$J_F(V^n)(V^{n+1} - V^n) = -F(V^n),$$

where the Jacobian of F(V) is $J_F(V) := I - T'(V)$, and I is the identity matrix.

Let us fix the residual r^n and the error e^n at the $n-$th iteration:

$$r^n \equiv V^n - T(V^n) = V^n - V^{n-1} \qquad e^n \equiv \|V^n - V^*\|_\infty.$$

Then, we fix $\eta > 0$ such that:

$$\|r^n\| \leq \eta \iff \|V^*_{VI} - V^*\| \leq \rho.$$

An explicit expression for ρ is given in (A.5). Then it is required:

$$\|F'(V^*_{VI})^{-1}F(V^*_{VI})\| \leq \alpha,$$

which is guarenteed since V_{VI}^* is the solution coming from the value iteration scheme which is known to be convergent for any initial condition. If finally we assume that the following inequality holds,

$$\|F'(V_{VI}^*)^{-1}(F'(y) - F'(x)\| \leq \bar{\omega}\|y - x\|,$$

we have proved the theorem given below

Theorem 3.3 *Assume all the assumptions in Theorem 3.1, Theorem 3.2 and Theorem A.1 hold. If the initial conditions of the Policy Iteration is V_{VI}^*, we get the following estimate for API:*

$$\|V_{API}^n - V^*\|_i nfty \leq e^{\gamma\lambda\Delta t}\|V_{VI}^0 - V_{VI}^*\|_i nfty + \left(C\frac{\lambda}{\Delta t(1-\lambda}\right)^{n-\gamma}\|V_{VI}^* - V^*\|_i nfty^2. \tag{3.20}$$

3.4. Numerical tests

This section presents a comprehensive set of tests assessing the performance of the proposed accelerated algorithm. We compare the results with solutions given by the classical value iteration algorithm, policy iteration, and the accelerated monotone value iteration method. In a first part we develop tests related to infinite horizon optimal control, to then switch to the study of minimum time problems. We conclude with an extension to applications related to optimal control of partial differential equations. We focus on grid resolution, size of the discretized control space, performance in presence of linear/nonlinear dynamics, targets, and state space dimension. All the numerical simulations reported in this chapter have been made on a MacBook Pro with 1 CPU Intel Core i5 2.3 Ghz and 8GB RAM.

Infinite horizon optimal control problems

3.4.1. Test 1: A non-smooth 1D value function

We first consider a one-dimensional optimal control problem appearing in [15, Appendix A]. Using a similar notation as in Section 3.1, we set the computational domain $\Omega =]-1\,,1[$, the control space $U = [-1\,,1]$, the discount factor $\lambda = 1$, the system dynamics $f(x,u) = u(1-|x|)$, and the cost function $L(x,a) = 3(1-|x|)$. The exact optimal solution for this problem is

$$v(x) = \begin{cases} \frac{3}{2}(x+1) & \text{for } x < 0\,, \\ \frac{3}{2}(1-x) & \text{elsewhere}\,, \end{cases}$$

which has a kink at $x = 0$. We implement every proposed algorithm, and results concerning CPU time and number of iterations are shown in Table 3.3; for different mesh configurations, we set $\Delta t = .8\Delta x$ and we discretize the control space into a set of 20

equidistant points. The notation VI($2\Delta x$) in Table 3.3 stands for the computation of the soluzion with a VI method considering a coarse grid of $2\Delta x$. Then it is applied the PI method with a stepsize Δx(PI(Δx) in the table). In this test case, as expected, we observe that the VI algorithm is always the slowest option, with iteration count depending on the number of mesh nodes; this feature is also observed for the PI algorithm, although the number of iterations and CPU time are considerably smaller. On the other hand, the AMVI scheme has an iteration count independent of the degrees of freedom of the system, with an almost fixed CPU time, as the time spent on fixed point iterations is negligible compared to the search of the optimal update direction. In this particular example, the exact boundary conditions of the problem are known ($v(x) = 0$ at $\partial\Omega$) and it is possible to construct monotone iterations by starting from the initial guess $v(x) = 0$. Finally, the API algorithm exhibits comparable CPU times as AMVI, performing always better than VI and PI. We note that the choice of the mesh ratio between the coarse and fine meshes can be suboptimal, as the time spent on the VI coarse pre-processing represents an important part of the overall CPU time. More details on the error evolution throughout the iterations can be observed in Figure 3.4; note that the error evolution is measured with respect to the exact solution and not with respect to the next iteration. From this latter figure it can be seen that precomputation of an optimal solution over a coarse mesh leads to faster PI convergence.

# nodes	Δx	VI	PI	AMVI	VI($2\Delta x$)	PI(Δx)	API
41	0.5	1.92 (101)	0.56 (10)	0.86 (3)	0.11 (6)	0.12 (2)	0.23 (8)
81	2.5E-2	4.36 (229)	1.24 (18)	0.87 (3)	0.44 (3)	0.21 (2)	0.65 (5)
161	1.25E-2	9.82 (512)	2.64 (34)	0.88 (3)	1.37 (73)	0.38 (2)	1.75 (75)
321	6.25E-3	21.91 (1135)	5.77 (65)	0.89 (3)	3.79 (200)	0.73 (2)	4.52 (202)

Table 3.3.: Test 1 (1D non-smooth value function): CPU time (iterations) for different algorithms.

3.4.2. Test 2: Van Der Pol oscillator

In a next step we consider two-dimensional, nonlinear system dynamics given by the Van der Pol oscillator:

$$f(x, y, u) = \begin{pmatrix} y \\ (1 - x^2)y - x + u \end{pmatrix} .$$

Remaining system parameters are set:

$$\Omega =]-2\,,2[^2\,, \quad U = [-1\,,1]\,, \quad \lambda = 1\,, \quad \Delta t = 0.3\Delta x\,, \quad g(x, y, u) = x^2 + y^2\,,$$

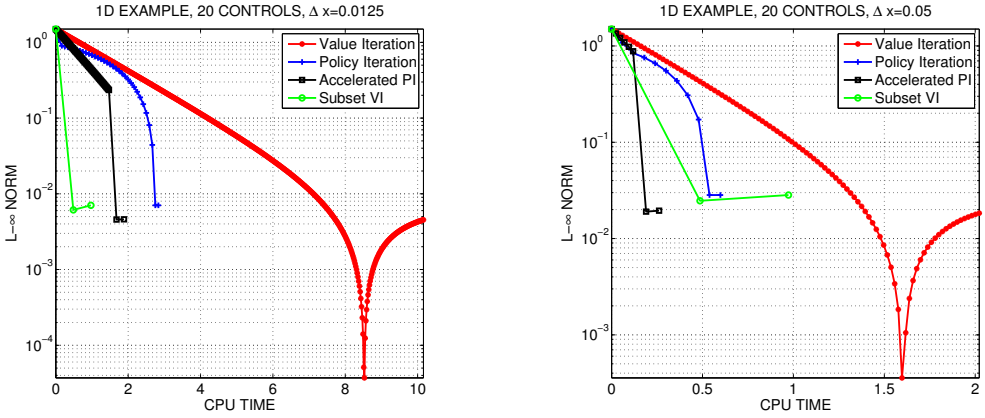

Figure 3.4.: Test 1 (1D non-smooth value function): error evolution for two meshes.

and the control space is discretized into 32 equidistant points. We perform a similar numerical study as in the previous example, and results are shown in Table 3.5. For computations requiring an exact solution, we consider as a reference a fine grid simulation with $\Delta x = 0.00625$ as in Figure 3.5.

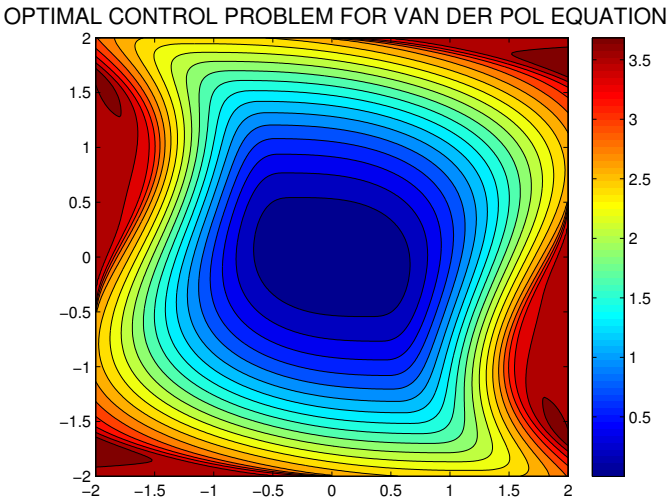

Figure 3.5.: Test 2: Contour plot of the value function for the Van Der Pol oscillator problem.

We set a constant boundary value $v(x) = 3.5$ at $\partial\Omega$, which can be interpreted as a penalization on the state. If accurate solutions near the boundary are required, a natural choice in our setting would be to perform simulations over an enlarged domain and then restrict the numerical results to a subset of interest. From this test we observe a serious limitation on the AMVI algorithm. The number of iterations now depends on the number

of nodes, and even though the number of iterations is still lower than in the VI algorithm, the CPU time increases as for every iteration a search procedure is required. As it is not possible to find monotone update directions, the AMVI algorithm becomes a VI method plus an expensive search procedure. This lack of possible monotone update can be due to several factors: the nonlinear dynamics, the existence of trajectories exiting the computational domain, and a sensitivity to the artificial boundary condition. We report having performed similar tests for the linear double integrator problem ($\ddot{x} = u$) with similar results, therefore we conjecture that in this case, the underperformance of the AMVI scheme is due to poor boundary resolution and its use by optimal trajectories. Unfortunately, this is a recurrent problem in the context of optimal control. This situation does not constitute a problem for the API algorithm, where a substantial speedup is seen in both coarse and fine meshes. Note that compared to PI, the accelerated scheme has a number of iterations on its second part which is independent of the mesh parameters as we are in a close neighborhood of the optimal solution.

# nodes	Δx	VI	PI	AMVI	VI($2\Delta x$)	PI(Δx)	API
81^2	5E-2	39.6 (529)	5.35 (8)	1.42E2 (3)	1.86 (207)	1.47 (4)	3.33 (211)
161^2	2.5E-2	3.22E2 (1267)	34.5 (11)	1.01E3 (563)	10.7(165)	6.87 (4)	17.5 (169)
321^2	1.25E-2	3.36E4 (2892)	3.36E2 (14)	1.55E4 (2247)	88.9 (451)	47.7 (4)	1.36E2 (455)

Table 3.5.: Test 2 (Van der Pol oscillator): CPU time (iterations) for different algorithms.

3.4.3. Test 3: Dubin's Car

Having tested some basic features of the proposed schemes, we proceed with our numerical study of the API method by considering a three-dimensional nonlinear dynamical system given by

$$f(x, y, z, u) = \begin{pmatrix} \cos(z) \\ \sin(z) \\ u \end{pmatrix},$$

corresponding to a simplified version of the so-called Dubin's car, a model extensively used in the context of reachable sets and differential games. System parameters are set:

$$\Omega =]-2\,,2[^2\,, \quad U = [-1\,,1]\,, \quad \lambda = 1\,, \quad \Delta t = 0.2\Delta x\,, \quad L(x, y, z, u) = x^2 + y^2\,,$$

and the control space is discretized into 11 equidistant points; the boundary value is set to $v(x) = 3$ in $\partial\Omega$ and reference solution is taken with $\Delta x = 0.0125$. Different isosurfaces for this optimal control problem can be seen in Figure 3.6, and CPU times for different meshes are shown in Table 3.6. This case is an example in which the mesh ratio between coarse and fine meshes is well-balanced, and the time spent in pre-processing via VI is not relevant in the overall API CPU time, despite leading to a considerable speedup of a factor 8 with an order of 10^6 grid points. In the last line of Table 3.6, the VI

algorithm was stopped after 4 hours of simulation without achieving convergence, which is illustrative of the fact that acceleration techniques in such problems are not only desirable but necessary in order to obtain results with acceptable levels of accuracy.

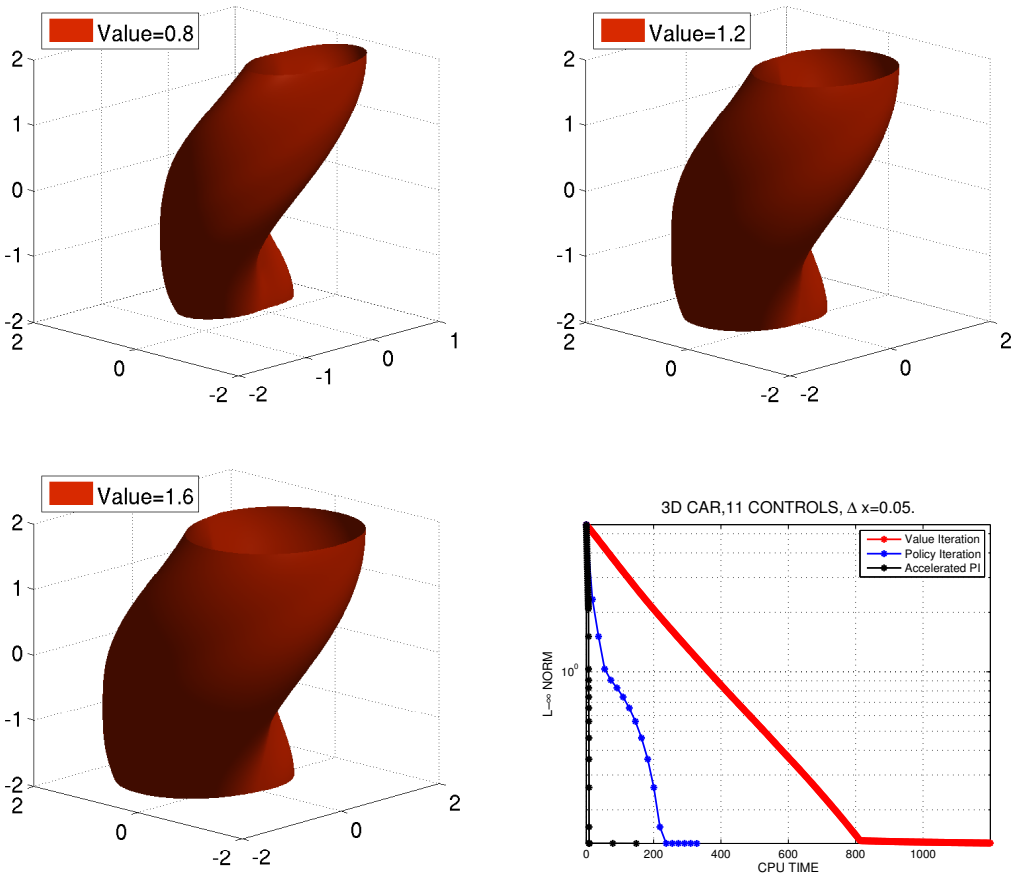

Figure 3.6.: Test 3: Dubin's car value function isosurfaces and error evolution (bottom right).

Minimum time problems

3.4.4. Tests 4 and 5: Minimum time problems in 2D

The next two cases are based on a two-dimensional Eikonal equation. For both problems, common seetings are given by

$$f(x, y, u) = \begin{pmatrix} \cos(u) \\ \sin(u) \end{pmatrix}, \quad U = [-\pi, \pi], \quad \Delta t = 0.8\Delta x\,.$$

What differentiates the problems is the domain and target definitions; Test 4 considers a domain $\Omega =]-1, 1[^2$ and a target $\mathcal{T} = (0, 0)$, while for Test 5, $\Omega =]-2, 2[^2$ and

# nodes	Δx	VI	PI	VI($2\Delta x$)	PI(Δx)	API
41^3	0.1	50.6 (192)	12.2 (12)	0.84 (8)	8.52 (3)	9.36 (11)
81^3	5E-2	1.19E3 (471)	3.28E2 (18)	8.98 (39)	1.39E2 (9)	1.48E2 (48)
161^3	2.5E-2	$\geq$ 1.44E4	9.93E3 (12)	3.02E2 (30)	2.92E3 (10)	2.62E3 (40)

Table 3.6.: Test 3 (Dubin's car): CPU time (iterations) for different algorithms

$\mathcal{T} = \{x \in \mathbf{R}^2 \ : \ ||x||_2 \leq 1\}$. Reference solutions are considered to be the distance function to the respective targets, which is an accurate approximation provided that the number of possible control directions is large enough. Contour plots of the approximated optimal value functions for both problems are shown in Figure 3.7. For Test 4, with a discretization of the control space into set of 64 equidistant points, CPU time results are presented in Table 3.7; it can be seen that API provides a speedup of a factor 8 with respect to VI over fine meshes despite the large set of discrete control points. For sake of completeness, we include Figure 3.8, which illustrates, for both problems, the way in which the API idea acts: pre-processing of the initial guess of PI leads to proximity to a "quadratic convergence neighborhood"; fast error decay that coarse mesh VI has in comparison with the fine mesh VI is clearly noticeable. For Test 4, Table 3.7 shows experimental convergence rates achieved by the fully discrete scheme, in both L^1 and L^∞ norms, which are in accordance with theoretically expected rate of 1/2 (see [22]). Test 5 features an enlarged target, and differences in terms of CPU times are presented in Table 3.7 where, for a discrete set of 72 equidistant controls, the speedup is reduced to a factor 4. In general, from a mesh node, larger or more complicated targets represent a difficulty in terms of the choice of the minimizing control, which translates into a larger number of iterations. In this case, the CPU time spent in the pre-processing is significant to the overall CPU time, but increasing this ratio in order to reduce its share will lead to an underperfomant PI part of the algorithm.

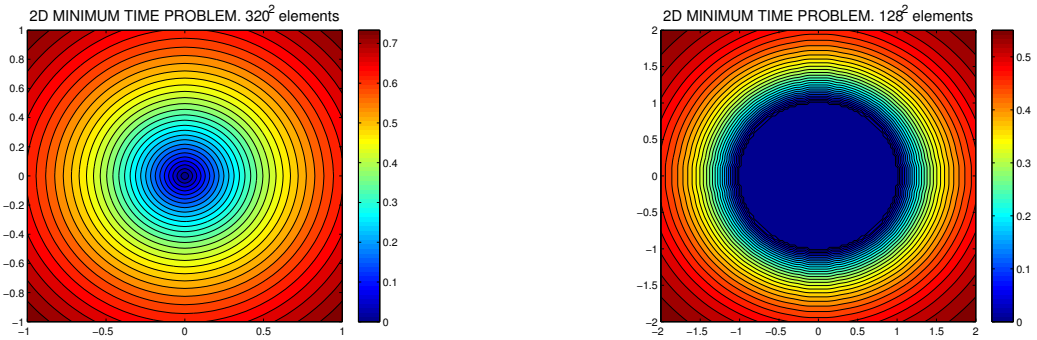

Figure 3.7.: Value function contours for 2D Eikonal equations: Test 4 (left) and Test 5 (right).

# nodes	Δx	VI	PI	VI($2\Delta x$)	PI(Δx)	API
41^2	5E-2	3.16 (37)	1.89 (12)	0.39 (5)	0.38 (2)	0.77 (7)
81^2	2.5E-2	8.23 (69)	4.43 (19)	0.80 (12)	0.53 (2)	1.33 (14)
161^2	1.25E-2	39.2 (133)	12.6 (13)	2.55 (31)	2.11 (3)	4.66 (34)

Table 3.7.: Test 4 (2D Eikonal): CPU time (iterations) for different algorithms.

# nodes	Δx	$L^1 - error$	rate	$L^\infty - error$	rate
41^2	5E-2	2.1E-2	0.60	8.9E-3	0.61
81^2	2.5E-2	1.4E-2	0.64	5.8E-3	0.64
161^2	1.25E-2	8.5E-3	0.68	3.7E-3	0.75
321^2	6.25E-3	5.3E-3		2.2E-3	

Table 3.7.: Test 4 (2D Eikonal): Rate of convergence for the API scheme with 64 controls.

3.4.5. Tests 6 and 7: Minimum time problems in 3D

We develop a three-dimensional extension of the previously presented examples. System dynamics and common parameters are given by

$$f(x, y, z, (u_1, u_2)) = \begin{pmatrix} \sin(u_1)\cos(u_2) \\ \sin(u_1)\sin(u_2) \\ \cos(u_1) \end{pmatrix}, \quad U = [-\pi,\, \pi] \times [0\,, \pi]\,, \quad \Delta t = 0.8\Delta x\,.$$

As in the two-dimensional study, we perform different tests by changing the domain and the target. For Test 6 we set $\Omega =]-1,\, 1[^3$ and $\mathcal{T} = (0, 0, 0)$, while for Test 7, $\Omega =]-6,\, 6[$ and $\mathcal{T}$ is the union of two unit spheres centered at $(-1, 0, 0)$ and $(1, 0, 0)$. In both cases, the set of controls is discretized into 16×8 points. Reachable sets for Test 7 are shown in Figure 3.9, and CPU times for both tests can be found in Tables 3.8 and 3.9. We observe similar results as in the 2D tests, with up to $10\times$ acceleration for a simple target, and $4\times$ with more complicated targets. Note that in the second case, the speedup is similar to the natural performance that would be achieved by a PI algorithm. This is due to a weaker influence of the coarse VI iteration, which is sensitive to poor resolution of a complex target.

In Table 3.8 and Table 3.9, θ and φ represent the number of angles we consider in the discretization of the control space (the unit ball in $\mathbb{R}^3$ with spherical coordinates).

# nodes	Δx	VI	PI	VI($2\Delta x$)	PI(Δx)	API
64^2	6.35E-2	4.02 (36)	1.42 (9)	0.84 (10)	0.53 (4)	1.37 (14)
128^2	3.15E-2	16.9 (70)	6.25 (14)	2.80 (25)	1.66 (2)	4.46 (27)
256^2	1.57E-2	1.09E2 (135)	38.7 (16)	15.8 (62)	11.7 (8)	27.5 (70)
512^2	7.8E-3	9.80E2 (262)	3.98E2(168)	1.07E2 (126)	1.09E2 (12)	2.16E2 (138)

Table 3.7.: Test 5 (2D Eikonal): CPU time (iterations) for different algorithms with 72 controls.

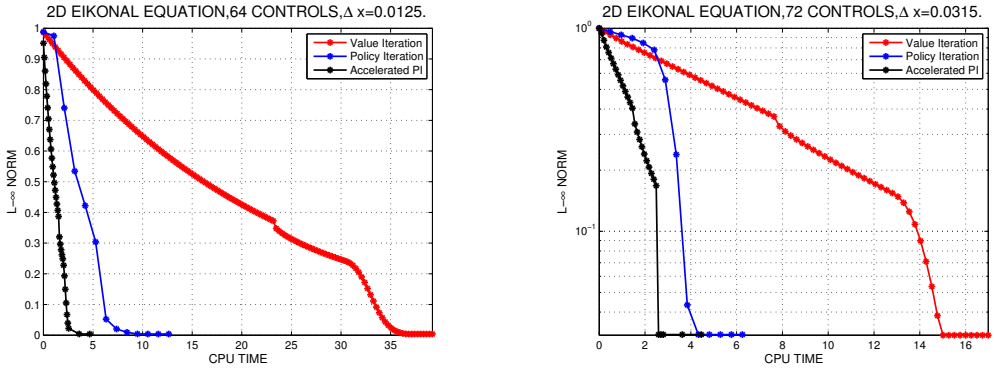

Figure 3.8.: Error evolution in 2D Eikonal equations: Test 4 (left) and Test 5:(right).

3.4.6. Test 8: Minimum time problem in 4D

We conclude our series of tests in minimum time problems by considering a four-dimensional problem with a relatively reduced control space. In the previous examples we have studied the performance of our scheme in cases where the set of discrete controls was fairly large, while in several applications, it is also often the case that the set of admissible discrete controls is limited and attention is directed towards the dimensionality of the state space. The following problem tries to mimic such setting. System dynamics

# nodes	Δx	VI	PI	VI($2\Delta x$)	PI(Δx)	API
41^3	0.05	4.83E2 (44)	1.22E3 (10)	4.61 (5)	1.19E2 (3)	1.23E2 (8)
81^3	0.025	7.67E3 (84)	1.47E3 (13)	24.3 (12)	3.88E2 (3)	4.12E2 (15)

Table 3.8.: Test 6 (3D Eikonal): CPU time (iterations) for different algorithms with $\theta = 16, \varphi = 8$ controls.

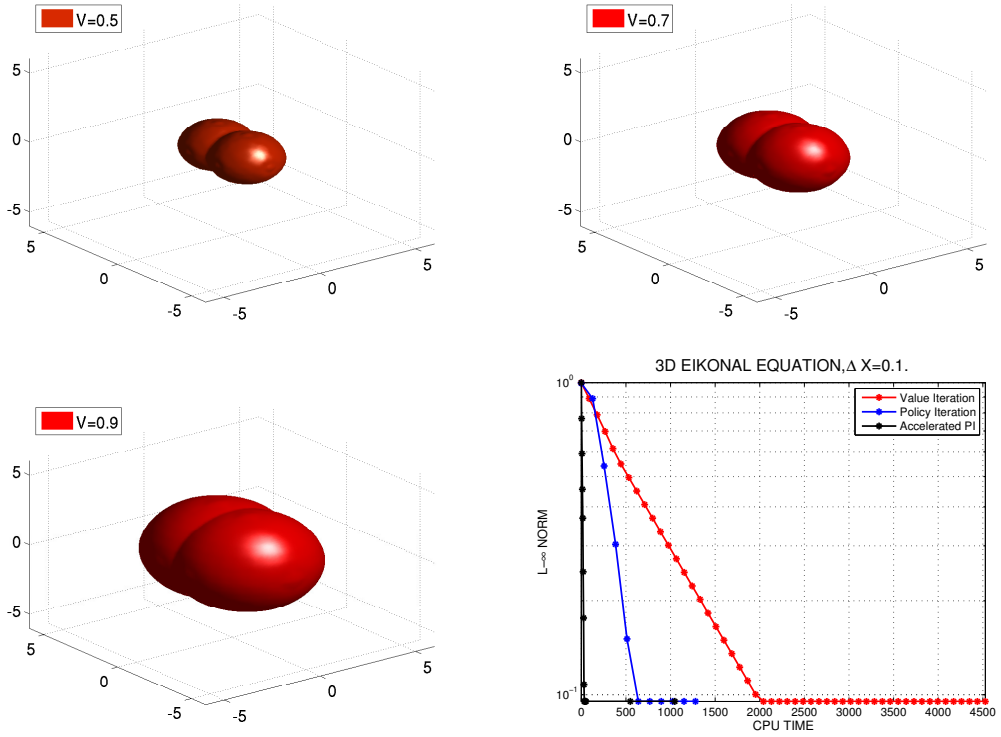

Figure 3.9.: Test 7 (3D Eikonal): different value function isosurfaces and error evolution (bottom right).

are given by

$$f(x, y, z, w, (u_1, u_2, u_3, u_4)) = \begin{pmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \end{pmatrix},$$

the domain is $\Omega =]-1, 1[^4$, the target is $\mathcal{T} = \partial\Omega$, $\Delta t = 0.8\Delta x$ and U is the set of 8 directions pointing to the facets of the four-dimensional hypercube. Figure 3.10 shows different reachable sets and CPU times are presented in Table 3.10. In the finest mesh a speedup of $8\times$ is observed, which is consistent with the previous results on simple targets. Thus, the performance of the presented algorithm is not sensitive neither to the number of discrete controls nor to the dimension of the state space, whereas it is affected by the complexity of the target.

3.4.7. Application to optimal control problems of PDEs

Having developed a comprehensive set of numerical tests concerning the solution of optimal control problems via static HJB equations, which assessed the performance of the proposed API algorithm, we present an application where the existence of acceler-

# nodes	Δx	VI	PI	VI($2\Delta x$)	PI(Δx)	API
61^3	0.2	2.67E2 (25)	1.22E2 (9)	1.44 (11)	68.0 (3)	69.1 (14)
121^3	0.1	4.52E3 (52)	1.28E3 (11)	25.15E1 (12)	9.96E2 (3)	1.01E3 (15)

Table 3.9.: Test 7 (3D Eikonal): CPU time (iterations) for different algorithms with $\theta = 16, \varphi = 8$.

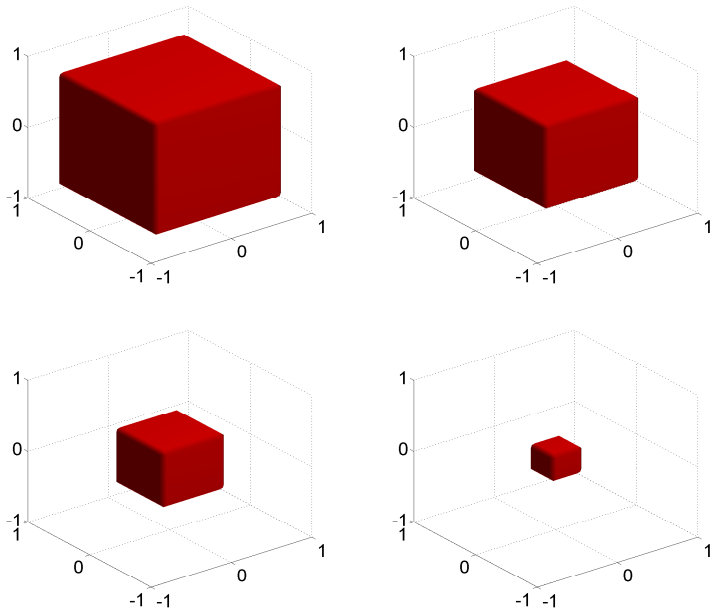

Figure 3.10.: Test 8 (4D minimum time): different value function isosurfaces with $x_4 = 0$.

ated solution techniques for high-dimensional problems is particularly relevant, namely, the optimal control of systems governed by partial differential equations. From an abstract perspective, optimal control problems where the dynamics are given by evolutive partial differential equations correspond to systems where the state lies in an infinite-dimensional Hilbert space (see [97]). Nevertheless, in terms of practical applications, different discretization arguments can be used to deal with this fact, and (sub)optimal control synthesis can be achieved through finite-dimensional, large-scale approximations of the system. At this step, the resulting large-scale version will scale according to a finite element mesh parameter, and excepting for the linear-quadratic case and some closely related versions, it would be still computationally unfeasible for modern architectures (for instance, for a 100 elements discretization of a 1D PDE, the resulting optimal control would be characterized as the solution of a HJB equation in $\mathbb{R}^{100}$). Therefore, a standard remedy in optimal control and estimation is the application of model order reduction techniques, which, upon a large-scale version of the system, recover its most

# nodes	Δx	VI	PI	VI($2\Delta x$)	PI(Δx)	API
21^4	0.1	13.6 (15)	16.2 (11)	0.30 (4)	2.79 (2)	3.09
41^4	5E-2	4.79E2 (29)	6.30E2 (21)	10.2 (12)	48.3 (2)	58.5

Table 3.10.: Test 8 (4D minimum time): CPU time (iterations) for different algorithms.

relevant dynamical features in a low-order approximation of prescribed size such as Proper Orthogonal Decomposition. In this context, surprisingly good control synthesis can be achieved with a reduced number of states (for complex nonlinear dynamics and control configurations an increased number of reduced states may be required). Previous attempts in this direction dates back to [67, 68] and more recently to [1, 2]. We present an example where we embedded our accelerated algorithm inside the described framework. We address the reader to Chapter 4 for major details on the application of the POD method to the HJB equation. Even if we focus on evolutive HJBs equation it will be straightforward to see how it could work for infinite horizon problems. Note that, in this example, model reduction method is only applied in order to make the problem feasible for the Dynamic Programming approach. The acceleration is due to the proposed API scheme. Let us consider a minimum time problem for the linear heat equation:

$$\begin{cases} y_t(x,t) = cy_{xx}(x,t) + y_0(x)u(t)\,, \\ y(0,t) = y(1,t) = 0\,, \\ y(x,0) = y_0(x)\,, \end{cases} \tag{3.21}$$

where $x \in [0,1]$, $t \in [0,T]$, $c = 1/80$ and $u(t) : [0,T] \to \{-1,0,1\}$. After performing a finite difference discretization of the uncontrolled system, we perform a Galerkin projection with basis function computed with a Proper Orthogonal Decomposition (POD) method, leading to a reduced order model (we refer to Chapter 2 and the references inside for an introduction to this topic). In general, model reduction techniques do have either a priori or a posteriori error estimates which allow to prescribe a certain number of reduced states yielding a desired level of accuracy. For this simple case, we consider the first 3 reduced states, which for a one-dimensional heat transfer process with one external source provides a reasonable description of the input-output behavior of the system. The system is reduced to:

$$\frac{d}{dt}\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} -0.123 & -0.008 & -0.001 \\ -0.008 & -1.148 & -0.321 \\ -0.001 & -0.321 & -3.671 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} + \begin{bmatrix} -5.770 \\ -0.174 \\ -0.022 \end{bmatrix} u(t). \tag{3.22}$$

Once the reduced model has been obtained, we solve the minimum time problem with target $\mathcal{T} = (0,0,0)$. Figure 3.11 shows contour plots of the value function in the reduced space, while in Figure 3.12 a comparison of the performance of the minimum time controller with respect to the uncontrolled solution and to a classical linear-quadratic

controller is presented. CPU times are included in Table 3.12; a speedup of 4× can be observed, the acceleration would become more relevant as soon as more refined meshes and complex control configurations are considered.

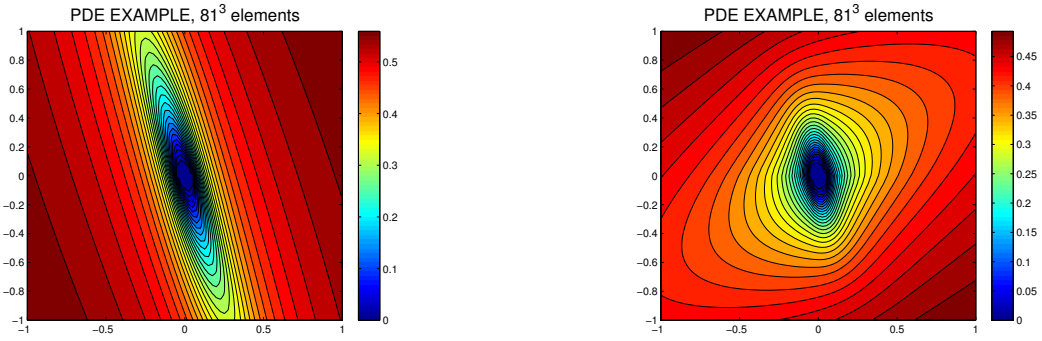

Figure 3.11.: Minimum time control of the heat equation: contour plots of different cuts across the reduced value function. Left: $x_3 = 0$. Right: $x_2 = 0$.

# nodes	Δx	VI	PI	VI($2\Delta x$)	PI(Δx)	API
21^3	0.1	1.87 (76)	0.91 (11)	0.32 (27)	0.59 (8)	0.98
41^3	5E-2	27.8 (178)	12.4 (15)	1.65 (76)	6.34 (10)	7.99
81^3	2.5E-2	6.13E2 (394)	2.68E2 (15)	27.7 (178)	1.45E2 (9)	1.72E2

Table 3.12.: Minimum time control of the heat equation: CPU time (iterations) for different algorithms.

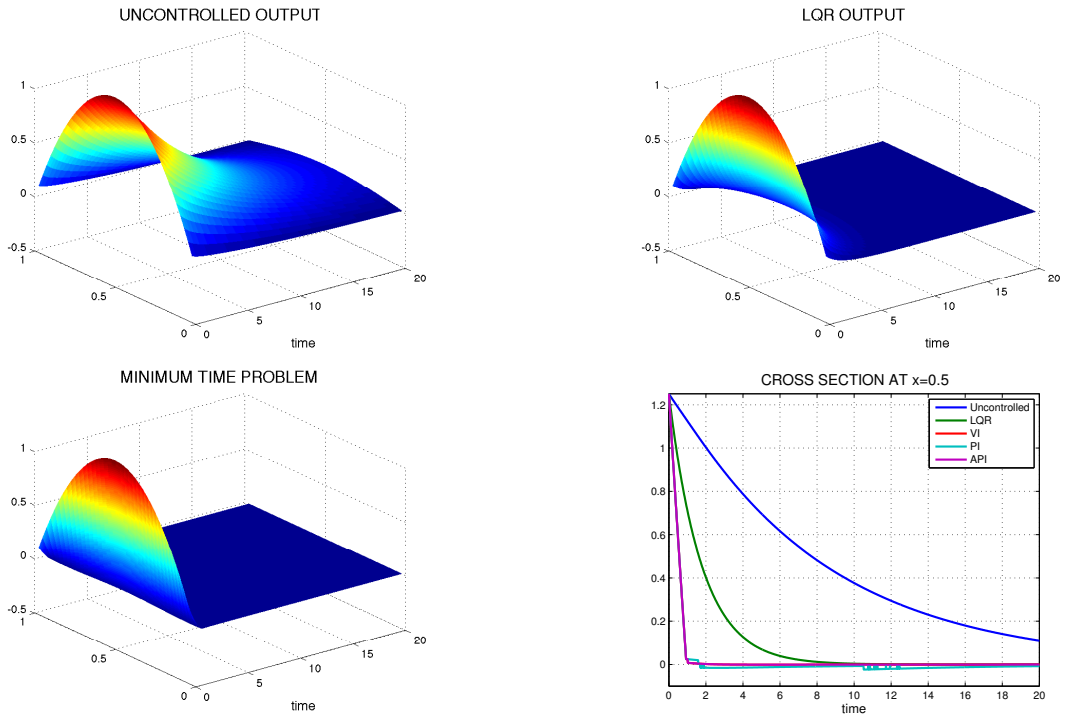

Figure 3.12.: Optimal control of the heat equation. Top left: uncontrolled output of the system. Top right: classical controlled output via linear-quadratic control. Bottom left: controlled output via proposed procedure of model reduction + minimum time HJB controller. Bottom right: cross section of the different outputs.

4. POD approximation for controlled dynamical systems and Dynamic Programming

In this chapter we present an algorithm for the approximation of a finite horizon optimal control problem for evolutive linear and semi–linear PDEs. The method is based on the coupling between an adaptive POD representation of the solution and a Dynamic Programming approximation scheme for the corresponding evolutive Hamilton-Jacobi equation characterizing the value function of the control problem for the reduced system. We discuss several features regarding the adaptivity of the method, the role of error estimate indicators to choose a time subdivision of the problem and the computation of the basis functions. Some test problems are presented to illustrate the method. This topic has already presented in [1, 2].

4.1. An optimal control problem

We will present this approach for the finite horizon control problem. Consider the controlled system

$$\begin{cases} \dot{y}(s) = f(y(s), u(s), s), \quad s \in (t, T] \\ y(t) = x \in \mathbb{R}^n, \end{cases} \tag{4.1}$$

we will denote by $y : [t, T] \to \mathbb{R}^n$ the solution, by u the control $u : [t, T] \to \mathbb{R}^m$, $f : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$, $s \in (t, T]$ and by

$$\mathcal{U} = \{u : [t, T] \to U\}$$

the set of admissible controls where $U \subset \mathbb{R}^m$ is a compact set. Whenever we want to emphasize the depence of the solution from the control u we will write $y(t; u)$. Assume that there exists a unique solution trajectory for (4.1), provided the controls are measurable (a precise statement can be found in [15]). The system (4.1) can be also interpreted as a semidiscrete problem with finite difference schemes. For the finite horizon optimal control problem the cost functional will be given by

$$\min_{u \in \mathcal{U}} J_{x,t}(u) := \int_t^T L(y(s, u), u(s), s) e^{-\lambda s} \, ds + g(y(T)) \tag{4.2}$$

where $L : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}$ is the running cost and $\lambda \geq 0$ is the discount factor.
The goal is to find a state-feedback control law $u(t) = \mu(y(t), t)$, in terms of the state

equation $y(t)$, where μ is the feedback map. To derive optimality conditions we use the well-known *dynamic programming principle* due to Bellman (see [15]). We first define the value function:

$$v(x,t) := \inf_{u \in \mathcal{U}} J_{x,t}(u). \tag{4.3}$$

Note that in finite horizon problems, the value function $v(x,t)$ is even a function of time.

Proposition 4.1 (DPP) *For all $x \in \mathbb{R}^n$ and $0 \leq \tau \leq t$ then:*

$$v(x,t) = \min_{u \in \mathcal{U}} \left\{ \int_t^\tau L(y(s), u(s), s) e^{-\lambda s} \, ds + v(y, t-\tau) \right\}. \tag{4.4}$$

Due to (4.4) we can derive the *Hamilton-Jacobi-Bellman* equations (HJB):

$$-\frac{\partial v}{\partial t}(y,t) - \lambda v(y,t) = \min_{u \in U} \{ L(y,u,t) + \nabla v(y,t) \cdot f(y,u,t) \}. \tag{4.5}$$

This is an evolutive nonlinear partial differential equation of the first order which is hard to solve analitically although a general theory of weak solutions is available [15]. Rather we can numerically solve it by means of finite difference or semi-Lagrangian schemes (see the book [40] for a comprehensive analysis of approximation schemes for Hamilton-Jacobi equations). For a semi-Lagrangian discretization one starts by a discrete version of HJB by discretizing the underlined control problem and then project the semi-discrete scheme on a grid obtaining the fully discrete scheme;

$$\begin{cases} v_i^{n+1} = \min_{u \in U} [\Delta t \, L(x_i, u, n\Delta t) + \mathcal{I}[v^n](x_i + \Delta t \, f(x_i, u, t_n))] \\ \\ v_i^0 = g(x_i), \end{cases}$$

with $x_i = i\Delta x$, $t_n = n\Delta t$, $v_i^n := v(x_i, t_n)$ and $\mathcal{I}[\cdot]$ is an interpolation operator which is necessary to compute the value of v^n at the point $x_i + \Delta t \, f(x_i, u, t_n)$ (in general, this point will not be a node of the grid). The interested reader will find in [41] a detailed presentation of the scheme and a priori error estimates for its numerical approximation. Note that, we also need to compute the minimum in order to get the value v_i^{n+1}. Since v^n is not a smooth function, we compute the minimum by means of a minimization method which does not use derivatives (this can be done by the Brent's algorithm as in [27]).
As we have already told the HJB allows to compute the optimal feedback via the value function, but there are two major difficulties: the solution of an HJB equation are in general non-smooth and the approximation in high dimension is not feasible.
As briefly introduced in Section 3.4.7 we will explain how to manage the curse of dimensionality by means of the POD method. Let us focus on the following infinte-dimensional

abstract problem:

$$\begin{cases} \dfrac{d}{ds}\langle y(s),\varphi\rangle_H + a(y(s),\varphi) = \langle B(u(s)),\varphi\rangle_{V',V} \quad \forall\varphi\in V \\ \\ y(t) = y_0 \quad \in H, \end{cases} \tag{4.6}$$

where $B : U \to V'$ is a linear and continuous operator. We assume that a space of admissible controls $\mathcal{U}_{ad}$ is given in such a way that for each $u \in \mathcal{U}_{ad}$ and $y_0 \in H$ there exists a unique solution y of (4.6). V and H are two Hilbert spaces, with $\langle\cdot,\cdot\rangle_H$ we denote the scalar product in H; $a : V \times V \to \mathbb{R}$ is symmetric coercive and bilinear. Then, we introduce the cost functional of the finite horizon problem

$$\mathcal{J}_{y_0,t}(u) := \int_t^T L(y(s),u(s),s)e^{-\lambda s}\,ds + g(y(T)),$$

where $L : V \times U \times [t,T] \to \mathbb{R}$. The optimal control problem is

$$\begin{aligned} &\min_{u\in\mathcal{U}_{ad}} \mathcal{J}_{y_0,t}(u) \\ \text{subject to the constraint:} \quad &y \in W(t,T)\times\mathcal{U} \text{ solves (4.6)}, \end{aligned} \tag{4.7}$$

where $W(t,T)$ is the standard Sobolev space:

$$W(t,T) = \{\varphi \in L^2(t,T;V), \varphi_t \in L^2(t,T;V')\}.$$

The model reduction approach for an optimal control problem (4.7) is based on the Galerkin approximation of the dynamics with some information on the controlled dynamics (snapshots). To compute a POD solution for (4.7) we make the following ansatz

$$y^\ell(\cdot,s) = \sum_{i=1}^{\ell} w_i(s)\psi_i(\cdot). \tag{4.8}$$

where $\{\psi\}_{i=1}^{\ell}$ is the POD basis functions computed as explained in Chapter 2.
Let us suppose we have an initial guess for the control u, the computation of the POD basis functions can be summarized by the following three steps:

1. compute the solution y at given times s_j, the $y(s_j;u)$ are called *snapshots*;
2. collect the snapshots into a matrix Y and compute the singular value decomposition of $Y = \Psi\Sigma V^T$;
3. the first ℓ columns of Ψ will be the POD basis of rank ℓ.

As explained in Section 2.1.2 a good indicator of the accuracy of our POD approximation is given by (2.7). We introduce mass and stiffness matrix:

$$M = (m_{ij}) \in \mathbb{R}^{\ell\times\ell} \text{ with } m_{ij} = \langle\psi_j,\psi_i\rangle_H,$$

$$S = (s_{ij}) \in \mathbb{R}^{\ell\times\ell} \text{ with } s_{ij} = a(\psi_j, \psi_i),$$

and the control map $b : U \to \mathbb{R}^\ell$ is defined by:

$$u \to b(u) = (b(u)_i) \in \mathbb{R}^\ell \text{ with } b(u)_i = \langle Bu, \psi_i\rangle_H.$$

The coefficients of the initial condition $y_0^\ell \in \mathbb{R}^\ell$ are determined by $(w_0)_i = \langle y_0, \psi_i\rangle_X$, $1 \leq i \leq \ell$, and the solution of the reduced dynamic problem is denoted by $w^\ell(s) \in \mathbb{R}^\ell$. Then, the Galerkin approximation is given by

$$\min J^\ell_{w_0^\ell,t}(u) \tag{4.9}$$

with $u \in \mathcal{U}_{ad}$ and w solves the following equation:

$$\begin{cases} \dot{w}^\ell(s) = F(w^\ell(s), u(s), s) \quad s > t, \\ \\ w^\ell(t) = w_0^\ell. \end{cases} \tag{4.10}$$

The cost functional is defined:

$$J^\ell_{w_0^\ell,t}(u) = \int_t^T L(w^\ell(s), u(s), s)e^{-\lambda s}\, ds + g(w^\ell(T)),$$

with w^ℓ and y^ℓ linked though (4.8) and the nonlinear map $F : \mathbb{R}^\ell \times U \to \mathbb{R}^\ell$ is given by:

$$F(w^\ell(s), u(s), s) = M^{-1}(-Sw^\ell(s) + b(u(s))).$$

The value function v^ℓ, defined for the state $w_0 \in \mathbb{R}^\ell$, is given by

$$v^\ell(w_0^\ell, t) = \inf_{u\in\mathcal{U}_{ad}} J^\ell_{w_0^\ell,t}(u),$$

and w^ℓ solves (4.10) with the control u and initial condition w_0.

We give an idea how we have computed the intervals for reduced HJB. HJBs are defined in $\mathbb{R}^n$, but we have restricted our numerically domain Υ_h which is a bounded subset of $\mathbb{R}^n$. This is justified since $y + \Delta t F(y, u) \in \Upsilon_h$ for each $y \in \Upsilon_h$ and $u \in \mathcal{U}_{ad}$. We can chose $\Upsilon_h = [a_1, b_1] \times [a_2, b_2] \times \ldots [a_\ell, b_\ell]$ with $a_1 \geq a_2 \geq \ldots \geq a_\ell$. How should we compute these intervals $[a_i, b_i]$?

Ideally the intervals should be chosen so that the dynamics contains all the components of the controlled trajectory. Moreover, they should be encapsulated because we expect that their importance should decrease monotonically with their index and that our interval lengths decrease quickly.

Let us suppose to discretize the space control with a finite number of elements $U = \{u_1, \ldots, u_M\}$ where U is symmetric, to be more precise if $\bar{u} \in U \Rightarrow -\bar{u} \in U$.

Hence, if $y^\ell(s) = \sum_{i=1}^{\ell} \langle y(s), \psi_i\rangle\psi_i = \sum_{i=1}^{\ell} w_i(s)\psi_i$, as a consequence, the coefficients $w_i(s) \in [a_i, b_i]$. We consider the trajectories solution $y(s, u_j)$ such that the control is

constant $u(s) \equiv u_j$ for each t_j, $j = 1, \dots, M$. Then, we have

$$y^\ell(s, u_j) = \sum_{i=1}^{\ell} \langle y(s, u_j), \psi_i \rangle \psi_i.$$

We write $y^\ell(s, u_j)$ to stress the dependence on the constant control u_j. Each trajectory $y^\ell(s, u_j)$ has some coefficients $w_i^{(j)}(t)$ for $i = 1, \dots, \ell$, $j = 1, \dots, M$. The coefficients $w_i^{(j)}(s) := \langle y(s, u_j), \psi_i \rangle$ will belong to intervals of the type $[\underline{\mathrm{w}}_i^{(j)}, \overline{\mathrm{w}}_i^{(j)}]$ where we chose a_i, b_i such that:

$$a_i \equiv \min\{\underline{\mathrm{w}}_i^{(1)}, \dots, \underline{\mathrm{w}}_i^{(M)}\}, \quad i = 1, \dots, \ell,$$

$$b_i \equiv \max\{\overline{\mathrm{w}}_i^{(1)}, \dots, \overline{\mathrm{w}}_i^{(M)}\}, \quad i = 1, \dots, \ell.$$

Then, we have a method to compute the intervals and we turn our attention to the numerical solution of an optimal control problem for evolutive equation, as we will see in the following section.

4.2. An error estimate for the coupled HJB and POD

This section is devoted to estimate the error of the approximation of the value function which is computed coupling HJB and POD methods. The result is valid for an infinite horizon problem. Although we have presented the method for a finite horizon problem, one can think to obtain the same strategies when the time t is going to infinity getting a steady HJB equation as (3.4).
Let us suppose the assumptions of Theorem 3.1 hold. Then, our goal is to estimate $\|V_{\Delta t}^\ell - V\|$, where $V_{\Delta t}^\ell$ is obtained from the reduced model. Applying the triangular inequality, we obtain:

$$\|V_{\Delta t}^\ell - V\|_\infty \leq \|V - V_{\Delta t}\|_\infty + \|V_{\Delta t} - V_{\Delta t}^\ell\|_\infty, \tag{4.11}$$

where the term $\|V - V_{\Delta t}\|_\infty$ is known from (3.17). Let us recall that $V_{\Delta t}$ solves (3.5) whereas:

$$V_{\Delta t}^\ell(x^\ell) = \min_{u \in U} \left\{ e^{-\lambda \Delta t} V_{\Delta t}^\ell(x^\ell + \Delta t f^\ell(x^\ell, u)) + \Delta t L^\ell(x^\ell, u) \right\}$$

for vertices $x^\ell \in \mathbb{R}^\ell$ such that $\Psi x \in \overline{\Omega}$, $f^\ell(x^\ell, u) := \Psi^T f(\Psi y^\ell, u)$ and $L^\ell(x^\ell, u) := \Psi^T L(\Psi y^\ell, u)$.

Thus, for a given optimal control u^*, we get the following inequalities from standard

triangular inequality:

$$\begin{aligned}\|V_{\Delta t}(x) - V^{\ell}_{\Delta t}(\Psi^T x)\|_\infty &= \left\| e^{-\lambda\Delta t}\left(V_{\Delta t}(x+\Delta t f(x,u^*)) - V^{\ell}_{\Delta t}(x+\Delta t f(\Psi\Psi^T x,u^*))\right) + \right. \\ &+ \left. \Delta t\left(L(x,u^*) - L(\Psi\Psi^T x,u^*)\right)\right\| \\ &\leq e^{-\lambda\Delta t}\left\|\left(V_{\Delta t}(x+\Delta t f(x,u^*)) - V^{\ell}_{\Delta t}(x+\Delta t f(\Psi\Psi^T x,u^*))\right)\right\| + \\ &+ \Delta t\left\|L(x,u^*) - L(\Psi\Psi^T x,u^*)\right\|. \end{aligned} \tag{4.12}$$

The maximum has to be thought with respect to $x \in \mathbb{R}^n$. As already explained the computation of $V_{\Delta t}\left(x+\Delta t f(x,u^*)\right)$ needs an interpolation operator since $x+\Delta t f(x,u^*)$ may not be a point of the grid.
Let us we briefly recall the stability property of Polynomial Interpolation. Consider a set of function values $\left\{\widetilde{f}(x_i)\right\}$ which is a perturbation of the data $f(x_i)$ relative to the nodes x_i, with $i = 0,\ldots,n$ in an interval $[a,b]$.
Denoting by $\mathcal{I}^n[\widetilde{f}]$ the interpolating polynomial on the set of values $\widetilde{f}(x_i)$ we have:

$$\left\|\mathcal{I}^n[f(x)] - \mathcal{I}^n[\widetilde{f}(x)]\right\|_\infty \leq \Lambda_n(x)\left\|f(x) - \widetilde{f}(x)\right\|_\infty. \tag{4.13}$$

where $\Lambda_n(x)$ denotes the Lebesgue's constant. From (4.12), applying (4.13) and the triangular inequality we have:

$$\begin{aligned}&\left\|V_{\Delta t}\left(x+\Delta t f(x,u^*)\right) - V^{\ell}_{\Delta t}\left(x+\Delta t f(\Psi\Psi^T x,u^*)\right)\right\| \leq \\ &\qquad \Lambda_n\left(\|x-\Psi\Psi^T x\| + \Delta t\|f(x,u^*) - f(\Psi\Psi^T x,u^*)\|\right),\end{aligned}$$

then for the function L we have:

$$\|L(x,u^*) - L(\Psi\Psi^T x,u^*)\| \leq K_L\|x-\Psi\Psi^T x\|.$$

Furthemore if we assume that Theorem A.2 hold with:

$$\|x-\Psi\Psi^T x\| \leq \delta^{\ell} \qquad \|f(x,u) - f(\Psi\Psi^T x,u)\| \leq \varepsilon^{\ell},$$

we have proved the following theorem:

Theorem 4.1 *Assume the assumptions of Theorem 3.1 in Chapter 3 and Theorem A.2 holds. Then*

$$\|V^{\ell}_{\Delta t} - V\|_\infty \leq C(\Delta t)^{1/2} + \frac{K_f}{\lambda(\lambda-K_f)}\frac{\Delta x}{\Delta t} + \delta^{\ell}(e^{-\lambda\Delta t}\Lambda_n + \Delta t K_L) + \Delta t\Lambda_n\varepsilon^{\ell}$$

.

In Appendix A.2 we present a Theorem on the perturbation of ordinary differential equations. Let us make a couple of remarks:

Remark 4.1 *1. Theorem A.2 is rather useful in Model Order Reduction, since one*

may think $y = \Psi\Psi^T x$ *and* g *as the perturbed dynamics coming from the POD, e.g.* $g(t, y) = \ominus^T f(t, \Psi y)$, *where* $\Psi\Psi^T$ *is the projection operator.*

2. *One can always take* $\varepsilon^\ell, \delta^\ell$ *small according to the computation of the snapthots. This is something easy to verify once the POD basis is computed. Moreover, it is guarenteed when* $\ell \to +\infty$ *that* $f \equiv g$. *This helps us because if we increase the number of basis functions* ℓ, *we will get the desided tolerance* $(\varepsilon^\ell, \delta^\ell)$.

4.3. Adapting POD approximation

In this section we present an adaptive method to compute POD basis functions. As we have explained, our final goal is to obtain the optimal feedback law by means of HJB equations, so we have a big constraint on the number of variables.
We will see that, for a parabolic equation, one can try to solve the problem with only three/four POD basis functions; they are sufficient to describe the solution in a rather accurate way. In fact the singular values decay rapidly and it is easier to work with a really low dimensional problem (see Section 4.4.1 and 4.4.2).
On the contrary, hyperbolic equations do not have this property for their singular values and they will require a rather large set of POD basis functions to get accurate results, since there is a high variability during the evolution. Note that we can not follow the approach suggested in [81] because we can not add more basis functions when it turns to be necessary due to the constraint already mentioned. Then, it is quite natural to split the problem into subproblems having different POD basis functions. The crucial point is to decide the splitting in order to have the same number of basis functions in each subdomain with a guaranteed accuracy in the approximation.
Let us first give an illustrative example for the parabolic case, considering a 1D advection-diffusion equation:

$$\begin{cases} y_s(x,s) - \varepsilon y_{xx}(x,s) + c y_x(x,s) + f(y(x,s)) = 0 \\ y(x,0) = y_0(x), \end{cases} \tag{4.14}$$

with $x \in [a,b], s \in [0,T], \varepsilon, c \in \mathbb{R}$.
We use a finite difference approximation for this equation based on an explicit Euler method in time combined with the standard centered approximation of the second order term and with an up-wind correction for the advection term. The snapshots will be taken from the sequence generated by the finite difference method. The final time is $T = 5$, the nonlinear term is $f \equiv 0$, moreover $a = -1$, $b = 4$. The initial condition is $y_0(x) = 5x - 5x^2$, when $0 \le x \le 1$, 0 otherwise.
For $\varepsilon = 0.05$ and $c = 1$ with only 3 POD basis functions, the approximation fails (see Figure 4.1). Note that in this case the advection is dominating the diffusion, a low number of POD basis functions will not suffice to get an accurate approximation (Figure 4.1 top-right). However, the adaptive method which only uses 3 POD basis functions will give accurate results (Figure 4.1 bottom-right).
The idea which is behind the adaptive method is the following: we do not consider all

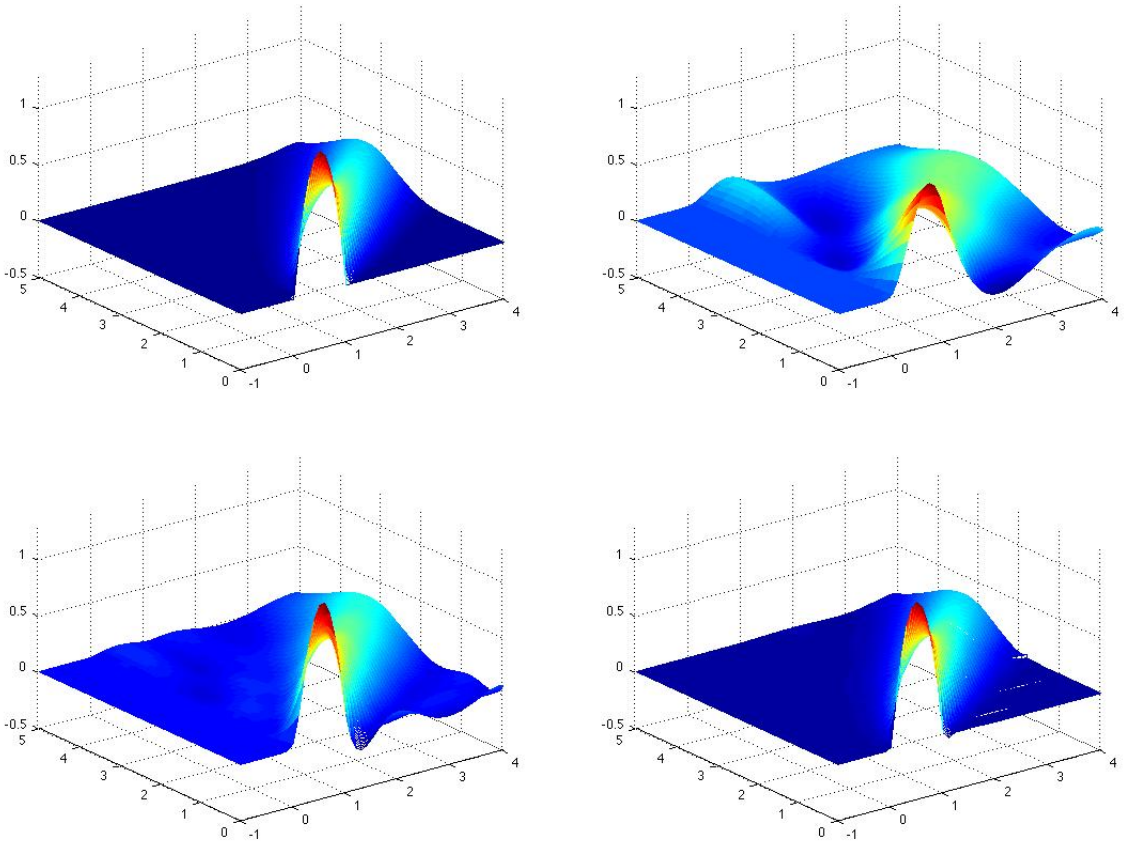

Figure 4.1.: Equation (4.14): solved with finite difference (top-left); POD-Galerkin approximation with 3 POD basis functions (top-right); solved via POD-Galerkin approximation with 5 POD basis functions (bottom-left); Adapting 3 POD basis functions (bottom-right).

the snapshots together in the whole interval $[0,T]$ but we group them. To this end we prefer to split $[0,T]$, in sub-intervals

$$[0,T] = \cup_{k=0}^{K-1}[T_k, T_{k+1}]$$

where K is a-priori unknown, $T_0 = 0$, $T_K = T$ and $T_k = t_i$ for some i. In this way, choosing properly the length of the $k-$th interval $[T_k, T_{k+1}]$, we consider only the snapshots falling in that sub-interval, by construction, we are sure to have enough snapshots in every sub-interval to compute POD basis functions. Then, we have enough informations in every sub-interval and we can apply the standard routines (explained in Section 2.1.2) to get a "time local" POD basis.
Now let us explain how to divide our time interval $[0,T]$. We will choose a parameter to check the accuracy of the POD approximation and define a threshold. Above that threshold we lose in accuracy so it is necessary to compute new POD basis functions. A good parameter to check the accuracy is $\mathcal{E}(\ell)$ (see (2.7)). The method to define the splitting of $[0,T]$ and the size of every sub-interval works as follows. We start computing the SVD of the matrix Y that gives us informations about our dynamics in the whole time interval. Then, we check the relative accuracy $\mathcal{E}(\ell)$ of the POD basis functions related to the snapthots of the interval we are considering. If that evaluation is not below the desired accuracy we split the snapshots into two subintervals and we compute again $\mathcal{E}(\ell)$, after having updated the POD basis functions. We iterate until for certain t_k that indicator is above the tolerance we set $T_1 = t_k$ and we divide the interval in two parts, $[0,T_1)$ and $(T_1,T]$. Now we just consider the snapshots related to the solution up to the time T_1. Then we iterate this idea until the indicator is below the threshold. When the first interval is found, we restart the procedure in the interval $[T_1,T]$ and we stop when we reach the final time T. Note that the extrema of every interval coincide by construction with one of our discrete times $t_i = i\Delta t$ so that the global solution is easily obtained linking all the sub-problems which always have a snapshot as initial condition. A low value for the threshold will also guarantee that we will not have big jumps passing from one sub-interval to the next. Once we know we got accurate POD basis functions we compute the solution of the problem in each sub-intervals. Moreover, in each interval $[T_k, T_{k+1}]$ we check the residual of the solution previously computed. For every $s \in [0,T]$, let us define the residual of the solution with respect to equation (4.16) without control:

$$\mathcal{R}(s) = \|y_s(x,s) - \varepsilon y_{xx}(x,s) + cy_x(x,s) + f(y(x,s))\|. \tag{4.15}$$

In our test we have always adopted the $L^1([0,T])$ norm. We have to choose a threshold for the residual. If $\mathcal{R}$ is below this value we can go on and solve the problem in the interval we are taking into account, otherwise we should divide again the interval. For instance, by bisection, we obtain two new intervals. We consider the snapshots falling in each interval to update the POD basis functions. After having updated the basis functions we can compute the optimal solution in each subdomain. Note that we are already working in a sub-interval selected according to $\mathcal{E}(\ell)$, so we do not need to check again that indicator but only to update the POD basis functions.

This idea can be applied also when we have a controlled dynamic (see (4.16)). First of all we have to decide how to collect the snapshots, since the control $u(t)$ is completely unknown. One can make a guess and use the dynamics and the functional corresponding to that guess, by these informations we can compute the POD basis. Once the POD basis is obtained we will get the optimal feedback law after having solved a reduced HJB equation as we already explained. Let us summarize the adaptive method in the following step-by-step Algorithm 8.

Algorithm 8: Adaptive POD algorithm (**Ad-POD**)

1: collect the snapshots in [0,T]
2: divide $[0,T]$ according to $\mathcal{E}(\ell)$
3: **for** $i = 0\, to\, K-1$ **do**
4: apply SVD to get the POD basis in each sub-interval $[T_i, T_{i+1}]$
5: discretize the space of controls
6: project the dynamics onto the (reduced) POD space
7: select the intervals for the POD reduced variables
8: solve the corresponding HJB in the reduced space for the interval $[T_i, T_{i+1}]$
9: check the norm of the residual of the solution in $[T_i, T_{i+1}]$.
10: **if** $\mathcal{R}(s) < \varepsilon_{\mathcal{R}}$ **then**
11: goto Step 17
12: **else**
13: Split again the interval into two sub-problem
14: Solve the corresponding HJB in the reduced space
15: goto Step 17 for both sub-problems.
16: **end if**
17: go back to the full domain space
18: **end for**

4.4. Numerical tests

In this section we present some numerical tests for the controlled heat equation and for the advection-diffusion equation with a non-linear source term and a quadratic cost functional. Consider the following advection-diffusion equation:

$$\begin{cases} y_s(x,s) - \varepsilon y_{xx}(x,s) + c y_x(x,s) + f(y(x,s)) = u(s) \\ y(x,0) = y_0(x), y(a,t) = \alpha,\ y(b,t) = \beta. \end{cases} \tag{4.16}$$

with $x \in [a,b]$, $s \in [0,T]$, $\varepsilon \in \mathbb{R}_+$ and $\alpha, \beta, c \in \mathbb{R}$.
Note that changing the parameters c and ε we can obtain the heat equation ($c = 0$) and the advection equation ($\varepsilon = 0$). In our tests the nonlinear term will be either $f(z) \equiv 0$ or $f(z) = \sigma \frac{z}{1+z}$ where $\sigma \in \mathbb{R}$. The functional to be minimized is:

$$J_{y_0,t}(u(\cdot)) = \int_0^T ||y(x,s) - \widehat{y}(x,s)||^2 + \gamma ||u(s)||^2 \, ds, \tag{4.17}$$

i.e. we want to stay close to a reference trajectory $\widehat{y}$ while minimizing the norm of u. Note that we dropped the discount factor setting $\lambda = 0$. Typically in our test problems $\widehat{y}$ is obtained by applying a particular control $\widehat{u}$ to the dynamics. The numerical simulations reported in this chapter have been made on a server SUPERMICRO 8045C-3RB with 2 cpu Intel Xeon Quad-Core 2.4 Ghz and 32 GB RAM under SLURM (`https://computing.llnl.gov/linux/slurm/`).

4.4.1. Test 1: Heat equation with smooth initial data

We compute the snapshots with a centered/forward Euler scheme with space step $\Delta x = 0.02$, and time step $\Delta t = 0.012$, $\varepsilon = 1/60, c = 0, R = 0.01$ and $T = 5$. The initial condition is $y_0(x) = 5x - 5x^2$, $f \equiv 0$ and $\widehat{y}(x, s) = 0$. In Figure 4.2 we compare four different approximations concerning the heat equation: (a) is the solution for $\widehat{u}(t) = 0$, (b) is its approximation via POD (non adaptive), (c) is the direct LQR solution computed by MATLAB without POD and, finally, the approximate optimal solution obtained coupling POD and HJB. The approximate value function is computed for $\Delta t = 0.1$ and $\Delta x = 0.1$ whereas the optimal trajectory as been obtained with $\Delta t = 0.01$. Test 1 and even Test 2 have been solved in about half an hour of CPU time.
Note that in this example the approximated solution is rather accurate because the regularity of the solution is high due to the diffusion term. Since in the limit the solution tends to the average value the choice of the snapshots will not affect too much the solution, i.e. even with a rough choice of the snapshots will give us a good approximation, therefore we have not applied the adaptive technique. The difference between Figure 4.2 (bottom-left) and Figure 4.2 (bottom-right) is due to the fact that the control space is continuous for Figure 4.2 (bottom-left) and discrete for Figure 4.2 (bottom-right) where the small oscillations are due to the low accuracy in the computation of the value function.

4.4.2. Test 2: Heat equation with non-smooth intial data

In this section we change the initial condition with a function which is only Lipschitz continuos: $y_0(x) = 1 - |x|$. According to Test 1, we consider the same parameters. (see Figure 4.3). Riccati's equation has been solved by a MATLAB LQR routine. Thus, we have used the solution given by this routine as the correct solution in order to compare the errors in L^1 and L^2 norm between the reduced Riccati's equation and our approach based on the reduced HJB equation. Since we do not have any information, the snapshots are computed for $\widehat{u} = 0$. This is only a guess, but in the parabolic case it fits well due to the diffusion term.

As in Test 1, the choice of the snapshots does not effect strongly the approximation due to the asymptotic behavior of the solution and the adaptive method was not applied at all. The presence of a Lipschitz continuous initial condition has almost no influence on the global error (see Table 4.3).

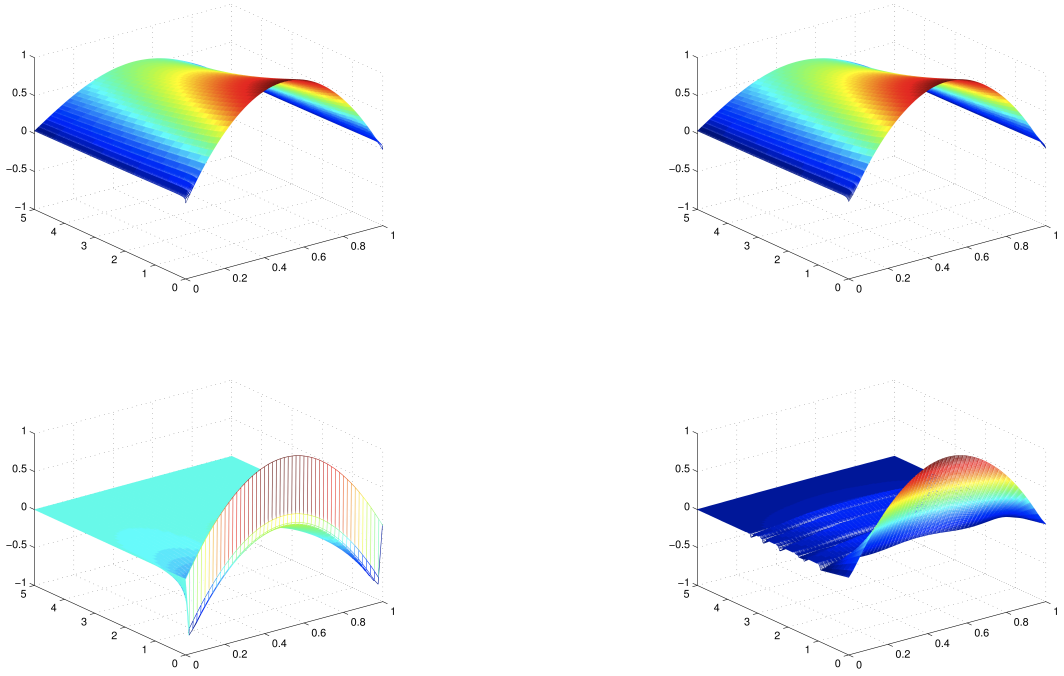

Figure 4.2.: Test 1:Heat Equation without control (top-left); Heat Equation without control and 3 POD basis functions (top-right); Controlled solution with LQR-MATLAB (bottom-left); Approximate solution with POD (3 basis functions) and HJB (bottom-right).

	L^1	L^2
$y^{LQR}(\cdot,T) - y^{POD+LQR}(\cdot,T)$	0.0221	0.0172
$y^{LQR}(\cdot,T) - y^{POD+HJB}(\cdot,T)$	0.0204	0.0171

Table 4.3.: Test 2: L^1 and L^2 errors at time T for the optimal approximate solution.

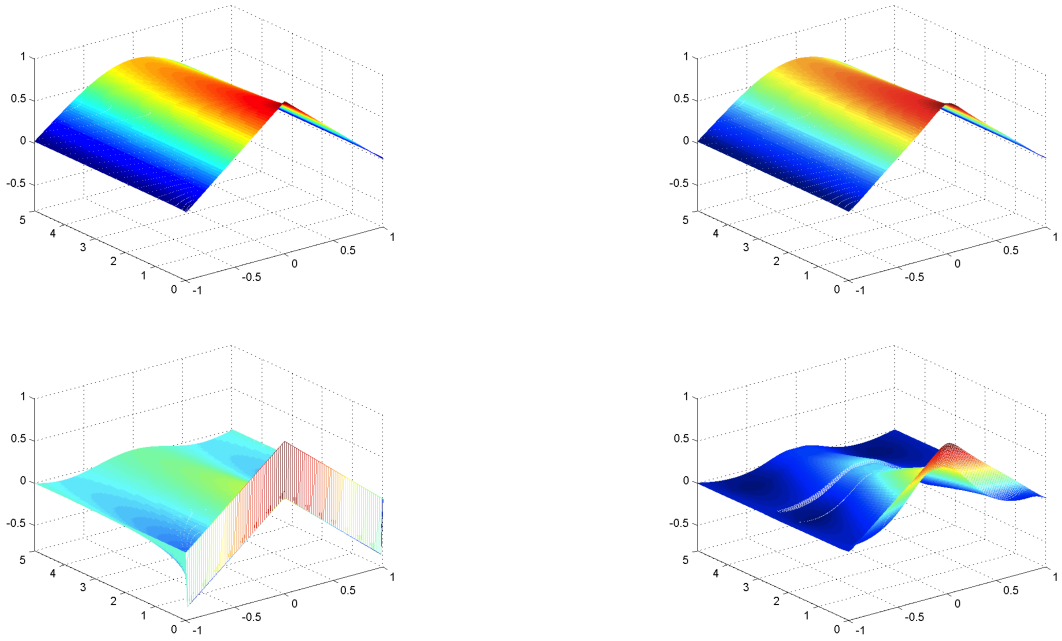

Figure 4.3.: Test 2: exact solution for $\widehat{u} = 0$ (top-left); Exact solution for $\widehat{u} = 0$ POD (3 basis functions) (top-right); Approximate optimal solution for LQR-MATLAB (bottom-left); Approximate solution with POD (3 basis functions)and HJB (bottom-right).

4.4.3. Test 3: Balance Truncation and POD for the control of heat equation

This test compares the solution of the LQR problem introduced in Section 4.4.1 obtained with the full model and two reduction techniques as POD and BT (see Figure 4.4). The initial condition in (4.14) is $y_0(x) = 2\sin(\pi x)$, $c = 0, \varepsilon = \dfrac{1}{100}$. The horizon is $T = 3$ and the desired state is $\widehat{y}(x,t) \equiv 0$. Moreover we took $f \equiv 0, \alpha = 0 = \beta, \gamma = 10^{-10}$. In this case the control in (4.16) is multiplied by an indicator functions equal to one when $x = 0.24, 0.49, 0.74$ and zero elsewhere. As we can see from Figure 4.4 balance truncation is much similar of the reference controlled solution. We have to say that the snapthots in the POD case are computed taking $u \equiv 0$. The error between the BT approximation and the reference solution is 0.03 whereas in the POD case is 0.05. At the bottom of Figure 4.4 we can see the basis functions obtained with POD and with BT method.

4.4.4. Test 4: Advection-Diffusion equation

The advection-diffusion equation needs a different method. We can not use the same $\widehat{y}$ we had in the parabolic case, mainly because in Riccati's equation the control is free and is not bounded, on the contrary when we solve an HJB we have to discretize the space of controls. We modified the problem in order to deal with bang-bang controls. We get $\widehat{y}$ in (4.17) just plugging in the control $\widehat{u} \equiv 0$. We have considered the control space corresponding only to three values in $[-1, 1]$, then $U = \{-1, 0, 1\}$. We first have tried to get a controlled solution, without any adaptive method and, as expected, we obtained a bad approximation (see Figure 4.5). From Figure 4.5 it is clear that POD with four basis functions is not able to catch the behavior of the dynamics, so we have applied our adaptive method.

We have considered: $T = 3, \Delta x = 0.1, \Delta t = 0.008$, $a = -1$, $b = 4$, $R = 0.01$ and $f \equiv 0$. According to our algorithm, the time interval $[0, 3]$ was divided into $[0, 0.744] \cup [0.744, 1.496] \cup [1.496, 3]$. As we can see our last interval is bigger than the others, this is due to the diffusion term (see Figure 4.6). The L^2-error is 0.0761, and the computation of the optimal solution via HJB has required about six hours of CPU time. In Figure 4.5 we compare the exact solution with the numerical solution based on a POD representation. Note that, in this case, the choice of only 4 basis functions for the whole interval $[0, T]$ gives a very poor result due to the presence of the advection term. Looking at Figure 4.6 one can see the improvement of our adaptive technique which takes always 4 basis functions in each sub-interval.

In order to check the quality of our approximation we have computed the numerical residual, defined as:

$$\mathcal{R}(s) = \|y_s(x,s) - \varepsilon y_{xx}(x,s) + c y_x(x,s) - u(s)\|.$$

The residual for the solution of the control problem computed without our adaptive technique is 1.1, whereas the residual for the adaptive method is $2 \cdot 10^{-2}$. As expected from the pictures, there is a big difference between these two values.

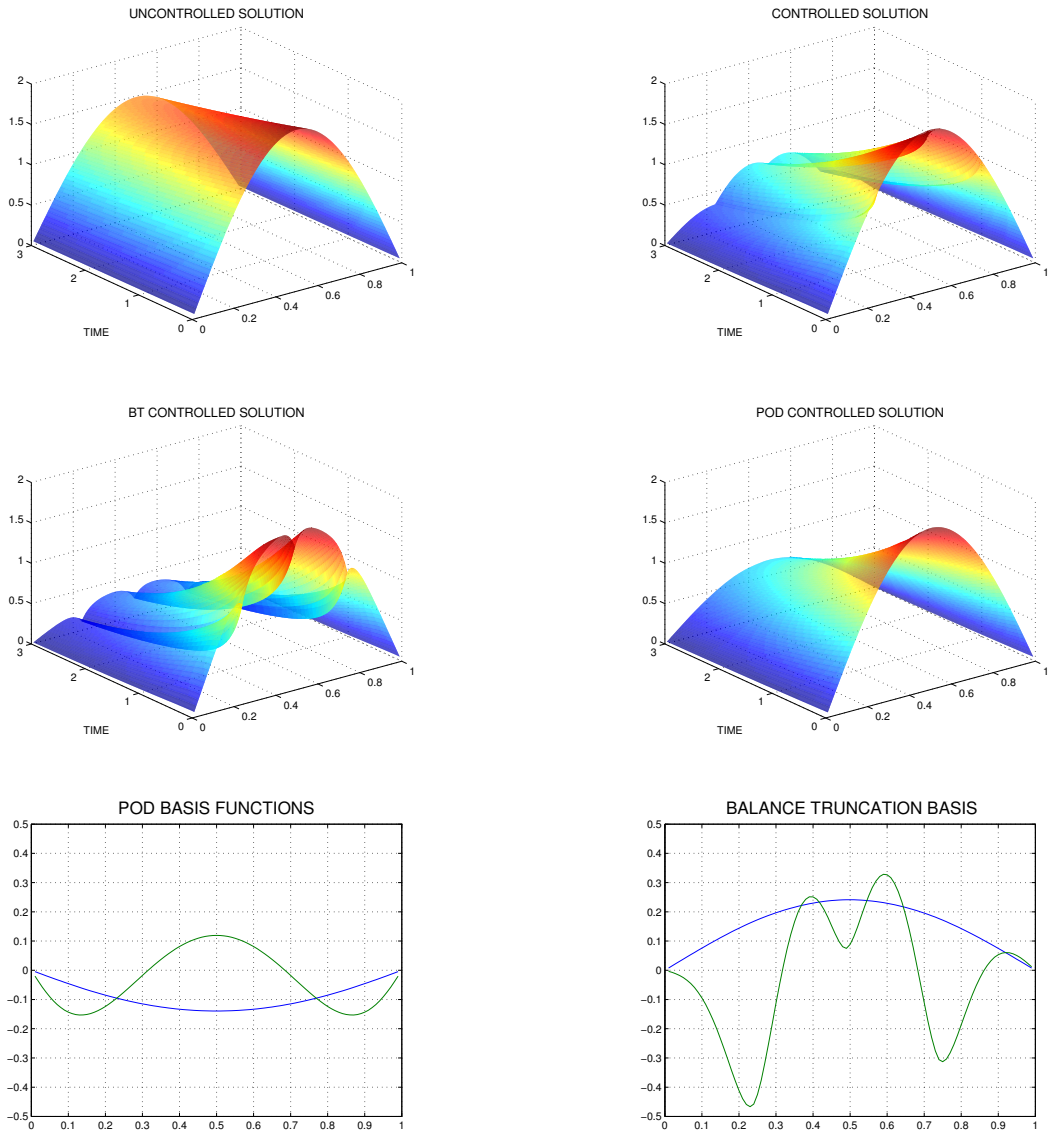

Figure 4.4.: Test 3: Heat Equation without control (top-left); LQR solution (top-right); Controlled solution with BT (middle-left); Controlled Solution with POD (2 basis functions) (middle-right); Balance Truncations basis functions (bottom-left); POD Basis functions (bottom-right).

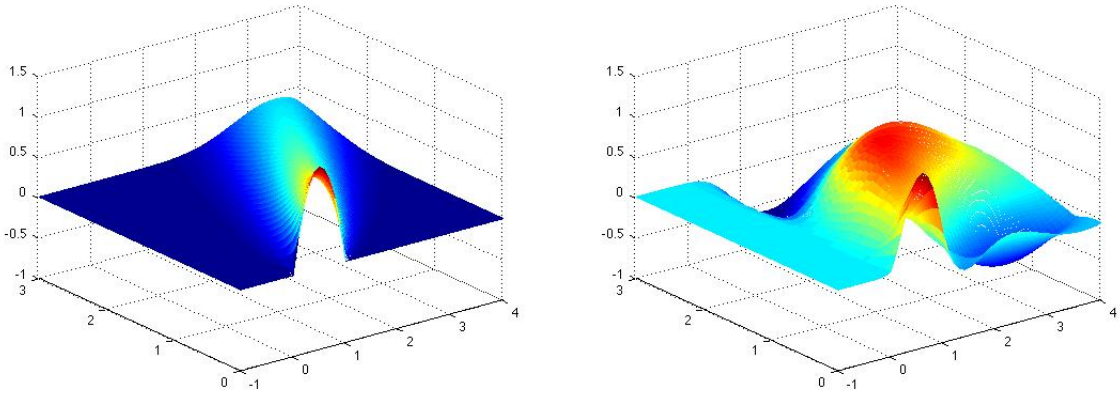

Figure 4.5.: Test 4: Solution $\widehat{y}$ on the left, approximate solution on the right with POD (4 basis functions).

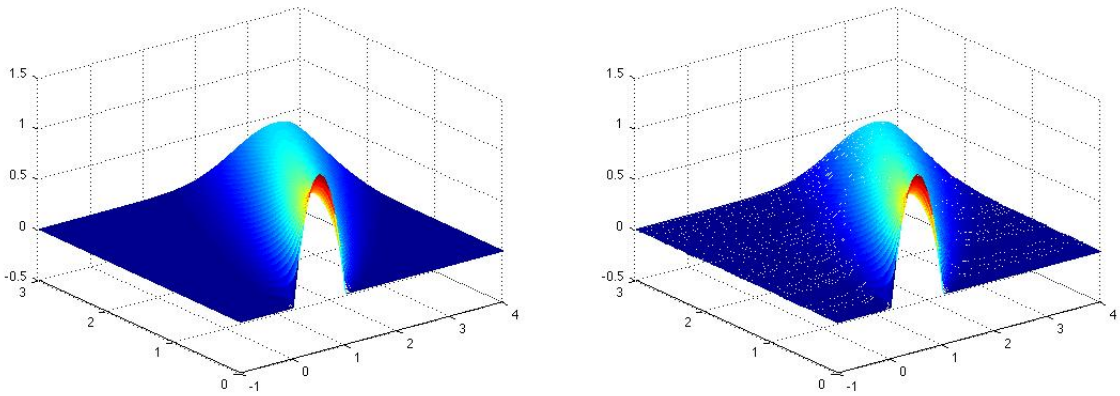

Figure 4.6.: Test 4: Solution for $\widehat{u} \equiv 0$ (left), approximate optimal solution (right).

4.4.5. Test 5: Advection-Diffusion equation

In this test we take a different $\widehat{y}$, namely the solution of (4.16) corresponding to the control

$$\widehat{u}(t) = \begin{cases} -1 & 0 \leq t < 1 \\ 0 & 1 \leq t < 2 \\ 1 & 2 \leq t \leq 3. \end{cases}$$

We want to emphasize we can obtain nice results when the space of controls has few elements. The parameters were the same used in Test 4. The L^2-error is 0.09, and the time was the same we had in Test 4. In Figure 4.7 we can see our approximation. In

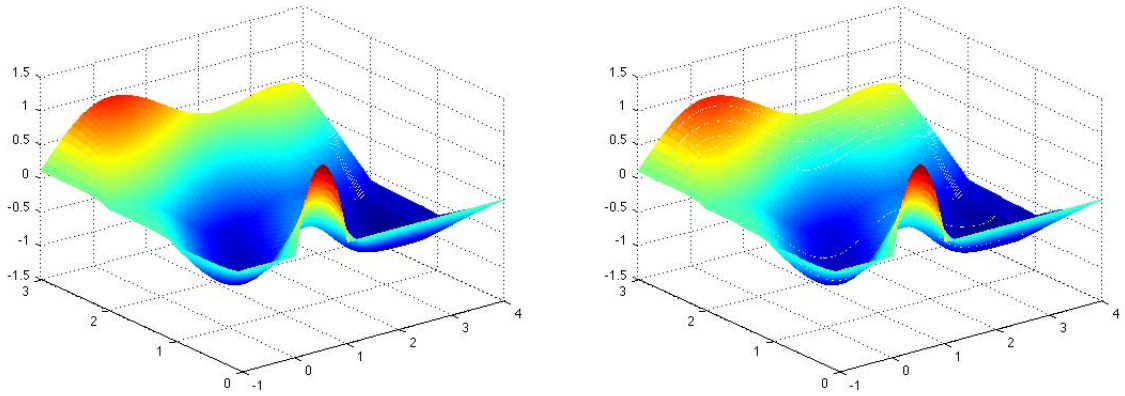

Figure 4.7.: Test 5: Solution for $\widehat{u}$ (left), approximate optimal solution (right).

Figure 4.7 one can see that the adaptive technique can also deal with discontinuous controls.
In this test, the residual for the solution of the control problem without our adaptive technique is 2, whereas the residual for the adaptive method is $3 \cdot 10^{-2}$. Again, the residual shows the higher accuracy of the adaptive routine.

4.4.6. Test 6: A nonlinear heat equation

We compute the snapshots with a centered/forward Euler scheme with space step $\Delta x = 0.05$, and time step $\Delta t = 0.0125$, $f(z) = \sigma \frac{z}{1+z}, \varepsilon = 1/10, c = 0, \gamma = 0.01, a = 0, b = 1, \sigma = 10$ and $T = 1$. The initial condition is $y_0(x) = 0, \alpha = 1, \beta = 1$. Δt and Δx are chosen according to the CFL condition for parabolic problems (see [80]). One may avoid this restriction performing an implicit scheme. In Figure 4.8 we compare three different approximations concerning the nonlinear heat equation: the first one is the solution for $\widehat{u}(t) = 0$, the second one is its approximation via POD (non adaptive), and the last one approximate optimal solution obtained coupling POD and HJB. The approximate value function is computed for $\Delta t = 0.1$ $\Delta x = 0.1$ whereas the optimal trajectory as been obtained with $\Delta t = 0.0125$.

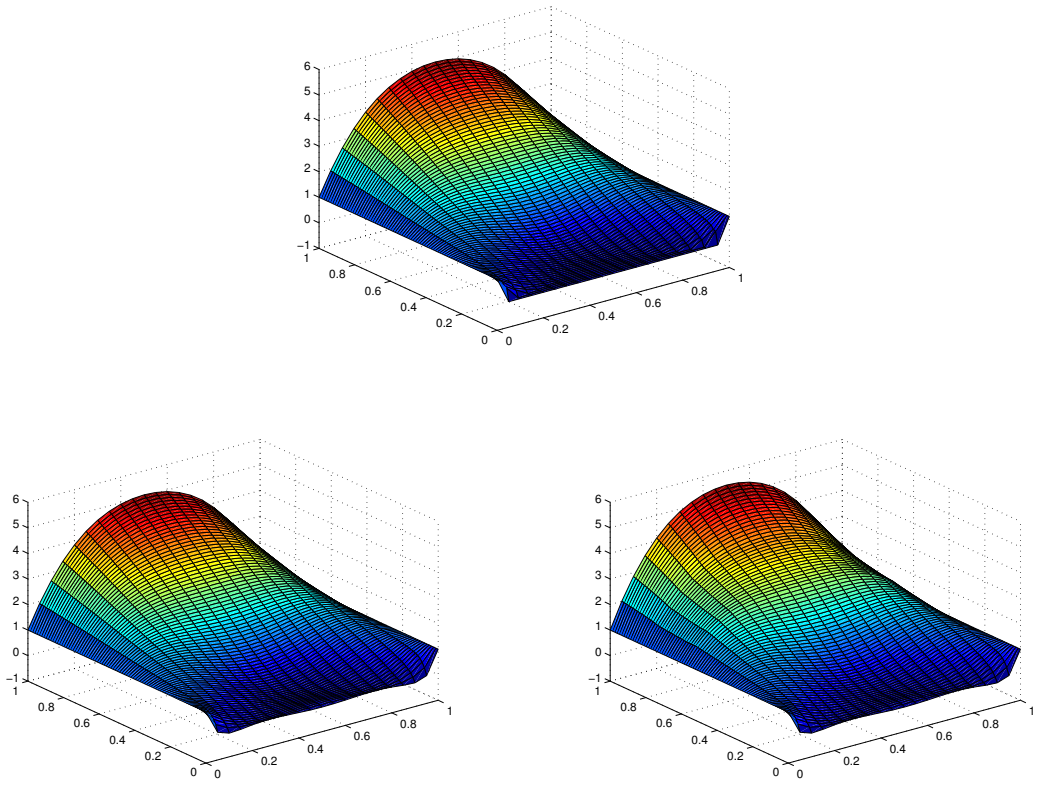

Figure 4.8.: TEST 6: Snapshots (top), POD approximation (bottom-left), controlled solution (bottom-right).

As we can see in Figure 4.8 we are able to reproduce the behavior of the reference solution either with only POD method either coupling POD method and dynamic programming. The function $\widehat{y}(x,s)$ is the solution of (4.16) plugging the control chosen before. We have considered the control space corresponding only to three values in $[-1,1]$, then $U=\{-1,0,1\}$. The control space is not bang-bang, in fact the control 0 will turn out to be the optimal solution. Note that in this example the approximate solution is rather accurate because the regularity of the solution is high due to the fact that the diffusion term is dominating. The nonlinear term do not force us to increase the number of POD basis functions or, in our case, to deal with an adaptive method. The error computed with ℓ^1-norm is $1.14\cdot e-04$ while the max norm is 0.1272. With this tests we want to emphasize the quality of the POD method when there are not any advection terms in the equation, even for nonlinear equations. Indeed the adaptive method was not applied.

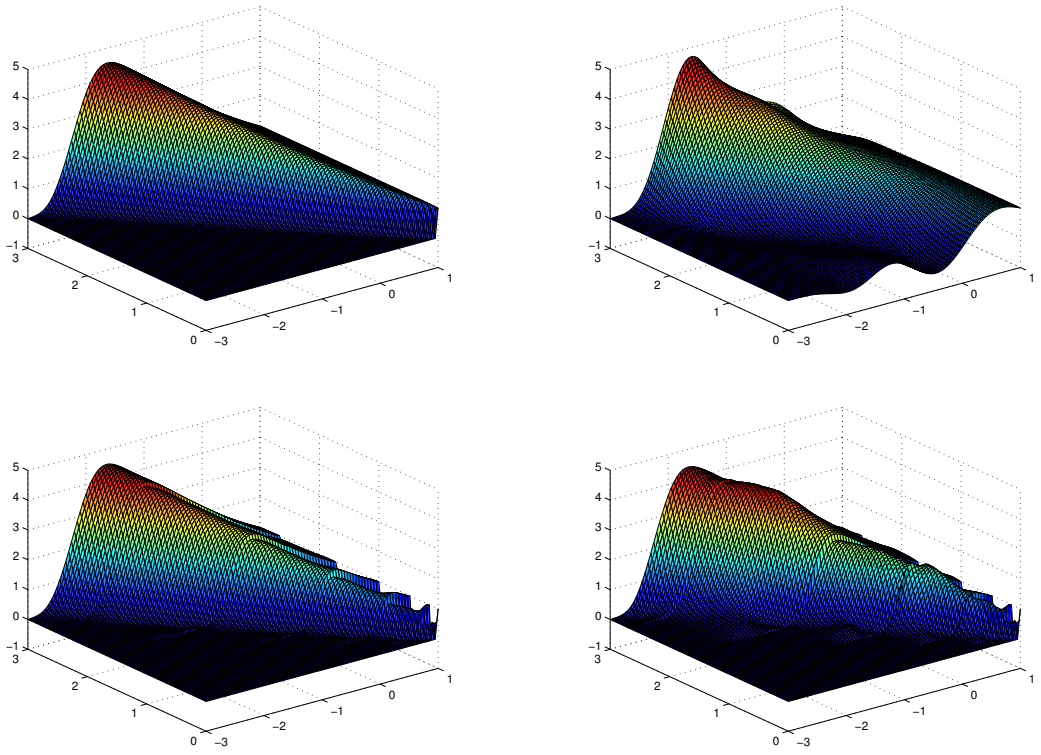

Figure 4.9.: Test 7: Snapshots (top-left), POD approximation without adaptive method (top-right), POD approximation with adaptive method (bottom-left), controlled approximation with adaptive method (bottom-right)

4.4.7. Test 7: A nonlinear advection-diffusion equation

The advection-diffusion equation needs a different approach. We get $\widehat{y}$ in (4.16) just plugging in the control $\widehat{u} \equiv 0$. We have considered the same control space we had in Test 6. We first have tried to get a controlled solution, without any adaptive method and, as expected, we obtained a bad approximation (see Figure 4.9). This is why we have to apply an adaptive method, if we do not get *good* POD basis functions it will not be possible to have a accurate approximation for the controlled equation. We have considered: $T = 3, \Delta x = 0.05, \Delta t = 0.025$, $a = -3$, $b = 1$, $\alpha = 0, \beta = 1, \sigma = 2, y_0(x) = 0$, $\gamma = 0.01$ and $f(x) = \sigma \frac{x}{1+x}$. According to our method, the time interval $[0, 3]$ was divided into six sub-intervals. In Figure 4.9 we compare the exact solution with the numerical solution based on a POD representation. Note that, in this case, the choice of only 4 basis functions for the whole interval $[0, T]$ gives a very poor result due to the presence of the advection term. Looking at Figure 4.9 one can see the improvement of the adaptive technique which takes always 4 basis functions in each sub-interval.
We can see that our adaptive method is able to follow the right direction of the solution.

The approximate solution is not smooth but this is normal since we are changing the basis function several times. In order to check the quality of our approximation we have computed the numerical residual and the error in ℓ^1-norm and $\ell^\infty-$norm. As we can see in Table 4.9 there is a big difference if we consider the adaptive method or not, expecially when we deal with ℓ^1-norm. Table 4.9 presents errors and residuals corresponding to

	POD	Ad-POD	Ad-POD+HJB
ℓ^1 error	1.338	0.303	0.5016
ℓ^∞ error	1.1801	0.92	0.8723
ℓ^1 residual	1.33	0.29	0.29

Table 4.9.: TEST 7: Errors and residuals for different approximations.

different approximations: POD solution (second column), adaptive POD method (third column) and controlled solution computed by the adaptive method (fourth column). As we can see, there is a big difference when we compute the ℓ^1- error in the case of the adaptive or the non-adaptive method. We can significatly decrease, split in half, the error. Of course, in the simulation which involves the dynamic approach there are other errors due to the numerical approximation of HJB equations. The $\ell^\infty-$ norm is meaningless since it computes only the max error in all the interval and it may happen, for instance, to have a big gap somewhere due to the adaptive method or to the coarse control discrete space we deal with.

5. Asymptotic Stability of POD based Model Predictive Control for a semilinear parabolic PDE

Nonlinear model predictive control (NMPC) is a method to approximately synthesize time infinite horizon optimal feedback laws from the iterative solution of finite horizon optimal control problems.We analyze the stability of NMPC without terminal constraints applied to a semilinear parabolic equation with advection term. A minimal finite horizon is determined to guarantee the stabilization of the system. Furthermore, we will deal with constrained optimal control problems. Since the minimal horizon can be large, the numerical approximation is very expensive. Therefore, POD model reduction is applied to substantially reduce the computational cost by computing suboptimal solutions. Numerical examples will illustrate the efficiency of the method.

5.1. Formulation of the control system

Let $\Omega = (0,1) \subset \mathbb{R}$ be the spatial domain. For the initial time $t_\circ \in \mathbb{R}_0^+ = \{s \in \mathbb{R} \,|\, s \geq 0\}$ we define the space–time cylinder $Q = \Omega \times (t_\circ, \infty)$. By $H = L^2(\Omega)$ we denote the Lebesgue space of (equivalence classes of) functions which are (Lebesgue) measurable and square integrable. We endow H by the standard inner product – denoted by $\langle \cdot\,, \cdot \rangle_H$ – and the associated induced norm $\|\varphi\|_H = \langle \varphi, \varphi \rangle_H^{1/2}$. Furthermore, $V = H_0^1(\Omega) \subset H$ stands for the Sobolev space

$$V = \left\{ \varphi \in H \,\middle|\, \varphi \text{ admits a weak derivative } \varphi' \in H \text{ and } \varphi(0) = \varphi(1) = 0 \right\}.$$

Recall that both H and V are Hilbert spaces. In V we use the inner product

$$\langle \varphi, \phi \rangle_V = \int_\Omega \varphi'(x)\phi'(x)\,\mathrm{d}x, \quad \text{for } \varphi, \phi \in V,$$

and set $\|\varphi\|_V = \langle \varphi, \varphi \rangle_V^{1/2}$ for $\varphi \in V$. For more details on Lebesgue and Sobolev spaces we refer the reader to [34], for instance. When the time t is fixed for a given function $\varphi : Q \to \mathbb{R}$, the expression $\varphi(t)$ stands for a function $\varphi(\cdot\,, t)$ considered as a function in Ω only.

We consider the following control system governed by the following semilinear parabolic

partial differential equation: $y = y(x,t)$ solves the initial boundary value problem

$$y_t - \theta y_{xx} + y_x + \rho(y^3 - y) = u \qquad \text{in } Q, \tag{5.1a}$$

$$y(0,\cdot) = y(1,\cdot) = 0 \qquad \text{in } (t_\circ, \infty), \tag{5.1b}$$

$$y(t_\circ) = y_\circ \qquad \text{in } \Omega. \tag{5.1c}$$

In (5.1a) it is assumed that the control $u = u(x,t)$ belongs to the set of admissible control inputs

$$\mathbb{U}_{ad}(t_\circ) = \big\{ u \in L^2(t_\circ, \infty; H) \,\big|\, u \in U_{ad} \text{ almost everywhere (a.e.) } (x,t) \in Q \big\},$$

where $\mathbb{U}(t_\circ) = L^2(t_\circ, \infty; H)$ and $U_{ad} = \{u \in \mathbb{R} \,|\, u_a \le u \le u_b\}$ and u_a, u_b are given constants with $u_a \le 0 \le u_b$. The parameters θ and ρ satisfy

$$(\theta, \rho) \in D_{ad} = \big\{ (\tilde\theta, \tilde\rho) \in \mathbb{R}^2 \,\big|\, \theta_a \le \tilde\theta \text{ and } \rho_a \le \tilde\rho \big\}$$

with positive constants θ_a and ρ_a. Further, in (5.1c) the initial condition $y_\circ = y_\circ(x)$ is supposed to belong to H.

A solution to (5.1) is interpreted in the weak sense as follows: for given $(t_\circ, y_\circ) \in \mathbb{R}_0^+ \times H$ and $u \in \mathbb{U}_{ad}(t_\circ)$ we call y a *weak solution* to (5.1) for fixed $(\theta,\rho) \in D_{ad}$ if $y(t) \in V$, $y_t(t) \in V'$ holds a.e. $t \ge t_\circ$ and y satisfies $y(t_\circ) = y_\circ$ in H as well as

$$\frac{\mathrm{d}}{\mathrm{d}t} \langle y(t), \varphi \rangle_H + \int_\Omega \theta y_x(t) \varphi' + \big(y_x(t) + \rho(y^3(t) - y(t))\big)\varphi \,\mathrm{d}x = \int_\Omega u(t)\varphi \,\mathrm{d}x \tag{5.2}$$

for all $\varphi \in V$ and a.e. $t > t_\circ$. The following result is proved in [28], for instance.

Proposition 5.1 *For given $(t_\circ, y_\circ) \in \mathbb{R}_0^+ \times H$ and $u \in \mathbb{U}_{ad}(t_\circ)$ there exists a unique weak solution $y = y_{[u,t_\circ,y_\circ]}$ to (5.1) for every $(\theta,\rho) \in D_{ad}$.*

Let $(t_\circ, y_\circ) \in \mathbb{R}_0^+ \times H$ be given. Due to Proposition 5.1 we define the quadratic cost functional:

$$J_{y_\circ,t_\circ}(u) := \frac{1}{2} \int_{t_\circ}^\infty \|y_{[u,t_\circ,y_\circ]}(t) - \widehat{y}\|_H^2 \mathrm{d}t + \frac{\lambda}{2} \int_{t_\circ}^\infty \|u(t)\|_H^2 \,\mathrm{d}t \tag{5.3}$$

for all $u \in L^2(Q) \supset \mathbb{U}_{ad}(t_\circ)$, where $y_{[u,t_\circ,y_\circ]}$ denotes the unique weak solution to (5.1). We suppose that $\widehat{y} = \widehat{y}(x)$ is a given desired state in H (e.g., the equilibrium $\widehat{y} = 0$) and that $\lambda > 0$ denotes a fixed weighting parameter. Then we consider the nonlinear infinite horizon optimal control problem

$$\min J_{y_\circ,t_\circ}(u) \quad \text{subject to} \quad u \in \mathbb{U}_{ad}(t_\circ). \tag{5.4}$$

Existence results for this optimal control problem can be found in [10]. Suppose that the trajectory y is measured at discrete time instances

$$t_n = t_\circ + n\Delta t, \quad n \in \mathbb{N}$$

where the time step $\Delta t > 0$ stands for the time step between two measurements. Thus, we want to select a control $u \in \mathbb{U}_{ad}(t)$ such that the associated trajectory $y_{[u,t_\circ,y_\circ]}$ follows a given desired state $\widehat{y}$ as good as possible. This problem is called a *tracking problem*, and, if $\widehat{y} = 0$ holds, a *stabilization problem*.

Since our goal is to be able to react to the current deviation of the state y at time $t = t_n$ from the given reference value $\widehat{y}$, we would like to have the control in *feedback form*, i.e., we want to determine a mapping $\mu : H \to \mathbb{U}_{ad}(t_\circ)$ with $u(t) = \mu(y(t_n))$ for $t \in [t_n, t_{n+1}]$.

5.2. Nonlinear model predictive control

We present a *nonlinear model predictive control* (NMPC) approach to compute a mapping μ which allows a representation of the control in feedback form. This strategy is also known as *receding horizon control* (RHC) method. For more details we refer the reader to the monographs [52, 82], for instance.

5.2.1. The NMPC method

In this subsection we will introduce the NMPC algorithm. For that purpose we write the weak form of our control system (5.1) as a parametrized nonlinear dynamical system. Let us introduce the θ-dependent linear operator $\mathcal{A}$ which maps the space V into its dual space V' as follows:

$$\mathcal{A}\varphi = -\theta\varphi_{xx} + \varphi_x \in V' \quad \text{for } \varphi \in V \text{ and } \theta \geq \theta_a.$$

Moreover, let $f(\cdot)$ be a mapping from V into V' given by

$$f(\varphi) = \rho(\varphi^3 - \varphi) \in V' \quad \text{for } \varphi \in V \text{ and } \rho \geq \rho_a.$$

Setting $\mathcal{F}(\varphi, v) = \mathcal{A}\varphi + f(\varphi) - v$ for $\varphi \in V$, $v \in H$ and $(\theta, \rho) \in D_{ad}$ we can express (5.2) as the nonlinear dynamical system

$$y'(t) = \mathcal{F}(y(t), u(t)) \in V' \text{ for all } t > t_\circ, \quad y(t_\circ) = y_\circ \text{ in } H, \tag{5.5}$$

for given $(t_\circ, y_\circ) \in \mathbb{R}_0^+ \times H$. The cost functional has been already introduced in (5.3). Summarizing, we want to solve the following infinite horizon minimization problem

$$\min J_{y_\circ,t_\circ}(u) = \int_{t_\circ}^{\infty} L\big(y_{[u,t_\circ,y_\circ]}(t), u(t)\big)\,\mathrm{d}t \quad \text{subject to} \quad u \in \mathbb{U}_{ad}(t_\circ), \tag{5.6}$$

where we have defined the running quadratic cost

$$L(\varphi, v) = \frac{1}{2}\left(\|\varphi - \widehat{y}\|_H^2 + \lambda \|v\|_H^2\right) \quad \text{for } \varphi, v \in H. \tag{5.7}$$

If we have determined a state feedback μ for (5.6), the control $u(t) = \mu(y(t_n))$, $n \in \mathbb{N}_0$, allows a closed loop representation for $t \in [t_\circ, \infty)$. Then, for a given initial condition $y_0 \in H$ we set $t_\circ = 0$, $y_\circ = y_0$ in (5.5) and insert μ to obtain the closed-loop form

$$\begin{aligned} y'(t) &= \mathcal{F}(y(t), \mu(y(t))) \quad \text{in } V' \text{ for } t \in (t_\circ, \infty), \\ y(t_\circ) &= y_\circ \qquad\qquad\qquad\quad \text{in } H. \end{aligned} \tag{5.8}$$

Although an infinite horizon problem may be very hard to solve due to the dimensionality of the problem, it guarantees the stabilization of the problem. This is a very important issue for optimal control problems. In an NMPC algorithm a state feedback law is computed for (5.6) by solving a sequence of finite time horizon problems.

To formulate the NMPC algorithm we introduce the finite horizon quadratic cost functional as follows: for $(t_\circ, y_\circ) \in \mathbb{R}_0^+ \times H$ and $u \in \mathbb{U}_{ad}^N(t_\circ)$ we set

$$J_{y_\circ, t_\circ}^N(u) = \int_{t_\circ}^{t_\circ^N} L\big(y_{[u,t_\circ,y_\circ]}(t), u(t)\big)\,\mathrm{d}t,$$

where N is a natural number, $t_\circ^N = t_\circ + N\Delta t$ is the final time and $N\Delta t$ denotes the length of the time horizon for the chosen time step $\Delta t > 0$. Further, we introduce the Hilbert space $\mathbb{U}^N(t_\circ) = L^2(t_\circ, t_\circ^N; H)$ and the set of admissible controls

$$\mathbb{U}_{ad}^N(t_\circ) = \big\{ u \in \mathbb{U}^N(t_\circ) \,\big|\, u(x,t) \in U_{ad} \text{ a.e. } (x,t) \in Q^N \big\}$$

with $Q^N = \Omega \times (t_\circ, t_\circ^N) \subset Q$. In Algorithm 9 the method is presented.
In each iteration over n we store the optimal control on the first time interval $[t_n, t_{n+1}]$

Algorithm 9: Nonlinear MPC algorithm (**NMPC**)

Require: time step $\Delta t > 0$, finite control horizon $N \in \mathbb{N}$, weighting param. $\lambda > 0$.

1: **for** $n = 0, 1, 2, \ldots$ **do**
2: Measure the state $y(t_n) \in H$ of the system at $t_n = n\Delta t$.
3: Set $t_\circ = t_n = n\Delta t$, $y_\circ = y(t_n)$ and compute a global solution to

$$\min \widehat{J}^N(u; t_\circ, y_\circ) \quad \text{subject to} \quad u \in \mathbb{U}_{ad}^N(t_\circ). \tag{5.9}$$

We denote the obtained optimal control by $\bar{u}^N$.
4: Define the NMPC feedback value $\mu^N(y(t)) = \bar{u}^N(t), t \in (t_\circ, t_\circ + \Delta t]$ and use this control to compute the associated state $y = y_{[\mu^N(\cdot), t_\circ, y_\circ]}$ by solving (5.5) on $[t_\circ, t_\circ + \Delta t]$.
5: **end for**

and the associated optimal trajectory of the sampling time. Then, we initialize a new finite horizon optimal control problem whose initial condition is given by the optimal trajectory $\bar{y}(t) = y_{[\mu^N(\cdot), t_\circ, y_\circ]}(t)$ at $t = t_\circ + \Delta t$ using the optimal control $\mu^N(y(t)) = \bar{u}(t)$

for $t \in (t_\circ, t_\circ + \Delta t]$. We iterate this process. Of course, the larger the horizon is, the better approximation one can have, but we would like to have the minimal horizon which can guarantee stability (see [8]). Note that (5.9) is an open loop problem on a finite time horizon $[t_\circ, t_\circ + N\Delta t]$ which will be studied in Section 5.3.

5.2.2. Dynamic programming principle and asymptotic stability

We, briefly, recall the essential theoretical results from DPP and stability analysis. The *value function* v is defined as follows for an infinite horizon optimal control problem:

$$v(t_\circ, y_\circ) := \inf_{u \in \mathbb{U}_{ad}(t_\circ)} J_{y_\circ, t_\circ}(u) \quad \text{for } (t_\circ, y_\circ) \in \mathbb{R}_0^+ \times H.$$

Let $N \in \mathbb{N}$ be chosen. Due to the *dynamic programming principle* (DPP) the value function v satisfies for any $k \in \{1, \dots, N\}$, with $t_\circ^k = t_k + k\Delta t$:

$$v(t_\circ, y_\circ) = \inf_{u \in \mathbb{U}_{ad}^k(t_\circ)} \left\{ \int_{t_\circ}^{t_\circ + k\Delta t} L\big(y_{[u,t_\circ,y_\circ]}(t), u(t)\big)\, \mathrm{d}t + v\big(y_{[u,t_\circ,y_\circ]}(t_\circ + k\Delta t)\big) \right\},$$

which holds under very general conditions on the data; see, e.g., [15] for more details. The value function for the finite horizon problem (5.9) is of the following form:

$$v^N(t_\circ, y_\circ) = \inf_{u \in \mathbb{U}_{ad}^N(t_\circ)} J_{y_\circ, t_\circ}^N(u) \quad \text{for } (t_\circ, y_\circ) \in \mathbb{R}_0^+ \times H.$$

The value function v^N satisfies the DPP for the finite horizon problem for $t_\circ + k\Delta t$, $0 < k < N$:

$$\begin{aligned} &v^N(t_\circ, y_\circ) \\ &\quad = \inf_{u \in \mathbb{U}_{ad}^k(t_\circ)} \left\{ \int_{t_\circ}^{t_\circ + k\Delta t} L\big(y_{[u,t_\circ,y_\circ]}(t), u(t)\big)\, \mathrm{d}t + v^N\big(y_{[u,t_\circ,y_\circ]}(t_\circ + k\Delta t)\big) \right\}. \end{aligned}$$

Nonlinear stability properties can be expressed by comparison functions; see, e.g., [52, Definition 2.13].

Definition 5.1 *We define the following classes of comparison functions:*

$$\begin{aligned} \mathcal{K} &= \big\{\beta : \mathbb{R}_0^+ \to \mathbb{R}_0^+ \,\big|\, \beta \text{ is continuous, strictly increasing and } \beta(0) = 0\big\}, \\ \mathcal{K}_\infty &= \big\{\beta : \mathbb{R}_0^+ \to \mathbb{R}_0^+ \,\big|\, \beta \in \mathcal{K},\ \beta \text{ is unbounded}\big\}, \\ \mathcal{L} &= \Big\{\beta : \mathbb{R}_0^+ \to \mathbb{R}_0^+ \,\Big|\, \beta \text{ is continuous, strictly decreasing, } \lim_{t \to \infty} \beta(t) = 0\Big\}, \\ \mathcal{KL} &= \big\{\beta : \mathbb{R}_0^+ \times \mathbb{R}_0^+ \to \mathbb{R}_0^+ \,\big|\, \beta \text{ is continuous, } \beta(\cdot\,, t) \in \mathcal{K},\ \beta(r, \cdot) \in \mathcal{L}\big\}. \end{aligned}$$

Using a comparison function $\beta \in \mathcal{KL}$ we introduce the concept of asymptotic stability; see, e.g. [52, Definition 2.14].

Definition 5.2 *Let Let* $y_{[\mu(\cdot),t_\circ,y_\circ]}$ *be the solution to* (5.8) *and* $y_* \in H$ *be an equilibrium for* (5.8)*, i.e., we have* $y_* = \mathcal{F}(y_*, \mu(y_*))$*. Then,* y_* *is said to be* locally asymptotically stable *if there exists a constant* $\eta > 0$ *and a function* $\beta \in \mathcal{KL}$ *such that the inequality*

$$\|y_{[\mu(\cdot),t_\circ,y_\circ]}(t) - y_*\|_H \leq \beta\big(\|y_\circ - y_*\|_H, t\big) \tag{5.10}$$

holds for all $y_\circ \in H$ *satisfying* $\|y_\circ - y_*\|_H < \eta$ *and all* $t \geq t_\circ$*.*

Let us recall the main result about asymptotic stability via DPP; see [51].

Proposition 5.2 *Let* $N \in \mathbb{N}$ *be chosen and the feedback mapping* μ^N *be computed by Algorithm* 9*. Assume that there exists an* $\alpha^N \in (0, 1]$ *such that for all* $(t_\circ, y_\circ) \in \mathbb{R}_0^+ \times H$ *the* relaxed dynamic programming principle

$$v^N(t_\circ, y_\circ) \geq v^N\big(t_\circ + \Delta t, y_{[\mu^N(y_\circ),t_\circ,y_\circ]}(t_\circ + \Delta t)\big) + \alpha^N L\big(y_\circ, \mu^N(y_\circ)\big)\big) \tag{5.11}$$

holds. Furthermore, we have for all $(t_\circ, y_\circ) \in \mathbb{R}_0^+ \times H$*:*

$$\alpha^N v(t_\circ, y_\circ) \leq \alpha^N J^N_{y_\circ,t_\circ}(\mu^N(y_\circ)) \leq v^N(t_\circ, y_\circ) \leq v(t_\circ, y_\circ), \tag{5.12}$$

where $y_{[\mu^N(\cdot),t_\circ,y_\circ]}$ *solves the closed-loop dynamics* (5.8) *with* $\mu = \mu^N$*. If, in addition, there exists an equilibrium* $y_* \in H$ *and* $\alpha_1, \alpha_2 \in \mathcal{K}_\infty$ *satisfying*

$$L_*(y_\circ) = \min_{u \in \mathcal{U}_{ad}} L(y_\circ, u) \geq \alpha_1\big(\|y_\circ - y_*\|_H\big), \tag{5.13a}$$

$$\alpha_2\big(\|y_\circ - y_*\|_H\big) \geq v^N(t_\circ, y_\circ) \tag{5.13b}$$

hold for all $(t_\circ, y_\circ) \in \mathbb{R}_0^+ \times H$*, then* y_* *is a globally* asymptotically stable equilibrium *for* (5.8) *with the feedback map* $\mu = \mu^N$ *and value function* v^N*.*

Remark 5.1 1) Our running cost L defined in (5.7) satisfies condition (5.13a) for the choice $\widehat{y} = y_*$. Further, (5.13b) follows from the finite horizon quadratic cost functional $\hat{J}^N$, the definition of the value function v^N and our a-priori analysis presented in Lemma 5.1 below. Therefore, we only have to check the relaxed DPP (5.11).

2) It is proved in [51] that $\lim_{N\to\infty} \alpha^N = 1$. Hence, we would like to find α^N close to one to have the best approximation of v in terms of v^N. On the other hand, a large N implies that the numerical solution of (5.9) is much more involved.

In order to estimate α^N in the relaxed DPP we require the exponential controllability property for the system.

Definition 5.3 *System* (5.5) *is called* exponentially controllable *with respect to the running cost* L *if for each* $(t_\circ, y_\circ) \in \mathbb{R}_0^+ \times H$ *there exists two real constants* $C > 0$*,* $\sigma \in [0, 1)$ *and an admissible control* $u \in \mathbb{U}_{ad}(t_\circ)$ *such that:*

$$L(y_{[u,t_\circ,y_\circ]}(t), u(t)) \leq C\sigma^{t-t_\circ} L_*(y_\circ) \quad \text{a.e. } t \geq t_\circ. \tag{5.14}$$

Lemma 5.1 *Let $(t_\circ, y_\circ) \in \mathbb{R}_0^+ \times H$ and $u = 0$. Then, the solution to (5.5) satisfies the a-priori estimate*

$$\|y(t)\|_H \leq e^{-\gamma(t-t_\circ)} \|y_\circ\|_H \quad \text{a.e. } t \geq t_\circ \tag{5.15}$$

with $\gamma = \gamma(\theta, \rho) = \theta/C_V - \rho$.

Proof. Recall that V is continuously (even compactly) embedded into H. Due to the Poincaré inequality [34] there exists a constant $C_V > 0$ such that

$$\|\varphi\|_H \leq C_V \|\varphi\|_V \quad \text{for all } \varphi \in V. \tag{5.16}$$

Using (5.16), choosing $u(t) = 0$ and $\varphi = y(t)$ in (5.2) we obtain

$$\frac{\mathrm{d}}{\mathrm{d}t} \|y(t)\|_H^2 + \frac{2\theta}{C_V} \|y(t)\|_H^2 \leq 2\rho \|y(t)\|_H^2 \quad \text{a.e. } t \geq t_\circ$$

which implies

$$\frac{\mathrm{d}}{\mathrm{d}t} \|y(t)\|_H^2 \leq 2\left(\rho - \frac{\theta}{C_V}\right) \|y(t)\|_H^2 \quad \text{a.e. } t \geq t_\circ.$$

Thus, by Gronwall's inequality we derive (5.15) with $\gamma = \theta/C_V - \rho$.

Remark 5.2 For $\theta > \rho C_V$ we have $\gamma > 0$. Then, (5.15) implies that $\|y(t)\|_H < \|y_\circ\|_H$ for any $t > t_\circ$. Moreover, it is easy to check that the origin $y_\circ = 0$ is unstable for $\gamma < 0$.

Let us choose $\widehat{y} = 0$. Suppose that we have a particular class of state feedback controls of the form $u(x,t) = -Ky(x,t)$ with a positive constant K; see [10]. This assumption helps us to derive the exponential controllability in terms of the running cost L and to compute a minimal finite time prediction horizon $N\Delta t$ ensuring asymptotic stability. In this case, (5.15) has to be modified. Using similar arguments as in the proof of Lemma 5.1 we find for a given $K > 0$ that the state $y = y_{[-Ky, t_\circ, y_\circ]}$ satisfies

$$\|y(t)\|_H \leq e^{-\gamma(K)(t-t_\circ)} \|y_\circ\|_H \quad \text{a.e. } t \geq t_\circ \tag{5.17}$$

with $\gamma(K) = \theta/C_V + K - \rho$. Thus, if $K > \rho - \theta/C_V$ holds, $\|y(t)\|_H$ tends to zero for $t \to \infty$. Combining (5.17) with the desired exponential controllability (5.14) and using $\widehat{y} = 0$ we obtain for all $t \geq t_\circ$ (see [10]):

$$\begin{aligned} L(y(t), u(t)) &= \frac{1}{2}\left(\|y(t)\|_H^2 + \lambda \|u(t)\|_H^2\right) = \frac{1}{2}\left(1 + \lambda K^2\right) \|y(t)\|_H^2 \\ &\leq \frac{1}{2} C(K) e^{-2\gamma(K)(t-t_\circ)} \|y_\circ\|_H^2 = C(K)\sigma(K)^{t-t_\circ} L_*(y_\circ) \end{aligned} \tag{5.18}$$

a.e. $t \geq t_\circ$ and for every $(t_\circ, y_\circ) \in \mathbb{R}_0^+ \times H$, where

$$C(K) = (1 + \lambda K^2), \quad \sigma(K) = e^{-2\gamma(K)}, \quad \gamma(K) = \theta/C_V + K - \rho. \tag{5.19}$$

In the following theorem we provide an explicit formula for the scalar α^N in (5.11). A complete discussion is given in [51].

K	$\underline{y}_\circ < 0$	$\underline{y}_\circ > 0$
$\overline{y}_\circ < 0$	$K < u_b/\lvert\overline{y}_\circ\rvert$	no constraints
$\overline{y}_\circ > 0$	$K < \min\left\{u_a/\overline{y}_\circ, u_b/\lvert\underline{y}_\circ\rvert\right\}$	$K < \lvert u_a\rvert/\overline{y}_\circ$

Table 5.0.: Constraints for the feedback factor K in $u(x,t) = -Ky(x,t)$ considering the bilateral control constraints (5.21) and the initial condition (5.22).

Theorem 5.1 *Assume that the system* (5.5) *and* L *statisfy the controllability condition* (5.14)*. Let the finite prediction horizon* $N\Delta t$ *be given with* $N \in \mathbb{N}$ *and* $\Delta t > 0$*. Then the parameter* α^N *depends on* K *and it is given by the explicit formula:*

$$\alpha^N = 1 - \frac{(\eta_N - 1)\prod_{i=2}^{N}(\eta_i - 1)}{\prod_{i=2}^{N}\eta_i - \prod_{i=2}^{N}(\eta_i - 1)} \tag{5.20}$$

where $\eta_i(K) = C(1-\sigma^i)/(1-\sigma)$ *and the constants* $C = C(K)$*,* $\sigma = \sigma(K)$ *are given by Definition* 5.3*.*

Remark 5.3 Theorem 5.1 suggests how we can compute the minimal horizon N which ensures asympotic stability. Due to we maximize

$$1 - \frac{(\eta_N(K) - 1)\prod_{i=2}^{N}(\eta_i(K) - 1)}{\prod_{i=2}^{N}\eta_i(K) - \prod_{i=2}^{N}(\eta_i(K) - 1)}, \eta_i(K) = (1 + \lambda K^2)\,\frac{1 - e^{-2i(\theta/C_V + K - \rho)}}{1 - e^{-2(\theta/C_V + K - \rho)}}$$

with respect to $K > \max(0, \rho - \theta/C_V)$ and $N \in \mathbb{N}$ in order to get $\alpha^N > 0$. Further, we suppose that $u \in \mathbb{U}^N(t_\circ)$ holds. Hence, we have to guarantee the bilateral control constraints

$$u_a \leq -Ky(x,t) \leq u_b \quad \text{a.e. } (x,t) \in Q^N, \tag{5.21}$$

where $u_a \leq 0 \leq u_b$ holds. Under these assumptions, the computation of K and N has to take into account the influence of the control constraints. Since we determine K in such a way that $\gamma(K) = \theta/C_V + K - \rho > 0$ is satisfied, we derive from (5.17) that

$$\|y(t)\|_H \leq \|y_\circ\|_H \quad \text{a.e. } t \geq t_\circ.$$

Let us suppose that we have a bounded initial condition such that $y_\circ \neq 0$ and $\|y(t)\|_{C(\overline{\Omega})} \leq \|y_\circ\|_{C(\overline{\Omega})}$ a.e. $t \geq t_\circ$. Then, we define

$$\underline{y}_\circ = \min_{x \in \overline{\Omega}} y_\circ(x), \quad \overline{y}_\circ = \max_{x \in \overline{\Omega}} y_\circ(x). \tag{5.22}$$

Then, K has to satisfy $K > \max(0, \rho - \theta/C_V)$ and the restrictions shown in Table 5.0. Summarizing, K has always an upper bound due to the constraints u_a, u_b and a lower bound due to the stabilization related to $\gamma(K) > 0$.

5.3. The finite horizon problem

In this section we discuss (5.9), which has to be solved at each level of Algorithm 9.

5.3.1. The open loop problem

Recall that we have introduced the final time $t_\circ^N = t_\circ + N\Delta t$ and the control space $\mathbb{U}^N(t_\circ) = L^2(t_\circ, t_\circ^N; H)$. The space $\mathbb{Y}^N(t_\circ) = W(t_\circ, t_\circ^N)$ is given by

$$W(t_\circ, t_\circ^N) = \big\{\varphi \in L^2(t_\circ, t_\circ^N; V) \,\big|\, \varphi_t \in L^2(t_\circ, t_\circ^N; V')\big\},$$

which is a Hilbert space endowed with the common inner product [32, pp. 472-479]. We define the Hilbert space $\mathbb{X}^N(t_\circ) = \mathbb{Y}^N(t_\circ) \times \mathbb{U}^N(t_\circ)$ endowed with the standard product topology. Moreover, we introduce the Hilbert space $\mathbb{Z}^N(t_\circ) = \mathbb{Z}_1^N(t_\circ) \times H$ with $\mathbb{Z}_1^N(t_\circ) = L^2(t_\circ, t_\circ^N; V)$ and the nonlinear operator $e = (e_1, e_2) : \mathbb{X}^N(t_\circ) \to \mathbb{Z}^N(t_\circ)'$ by

$$\begin{aligned}
\langle e_1(x), \varphi\rangle_{\mathbb{Z}_1^N(t_\circ)', \mathbb{Z}_1^N(t_\circ)} &= \int_{t_\circ}^{t_\circ^N} \langle y_t(t), \varphi(t)\rangle_{V',V} \,\mathrm{d}t \\
&\quad + \int_{t_\circ}^{t_\circ^N} \int_\Omega \theta y_x(t)\varphi(x) + \Big(y_x(t) + \rho\big(y(t)^3 - y(t)\big) - u(t)\Big)\varphi(t) \,\mathrm{d}x\mathrm{d}t, \\
\langle e_2(x), \phi\rangle_H &= \langle y(t_\circ) - y_\circ, \phi\rangle_H
\end{aligned}$$

for $x = (y, u) \in \mathbb{X}^N(t_\circ)$, $(\varphi, \phi) \in \mathbb{Z}^N(t_\circ)$, where we identify the dual $\mathbb{Z}^N(t_\circ)'$ of $\mathbb{Z}^N(t_\circ)$ with $L^2(t_\circ, t_\circ^N; V') \times H$ and $\langle\cdot,\cdot\rangle_{\mathbb{Z}_1^N(t_\circ)', \mathbb{Z}_1^N(t_\circ)}$ denotes the dual pairing between $\mathbb{Z}_1^N(t_\circ)'$ and $\mathbb{Z}_1^N(t_\circ)$. Then, for given $u \in \mathbb{U}^N(t_\circ)$ the weak formulation for (5.2) can be expressed as the operator equation $e(x) = 0$ in $\mathbb{Z}^N(t_\circ)'$. Further, we can write (5.9) as a constrained infinite dimensional minimization problem

$$\min J(x) = \int_{t_\circ}^{t_\circ^N} L(y(t), u(t)) \,\mathrm{d}t \quad \text{s.t.} \quad x \in \mathbb{F}_{ad}^N(t_\circ) \tag{5.23}$$

with the feasible set

$$\mathbb{F}_{ad}^N(t_\circ) = \big\{x = (y, u) \in \mathbb{X}^N(t_\circ) \,\big|\, e(x) = 0 \text{ in } \mathbb{Z}^N(t_\circ)' \text{ and } u \in \mathbb{U}_{ad}^N(t_\circ)\big\}.$$

For given fixed control $u \in \mathbb{U}_{ad}^N(t_\circ)$ we consider the state equation $e(y, u) = 0 \in \mathbb{Z}^N(t_\circ)'$, i.e., y satisfies

$$\begin{aligned}
&\frac{\mathrm{d}}{\mathrm{d}t}\langle y(t), \varphi\rangle_H + \int_\Omega \theta y_x(t)\varphi' + \big(y_x(t) + \rho(y(t)^3 - y(t))\big)\varphi \,\mathrm{d}x \\
&\qquad\qquad = \int_\Omega u(t)\varphi \,\mathrm{d}x \quad \text{f.f.a. } t \in (t_\circ, t_\circ^N], \\
&\langle y(t_\circ), \varphi\rangle_H = \langle y_\circ, \varphi\rangle_H
\end{aligned} \tag{5.24}$$

for all $\varphi \in V$. The following result is proved in [96, Theorem 5.5].

Proposition 5.3 *For given $(t_\circ, y_\circ) \in \mathbb{R}_0^+ \times H$ and $u \in \mathbb{U}_{ad}^N(t_\circ)$ there exists a unique weak solution $y \in \mathbb{Y}^N(t_\circ)$ to (5.24) for every $(\theta, \rho) \in D_{ad}$. If, in addition, $y_\circ$ is essentially bounded in Ω, i.e., $y_\circ \in L^\infty(\Omega)$ holds, we have $y \in L^\infty(Q^N)$ satisfying*

$$\|y\|_{\mathbb{Y}^N(t_\circ)} + \|y\|_{L^\infty(Q^N)} \le C\big(\|u\|_{\mathbb{U}^N(t_\circ)} + \|y_\circ\|_{L^\infty(\Omega)}\big) \tag{5.25}$$

for a $C > 0$, which is independent of u and $y_\circ$.

Adopting (5.25) it can be shown that (5.23) possesses at least one (local) optimal solution which we denote by $\bar{x}^N = (\bar{y}^N, \bar{u}^N) \in \mathbb{F}_{ad}^N(t_\circ)$; see [96, Chapter 5]. For the numerical computation of $\bar{x}^N$ we turn to first-order necessary optimality conditions for (5.23). To ensure the existence of a unique Lagrange multiplier we investigate the surjectivity of the linearization $e'(\bar{x}^N) : \mathbb{X}^N(t_\circ) \to \mathbb{Z}^N(t_\circ)'$ of the operator e at a given point $\bar{x}^N = (\bar{y}^N, \bar{u}^N) \in \mathbb{X}^N(t_\circ)$. Note that the Fréchet derivative $e'(\bar{x}^N) = (e_1'(\bar{x}^N), e_2'(\bar{x}^N))$ of e at $\bar{x}^N$ is given by

$$\begin{aligned}
\langle e_1'(\bar{x}^N)x, \varphi\rangle_{\mathbb{Z}_1^N(t_\circ)',\mathbb{Z}_1^N(t_\circ)} &= \int_{t_\circ}^{t_\circ^N} \langle y_t(t), \varphi(t)\rangle_{V',V}\,\mathrm{d}t \\
&\quad + \int_{t_\circ}^{t_\circ^N}\int_\Omega \theta y_x(t)\varphi(x) + \Big(y_x(t) + \rho\big(3\bar{y}^N(t)^2 - 1\big)y(t) - u(t)\Big)\varphi(t)\,\mathrm{d}x\mathrm{d}t, \\
\langle e_2'(\bar{x}^N)x, \phi\rangle_H &= \langle y(t_\circ), \phi\rangle_H
\end{aligned}$$

for $x = (y, u) \in \mathbb{X}^N(t_\circ)$, $(\varphi, \phi) \in \mathbb{Z}^N(t_\circ)$. Now, the operator $e'(\bar{x}^N)$ is surjective if and only if for an arbitrary $F = (F_1, F_2) \in \mathbb{Z}^N(t_\circ)'$ there exists a pair $x = (y, u) \in \mathbb{X}^N(t_\circ)$ satisfying $e'(\bar{x}^N) = F$ in $\mathbb{Z}^N(t_\circ)'$ which is equivalent with the fact that there exists an $u \in \mathbb{U}^N(t_\circ)$ and an $y \in \mathbb{Y}^N(t_\circ)$ solving the linear parabolic problem

$$y_t - \theta y_{xx} + y_x + \rho(3\bar{y}^2 - 1)y = F_1 \text{ in } \mathbb{Z}_1^N(t_\circ)', \quad y(t_\circ) = F_2 \text{ in } H. \tag{5.26}$$

Using standard arguments [32] it follows that there exists for any $u \in \mathbb{U}^N(t_\circ)$ a unique $y \in \mathbb{Y}^N(t_\circ)$ solving (5.26). Thus, $e'(\bar{x}^N)$ is a surjective operator and the local solution $\bar{x}^N$ to (5.23) can be characterized by first-order optimality conditions. We introduce the Lagrangian by

$$L(x, p, p_\circ) = J(x) + \langle e(x), (p, p_\circ)\rangle_{\mathbb{Z}^N(t_\circ)',\mathbb{Z}^N(t_\circ)}$$

for $x \in \mathbb{X}^N(t_\circ)$ and $(p, p_\circ) \in \mathbb{Z}^N(t_\circ)$. Then, there exists a unique associated Lagrange multiplier pair $(\bar{p}^N, \bar{p}_\circ)$ to (5.23) satisfying the optimality system

$$\begin{aligned}
&\nabla_y L(\bar{x}^N, \bar{p}^N, \bar{p}_\circ^N)y = 0 && \forall y \in \mathbb{Y}^N(t_\circ) && \textit{(adjoint equation)} \\
&\nabla_u L(\bar{x}^N, \bar{p}^N, \bar{p}_\circ^N)(u - \bar{u}^N) \ge 0 && \forall u \in \mathbb{U}_{ad}^N(t_\circ) && \textit{(variational inequality)}, \\
&\langle e(\bar{x}^N), (p, p_\circ)\rangle_{\mathbb{Z}^N(t_\circ)',\mathbb{Z}^N(t_\circ)} = 0 && \forall (\bar{p}, \bar{p}_0) \in \mathbb{Z}^N(t_\circ) && \textit{(state equation)}.
\end{aligned}$$

It follows from variational arguments that the strong formulation for the adjoint equation is of the form

$$\begin{aligned} -\bar{p}_t^N - \theta\bar{p}_{xx}^N - \bar{p}_x^N - \rho\big(1-3(\bar{y}^N)^2\big)\bar{p}^N &= \widehat{y} - \bar{y}^N && \text{in } Q^N, \\ \bar{p}^N(0,\cdot) = \bar{p}^N(1,\cdot) &= 0 && \text{in } (t_\circ, t_\circ^N), \\ \bar{p}^N(t_\circ^N) &= 0 && \text{in } \Omega. \end{aligned} \tag{5.27}$$

Moreover, we have $\bar{p}_\circ^N = \bar{p}^N(t_\circ)$. The variational inequality has the form

$$\int_{t_\circ}^{t_\circ^N} \int_\Omega (\lambda\bar{u}^N - \bar{p}^N)(u - \bar{u}^N)\,\mathrm{d}x\mathrm{d}t \geq 0 \quad \text{for all } u \in \mathbb{U}_{ad}^N(t_\circ). \tag{5.28}$$

Using the techniques as in [99, Proposition 2.12] one can proof that second-order sufficient optimality conditions can be ensured provided the residuum $\|\bar{y}^N - \widehat{y}\|_{L^2(t_\circ,t_\circ^N;H)}$ is sufficiently small.

5.3.2. POD reduced order model for open-loop problem

To solve (5.23) we apply a reduced-order discretization based on proper orthogonal decomposition (POD); see [53]. In this subsection we briefly recall the POD method, present an a-priori error estimate for the POD solution to the state equation $e(x) = 0 \in \mathbb{Z}^N(t_\circ)'$ and formulate the POD Galerkin approach for (5.23).

The POD method for dynamical systems

By X we denote either the function space H or V. Then, for $\wp \in \mathbb{N}$ let the so-called *snapshots* or *trajectories* $y^k(t) \in X$ are given a.e. $t \in [t_\circ, t_\circ^N]$ and for $1 \leq k \leq \wp$. At least one of the trajectories y^k is assumed to be nonzero. Then we introduce the linear subspace

$$\mathcal{V} = \operatorname{span}\Big\{ y^k(t) \,|\, t \in [t_\circ, t_\circ^N] \text{ a.e. and } 1 \leq k \leq \wp \Big\} \subset X \tag{5.29}$$

with dimension $d \geq 1$. We call the set $\mathcal{V}$ *snapshot subspace*. The method of POD consists in choosing a complete orthonormal basis in X such that for every $\ell \leq d$ the mean square error between $y^k(t)$ and their corresponding ℓ-th partial Fourier sum is minimized on average:

$$\begin{cases} \min \displaystyle\sum_{k=1}^{\wp} \int_{t_\circ}^{t_\circ^N} \Big\| y^k(t) - \sum_{i=1}^{\ell} \langle y^k(t), \psi_i \rangle_X \, \psi_i \Big\|_X^2 \,\mathrm{d}t \\ \text{s.t. } \{\psi_i\}_{i=1}^{\ell} \subset X \text{ and } \langle \psi_i, \psi_j \rangle_X = \delta_{ij},\ 1 \leq i,j \leq \ell, \end{cases} \tag{5.30}$$

where the symbol δ_{ij} denotes the Kronecker symbol satisfying $\delta_{ii} = 1$ and $\delta_{ij} = 0$ for $i \neq j$. An optimal solution $\{\bar{\psi}_i\}_{i=1}^{\ell}$ to (5.30) is called a *POD basis of rank* ℓ. The solution to (5.30) is given by the next theorem. For its proof we refer the reader to [53, Theorem 2.13].

Theorem 5.2 *Let X be a separable real Hilbert space and $y_1^k, \ldots, y_n^k \in X$ are given snapshots for $1 \le k \le \wp$. Define the linear operator $\mathcal{R} : X \to X$ as follows:*

$$\mathcal{R}\psi = \sum_{k=1}^{\wp} \int_{t_\circ}^{t_\circ^N} \langle \psi, y^k(t) \rangle_X \, y^k(t) \, \mathrm{d}t \quad \text{for } \psi \in X. \tag{5.31}$$

Then, $\mathcal{R}$ is a compact, nonnegative and symmetric operator. Suppose that $\{\bar{\lambda}_i\}_{i\in\mathbb{N}}$ and $\{\bar{\psi}_i\}_{i\in\mathbb{N}}$ denote the nonnegative eigenvalues and associated orthonormal eigenfunctions of $\mathcal{R}$ satisfying

$$\mathcal{R}\bar{\psi}_i = \bar{\lambda}_i \bar{\psi}_i, \quad \bar{\lambda}_1 \ge \ldots \ge \bar{\lambda}_d > \bar{\lambda}_{d+1} = \ldots = 0, \quad \bar{\lambda}_i \to 0 \text{ as } i \to \infty. \tag{5.32}$$

Then, for every $\ell \le d$ the first ℓ eigenfunctions $\{\bar{\psi}_i\}_{i=1}^{\ell}$ solve (5.30). Moreover, the value of the cost evaluated at the optimal solution $\{\bar{\psi}_i\}_{i=1}^{\ell}$ satisfies

$$\mathcal{E}(\ell) = \sum_{k=1}^{\wp} \int_{t_\circ}^{t_\circ^N} \Big\| y^k(t) - \sum_{i=1}^{\ell} \langle y^k(t), \bar{\psi}_i \rangle_X \, \bar{\psi}_i \Big\|_X^2 \, \mathrm{d}t = \sum_{i=\ell+1}^{d} \bar{\lambda}_i. \tag{5.33}$$

The Galerkin POD scheme for the state equation

Suppose that $(t_\circ, y_\circ) \in \mathbb{R}_0^+ \times H$ and $t_\circ^N = t_\circ + N\Delta t$ with prediction horizon $N\Delta t > 0$. For given fixed control $u \in \mathbb{U}_{ad}^N(t_\circ)$ we consider the state equation $e(y,u) = 0 \in \mathbb{Z}^N(t_\circ)'$, i.e., y satisfies (5.24). Let us turn to a POD discretization of (5.24). To keep the notation simple we apply only a spatial discretization with POD basis functions, but no time integration by, e.g., the implicit Euler method. Therefore, we use the continuous version of the POD method introduced in Section 5.3.2. In this section we distinguish two choices for X: $X = H$ and $X = V$. We choose the snapshots $y^1 = y$ and $y^2 = y_t$, i.e., we set $\wp = 2$. By Proposition 5.3 the snapshots y^k, $k = 1, \ldots, \wp$, belong to $L^2(0,T;V)$. In fact, due to Riesz Representation theorem, there exists a linear isomorphism from V' to V such that we can assert $y_t \in L^2(0,T,V)$. According to (5.32) let us introduce the following notations:

$$\mathcal{R}_V\psi = \sum_{k=1}^{\wp} \int_{t_\circ}^{t_\circ^N} \langle \psi, y^k(t) \rangle_V \, y^k(t) \, \mathrm{d}t \qquad \text{for } \psi \in V,$$

$$\mathcal{R}_H\psi = \sum_{k=1}^{\wp} \int_{t_\circ}^{t_\circ^N} \langle \psi, y^k(t) \rangle_H \, y^k(t) \, \mathrm{d}t \qquad \text{for } \psi \in H.$$

To distinguish the two choices for the Hilbert space X we denote by the sequence $\{(\lambda_i^V, \psi_i^V)\}_{i\in\mathbb{N}} \subset \mathbb{R}_0^+ \times V$ the eigenvalue value decomposition for $X = V$, i.e., we have

$$\mathcal{R}_V\psi_i^V = \lambda_i^V \psi_i^V \quad \text{for all } i \in \mathbb{N}.$$

Furthermore, let $\{(\lambda_i^H, \psi_i^H)\}_{i\in\mathbb{N}} \subset \mathbb{R}_0^+ \times H$ in satisfy

$$\mathcal{R}_H \psi_i^H = \lambda_i^H \psi_i^H \quad \text{for all } i \in \mathbb{N}.$$

Then, $d = \dim \mathcal{R}_V(V) = \dim \mathcal{R}_H(H) \leq \infty$; see [94]. The next result – also taken from [94] – ensures that the POD basis $\{\psi_i^H\}_{i=1}^{\ell}$ of rank ℓ build a subset of the test space V.

Lemma 5.2 *Suppose that the snapshots $y^k \in L^2(0,T;V)$, $k = 1, \ldots, \wp$. Then, we have $\psi_i^H \in V$ for $i = 1, \ldots, d$.*

Let us define the two POD subspaces

$$V^\ell = \operatorname{span}\left\{\psi_1^V, \ldots, \psi_\ell^V\right\} \subset V, \quad H^\ell = \operatorname{span}\left\{\psi_1^H, \ldots, \psi_\ell^H\right\} \subset V \subset H,$$

where $H^\ell \subset V$ follows from Lemma 5.2. Moreover, we introduce the orthogonal projection operators $\mathcal{P}_H^\ell : V \to H^\ell \subset V$ and $\mathcal{P}^\ell : V \to V^\ell \subset V$ as follows:

$$\begin{aligned} v^\ell = \mathcal{P}_H^\ell \varphi \text{ for any } \varphi \in V \quad &\text{iff } v^\ell \text{ solves } \min_{w^\ell \in H^\ell} \|\varphi - w^\ell\|_V, \\ v^\ell = \mathcal{P}_V^\ell \varphi \text{ for any } \varphi \in V \quad &\text{iff } v^\ell \text{ solves } \min_{w^\ell \in V^\ell} \|\varphi - w^\ell\|_V. \end{aligned} \tag{5.34}$$

It follows from the first-order optimality conditions for (5.34) that $v^\ell = \mathcal{P}_H^\ell \varphi$ satisfies

$$\langle v^\ell, \psi_i^H \rangle_V = \langle \varphi, \psi_i^H \rangle_V, \quad 1 \leq i \leq \ell. \tag{5.35}$$

Writing $v^\ell \in H^\ell$ in the form $v^\ell = \sum_{j=1}^{\ell} \mathrm{v}_j^\ell \psi_j^H$ we derive from (5.35) that the vector $\mathrm{v}^\ell = (\mathrm{v}_1^\ell, \ldots, \mathrm{v}_\ell^\ell)^\top \in \mathbb{R}^\ell$ satisfies the linear system

$$\sum_{j=1}^{\ell} \langle \psi_j^H, \psi_i^H \rangle_V \, \mathrm{v}_j^\ell = \langle \varphi, \psi_i^H \rangle_V, \quad 1 \leq i \leq \ell.$$

For the operator $\mathcal{P}_V^\ell$ we have the explicit representation

$$\mathcal{P}_V^\ell \varphi = \sum_{i=1}^{\ell} \langle \varphi, \psi_i^V \rangle_V \, \psi_i^V \text{ for } \varphi \in V.$$

Moreover, we introduce the orthogonal projection operator $\mathcal{P}^\ell : V \to V^\ell$ by

$$\mathcal{P}_V^\ell \varphi = \sum_{i=1}^{\ell} \langle \varphi, \psi_i^V \rangle_V \, \psi_i^V \quad \text{for } \varphi \in V. \tag{5.36}$$

Further, we conclude from (5.33) that

$$\sum_{k=1}^{\wp} \int_0^T \|y^k(t) - \mathcal{P}_V^\ell y^k(t)\|_V^2 \, \mathrm{d}t = \sum_{i=\ell+1}^{d} \lambda_i^V. \tag{5.37}$$

The projection operator of $\mathcal{P}_V^\ell y_t$ is well-defined, since $y_t \in L^2(0,T,V)$ thanks to Riesz's theorem.
Next we review an essential result from [94, Theorem 6.2], which is essential in our a-priori error analysis for the choice $X = H$. Recall that $H^\ell \subset V$ holds. Consequently, $\|\psi_i^H - \mathcal{P}_H^\ell \psi_i^H\|_V$ is well-defined for $1 \le i \le \ell$.

Theorem 5.3 *Suppose that $y^k \in L^2(0,T;V)$ for $1 \le k \le \wp$. Then,*

$$\sum_{k=1}^{\wp} \int_0^T \|y^k(t) - \mathcal{P}_H^\ell y^k(t)\|_V^2 \,\mathrm{d}t = \sum_{i=\ell+1}^{d} \lambda_i^H \, \|\psi_i^H - \mathcal{P}_H^\ell \psi_i^H\|_V^2.$$

Moreover, $\mathcal{P}_H^\ell y^k$ converges to y^k in $L^2(0,T;V)$ as ℓ tends to ∞ for each $k \in \{1,\dots,\wp\}$.

Let us define the linear space $X^\ell \subset V$ as

$$X^\ell = \operatorname{span}\{\psi_1, \dots, \psi_\ell\},$$

where $\psi_i = \psi_i^V$ in case of $X = V$ and $\psi_i = \psi_i^H$ in case of $X = H$. Hence, $X^\ell = V^\ell$ and $X^\ell = H^\ell$ for $X = V$ and $X = H$, respectively. Now, a POD Galerkin scheme for (5.24) is given as follows: find $y^\ell(t) \in X^\ell$ a.e. $t \in [t_\circ, t_\circ^N]$ satisfying

$$\begin{aligned} &\frac{\mathrm{d}}{\mathrm{d}t} \langle y^\ell(t), \psi \rangle_H + \int_\Omega \theta y_x^\ell(t)\psi' + \big(y_x^\ell(t) + \rho(y^\ell(t)^3 - y^\ell(t))\big)\psi \,\mathrm{d}x \\ &\qquad\qquad = \int_\Omega u(t)\psi \,\mathrm{d}x \quad \text{f.f.a. } t \in (t_\circ, t_\circ^N], \\ &\langle y^\ell(t_\circ), \psi \rangle_H = \langle y_\circ, \psi \rangle_H \end{aligned} \tag{5.38}$$

for all $\psi \in X^\ell$. It follows by similar arguments as in the proof of Proposition 5.3 that there exists a unique solution to (5.38). If $y_\circ \in L^\infty(Q^N)$ holds, y^ℓ satisfies the a-priori estimate

$$\|y^\ell\|_{\mathbb{Y}^N(t_\circ)} + \|y^\ell\|_{L^\infty(Q^N)} \le C\big(\|y_\circ\|_{L^\infty(\Omega)} + \|u\|_{\mathbb{U}^N(t_\circ)}\big). \tag{5.39}$$

where the constant $C > 0$ is independent of ℓ and $y_\circ$. Let $\mathcal{P}^\ell$ denote $\mathcal{P}_V^\ell$ in case of $X = V$ and $\mathcal{P}_H^\ell$ in case of $X = H$. To derive an error estimate for $\|y - y^\ell\|_{\mathbb{Y}^N(t_\circ)}$ we make use of the decomposition

$$y(t) - y^\ell(t) = y(t) - \mathcal{P}^\ell y(t) + \mathcal{P}^\ell y(t) - y^\ell(t) = \varrho^\ell(t) + \vartheta^\ell(t) \text{ a.e. } t \in [t_\circ, t_\circ^N]$$

with $\varrho^\ell(t) = y(t) - \mathcal{P}^\ell y(t) \in (X^\ell)^\perp$ and $\vartheta^\ell(t) = \mathcal{P}^\ell y(t) - y^\ell(t) \in X^\ell$. From (5.37) and Theorem 5.3 it follows that

$$\begin{aligned}\|\varrho^\ell\|^2_{\mathbb{Y}^N(t_\circ)} &= \int_{t_\circ}^{t_\circ^N} \|y(t) - \mathcal{P}^\ell y(t)(t)\|_V^2 + \|y_t(t) - \mathcal{P}^\ell y_t(t)\|_V^2 \,\mathrm{d}t \\ &= \begin{cases} \displaystyle\sum_{i=\ell+1}^{d} \lambda_i^V & \text{for } X = V, \\ \displaystyle\sum_{i=\ell+1}^{d} \lambda_i^H \|\psi_i^H - \mathcal{P}_H^\ell \psi_i^H\|_V^2 & \text{for } X = H. \end{cases}\end{aligned} \tag{5.40}$$

Next we estimate $\vartheta^\ell(t)$. We infer from (5.24) and (5.38) that

$$\begin{aligned}&\langle \vartheta_t^\ell(t), \psi\rangle_{V',V} + \langle \theta\vartheta^\ell(t), \psi\rangle_V \\ &\quad = \langle \rho(y(t) - y^\ell(t)), \psi\rangle_H + \langle \rho(y(t)^3 - y^\ell(t)^3), \psi\rangle_H + \langle \mathcal{P}^\ell y_t(t) - y_t(t), \psi\rangle_{V',V}\end{aligned}$$

for all $\psi \in X^\ell$ and a.e. $t \in [t_\circ, t_\circ^N]$. For $s \in [0,1]$ we define the function $\xi^\ell(s) = y^\ell + s(y - y^\ell)$. Then it follows from (5.25) and (5.39) that

$$\|\xi^\ell(s)\|_{L^\infty(Q^N)} \le s\,\|y\|_{L^\infty(Q^N)} + (1-s)\,\|y^\ell\|_{L^\infty(Q^N)} \le C_1 \quad \text{for all } s \in [0,1]$$

with a constant $C_1 > 0$ dependent on $y_\circ$, u_a and u_b, but independent of y, y^ℓ and ℓ. By the mean value theorem we obtain

$$\begin{aligned}\langle y(t)^3 - y^\ell(t)^3, \psi\rangle_H &= \left\langle \frac{1}{4}\int_0^1 \xi^\ell(s;t)^4 \big(y(t) - y^\ell(t)\big)\,\mathrm{d}s, \psi \right\rangle_H \\ &\le C_2\,\|y(t) - y^\ell(t)\|_H \|\psi\|_H \quad \text{for all } \psi \in H\end{aligned}$$

with $C_2 = C_1^4/4$. We choose $\psi = \vartheta^\ell(t) \in X^\ell$ and set $C_3 = \rho(1 + C_2)$. Then, we derive from Poincaré's inequality (5.16) and Young's inequality

$$\begin{aligned}\frac{\mathrm{d}}{\mathrm{d}t}\|\vartheta^\ell(t)\|_H^2 + \theta_a\|\vartheta^\ell(t)\|_V^2 &\le 2C_3\left(C_V\,\|\varrho^\ell(t)\|_V \|\vartheta^\ell(t)\|_H + \|\vartheta^\ell(t)\|_H^2\right) \\ &\quad + \frac{1}{\theta_a}\|\mathcal{P}^\ell y_t(t) - y_t(t)\|_{V'}^2 \\ &\le C_4\left(\|\varrho^\ell(t)\|_V^2 + (\|\varrho_t^\ell(t)\|_{V'}^2 + \|\vartheta^\ell(t)\|_H^2\right),\end{aligned} \tag{5.41}$$

where we have put $C_4 = \max(C_3 C_V^2, 2C_3, 1/\theta_a)$. By Gronwall's inequality we have

$$\|\vartheta^\ell(t)\|_H^2 \le C_5 \cdot \begin{cases} \|\vartheta^\ell(t_\circ)\|_H^2 + \sum_{i=\ell+1}^{d} \lambda_i^V & \text{for } X = V, \\ \|\vartheta^\ell(t_\circ)\|_H^2 + \sum_{i=\ell+1}^{d} \lambda_i^H \|\psi_i^H - \mathcal{P}_H^\ell \psi_i^H\|_V^2 & \text{for } X = H \end{cases} \tag{5.42}$$

with $C_5 = e^{C_4(t_\circ^N - t_\circ)} \max(1, C_4)$. Furthermore, (5.41) implies that

$$\begin{aligned} \|\vartheta^\ell\|_{L^2(t_\circ,t_\circ^N;V)}^2 &\le \frac{1}{\theta_a} \|\vartheta^\ell(t_\circ)\|_H^2 + \frac{C_4}{\theta_a} \left(\|\varrho^\ell\|_{W(t_\circ,t_\circ^N)} + \|\vartheta^\ell\|_{L^2(t_\circ,t_\circ^N;H)}^2 \right) \\ &\le C_6 \cdot \begin{cases} \|\vartheta^\ell(t_\circ)\|_H^2 + \sum_{i=\ell+1}^{d} \lambda_i^V & \text{for } X = V, \\ \|\vartheta^\ell(t_\circ)\|_H^2 + \sum_{i=\ell+1}^{d} \lambda_i^H \|\psi_i^H - \mathcal{P}_H^\ell \psi_i^H\|_V^2 & \text{for } X = H \end{cases} \end{aligned} \tag{5.43}$$

with $C_6 = \max(1, C_4, C_4 C_5 (t_\circ^N - t_\circ))/\theta_a$. From estimates (5.42), (5.43), from

$$y(t)^3 - y^\ell(t) = \big(\varrho^\ell(t) + \vartheta^\ell(t)\big)\big(y(t)^2 + y(t)y^\ell(t) + y^\ell(t)^2\big) \quad \text{a.e. } t \in [t_\circ, t_\circ^N]$$

and from the embedding inequalities [34]

$$\begin{aligned} \|\varphi\|_{L^\infty(\Omega)} &\le C_\infty \|\varphi\|_V && \text{for all } \varphi \in V, \\ \|\varphi\|_{L^\infty(t_\circ,t_\circ^N;H)} &\le C_W \|\varphi\|_{W(t_\circ,t_\circ^N)} && \text{for all } \varphi \in W(t_\circ, t_\circ^N) \end{aligned}$$

for two constants C_∞, $C_W > 0$ we infer that

$$\begin{aligned} \|\vartheta^\ell\|_{L^2(t_\circ,t_\circ^N;V')} &= \sup_{\|\varphi\|_{L^2(t_\circ,t_\circ^N;V)}=1} \int_{t_\circ}^{t_\circ^N} \langle \vartheta^\ell(t), \varphi(t) \rangle_{V',V} \,\mathrm{d}t \\ &\le \sup_{\|\varphi\|_{L^2(t_\circ,t_\circ^N;V)}=1} \int_{t_\circ}^{t_\circ^N} \langle \rho(y(t) - y^\ell(t)), \varphi(t) \rangle_H + \langle \rho(y(t)^3 - y^\ell(t)^3), \varphi(t) \rangle_H \,\mathrm{d}t \\ &\quad + \sup_{\|\varphi\|_{L^2(t_\circ,t_\circ^N;V)}=1} \int_{t_\circ}^{t_\circ^N} \langle \mathcal{P}^\ell y_t(t) - y_t(t), \varphi(t) \rangle_{V',V} - \langle \theta \vartheta^\ell(t), \varphi(t) \rangle_V \,\mathrm{d}t \\ &\le \rho C_V \left(\|\varrho^\ell\|_{L^2(t_\circ,t_\circ^N;H)} + \|\vartheta^\ell\|_{L^2(t_\circ,t_\circ^N;H)} \right) + \theta \|\vartheta^\ell\|_{L^2(t_\circ,t_\circ^N;V)} \\ &\quad + C_7 \left(\|\varrho^\ell\|_{L^\infty(t_\circ,t_\circ^N;H)} + \|\vartheta^\ell\|_{L^\infty(t_\circ,t_\circ^N;H)} \right) + \|\varrho_t^\ell\|_{L^2(t_\circ,t_\circ^N;V')} \end{aligned}$$

where $C_7 > 0$ satisfies $C_\infty \|y^2 + yy^\ell + (y^\ell)^2\|_{L^2(t_\circ,t_\circ^N;H)} \le C_7$. Hence, there is a constant $C_8 > 0$ depending on θ, ρ, C_W, C_5, C_6, C_7 such that

$$\begin{aligned} &\|\vartheta^\ell\|^2_{L^2(t_\circ,t_f;V')} \\ &\quad \le C_8 \cdot \begin{cases} \|\vartheta^\ell(t_\circ)\|_H^2 + \sum_{i=\ell+1}^{d} \lambda_i^V & \text{for } X = V, \\ \|\vartheta^\ell(t_\circ)\|_H^2 + \sum_{i=\ell+1}^{d} \lambda_i^H \|\psi_i^H - \mathcal{P}_H^\ell \psi_i^H\|_V^2 & \text{for } X = H. \end{cases} \end{aligned} \tag{5.44}$$

Form (5.42), (5.43) and (5.44) we infer the next result, which motivates the use of a POD approximation for our state equation (5.24).

Theorem 5.4 *Suppose that $(t_\circ, y_\circ) \in \mathbb{R}_0^+ \times L^\infty(\Omega)$, $t_\circ^N = t_\circ + N\Delta$ with prediction horizon $N\Delta t > 0$. Further, let $u \in \mathbb{U}_{ad}^N(t_\circ)$ be a fixed control input. By y and y^ℓ we denote the unique solution to* (5.24) *and* (5.38)*, respectively, where the POD basis of rank ℓ is computed by choosing $\wp = 2$, $y^1 = y$ and $y^2 = y_t$. Then,*

$$\|y - y^\ell\|^2_{\mathbb{Y}^N(t_\circ)} \le C \cdot \begin{cases} \|\vartheta^\ell(t_\circ)\|_H^2 + \sum_{i=\ell+1}^{d} \lambda_i^V & \text{for } X = V, \\ \|\vartheta^\ell(t_\circ)\|_H^2 + \sum_{i=\ell+1}^{d} \lambda_i^H \|\psi_i^H - \mathcal{P}_H^\ell \psi_i^H\|_V^2 & \text{for } X = H \end{cases}$$

for a $C > 0$ which is independent of ℓ. In particular, $\lim_{\ell\to\infty} \|y - y^\ell\|_{\mathbb{Y}^N(t_\circ)} = 0$.

The Galerkin POD scheme for the optimality system

Suppose that we have computed a POD basis $\{\psi_i\}_{i=1}^\ell$ of rank ℓ by choosing $X = H$ or $X = V$. Suppose that for $u \in \mathbb{U}_{ad}^N(t_\circ)$ the function y^ℓ is the POD Galerkin solution to (5.38). Then the POD Galerkin scheme for the adjoint equation (5.27) is given as follows: find $p^\ell \in X^\ell = \operatorname{span}\{\psi_1, \dots, \psi_\ell\}$ a.e. $t \in [t_\circ, t_\circ^N]$ satisfying

$$\begin{aligned} &-\frac{\mathrm{d}}{\mathrm{d}t}\langle p^\ell(t), \psi\rangle_H + \int_\Omega \theta p_x^\ell(t)\psi' - \big(p_x^\ell(t) + \rho(1 - 3y^\ell(t)^2)\big)p^\ell(t)\psi \,\mathrm{d}x \\ &\qquad = \int_\Omega \big(\widehat{y} - y^\ell(t)\big)\psi \,\mathrm{d}x = 0 \quad \text{f.f.a. } t \in [t_\circ, t_\circ^N), \\ &\langle p^\ell(t_\circ^N), \psi\rangle_H \end{aligned} \tag{5.45}$$

for all $\psi \in X^\ell$. A-priori error estimates for the POD solution p^ℓ to (5.45) can be derived by variational arguments; compare [87] and [53, Theorem 4.15]. If p^ℓ is computed, we

can derive a POD approximation for the variational inequality (5.28):

$$\int_{t_\circ}^{t_\circ^N} \int_\Omega (\lambda u - p^\ell)(\tilde{u} - u)\,\mathrm{d}x\mathrm{d}t \geq 0 \quad \text{for all } \tilde{u} \in \mathbb{U}_{ad}^N(t_\circ). \tag{5.46}$$

Summarizing, a POD suboptimal solution $\bar{x}^{N,\ell} = (\bar{y}^{N,\ell}, \bar{u}^{N,\ell}) \in \mathbb{X}_{ad}^N(t_\circ)$ to (5.9) satisfies together with the associated Lagrange multiplier $\bar{p}^{N,\ell} \in \mathbb{Y}_1^N(t_\circ)$ the coupled system (5.38), (5.45) and (5.46). The POD approximation of the finite horizon quadratic cost functional (5.23) reads

$$\hat{J}^{N,\ell}(u; t_\circ, y_\circ) = \int_{t_\circ}^{t_\circ^N} L\big(y^\ell_{[u,t_\circ,y_\circ]}(t), u(t)\big)\,\mathrm{d}t,$$

where $y^\ell_{[u,t_\circ,y_\circ]}$ is the solution to (5.38). In Algorithm 10 we set up the POD discretization for Algorithm 9. Due to our POD reduced-order approach an optimal solution to (5.47)

Algorithm 10: (POD-NMPC algorithm)

Require: time step $\Delta t > 0$, finite control horizon $N \in \mathbb{N}$, weighting param. $\lambda > 0$, POD tolerance $\tau_{pod} > 0$.

1: Compute a POD basis $\{\psi_i\}_{i=1}^\ell$ satisfying (5.33) with $\mathcal{E}(\ell) \leq \tau_{pod}$.
2: **for** $n = 0, 1, 2, \ldots$ **do**
3: Measure the state $y(t_n) \in V$ of the system at $t_n = n\Delta t$.
4: Set $t_\circ = t_n = n\Delta t$, $y_\circ = y(t_n)$ and compute a global solution to

$$\min \hat{J}^{N,\ell}(u^\ell; t_\circ, y_\circ^\ell) \quad \text{s.t.} \quad u^\ell \in \mathbb{U}_{ad}^N(t_\circ). \tag{5.47}$$

We denote the obtained optimal control by $\bar{u}^{N,\ell}$.
5: Define the NMPC feedback value $\mu^{N,\ell}(y(t)) = \bar{u}^{N,\ell}(t)$ and use this control to compute the associated state $y = y_{[\mu^{N,\ell}(\cdot),t_\circ,y_\circ]}$ by solving (5.5) on $[t_\circ, t_\circ + \Delta t]$.
6: **end for**

can be computed much faster than the one to (5.9). In the next subsection we address the question, how the suboptimality of the control influences the asymptotic stability.

5.3.3. Asymptotic stability for the POD-MPC algorithm

In this subsection we present the main results of the chapter. We give sufficient conditions that Algorithm 10 gives a stabilizing feedback control.

As in Section 5.2.2 we choose $\widehat{y} = y_* = 0$. Let $y_{[\bar{u}^{N,\ell},t_\circ,y_\circ]}(t)$ denote the solution to (5.5) with the control law $u = \bar{u}^{N,\ell}$; compare step 5 of Algorithm 10. By y^ℓ we denote the solution to (5.38) with the admissible control $u = -Ky^\ell$. Then,

$$\|y^\ell(t)\|_H^2 \leq \sigma(K)^{t-t_\circ} \|y_\circ\|_H^2 \quad \text{a.e. } t \geq t_\circ \tag{5.48}$$

with the same constants $C(K)$ and $\sigma(K)$ as in (5.19). Note that 5.48 holds for the uncontrolled problem. Since $u = -Ky^\ell \in \mathbb{U}_{ad}^N(t_\circ)$ is an admissible control for (5.47) and $\bar{x}^{N,\ell} = (\bar{y}^{N,\ell}, \bar{u}^{N,\ell})$ is a global solution to (5.47), we derive from (5.48)

$$L\big(\bar{y}^{N,\ell}(t), \bar{u}^{N,\ell}(t)\big) \leq L\big(y^\ell(t), -Ky^\ell(t)\big) = \frac{C(K)}{2} \left\|y^\ell(t)\right\|_H^2. \tag{5.49}$$

Using the Cauchy-Schwarz inequality we get

$$\begin{aligned} &L\big(y_{[\bar{u}^{N,\ell},t_\circ,y_\circ]}(t), \bar{u}^{N,\ell}(t)\big) \\ &\quad\leq \frac{1}{2} \left\|y_{[\bar{u}^{N,\ell},t_\circ,y_\circ]}(t) - \bar{y}^{N,\ell}(t)\right\|_H^2 + L\big(\bar{y}^{N,\ell}(t), \bar{u}^{N,\ell}(t)\big) \\ &\qquad+ \left\|y_{[\bar{u}^{N,\ell},t_\circ,y_\circ]}(t) - \bar{y}^{N,\ell}(t)\right\|_H \left\|\bar{y}^{N,\ell}(t)\right\|_H. \end{aligned} \tag{5.50}$$

Further, it follows from $L(\bar{y}^{N,\ell}(t), \bar{u}^{N,\ell}(t)\big) \leq L\big(y^\ell, -Ky^\ell(t)\big)$, $\lambda > 0$ and $C(K) > 1$ that

$$\left\|\bar{y}^{N,\ell}(t)\right\|_H^2 < \left\|\bar{y}^{N,\ell}(t)\right\|_H^2 + \lambda \left\|\bar{u}^{N,\ell}\right\|_H^2 \leq C(K) \left\|y^\ell(t)\right\|_H^2 < \left\|y^\ell(t)\right\|_H^2.$$

Therefore, we have $\|\bar{y}^{N,\ell}(t)\|_H / \|y^\ell(t)\|_H < 1$. Hence, we conclude from (5.49), (5.50) and (5.48) that the exponential controllability condition (5.14) holds:

$$\begin{aligned} L\big(y_{[\bar{u}^{N,\ell},t_\circ,y_\circ]}(t), \bar{u}^{N,\ell}(t)\big) &\leq \frac{1}{2}\left(C(K) + 2\mathsf{Err}(t;\ell) + \mathsf{Err}(t;\ell)^2\right) \left\|y^\ell(t)\right\|_H^2 \\ &\leq \frac{1}{2} C^\ell(K)\, \sigma(K)^{t-t_\circ} \left\|y_\circ\right\|_H^2 = C^\ell(K)\, \sigma(K)^{t-t_\circ}\, L_*(y_\circ) \end{aligned}$$

with the error term

$$\mathsf{Err}(t;\ell) = \frac{\left\|y_{[\bar{u}^{N,\ell},t_\circ,y_\circ]}(t) - \bar{y}^{N,\ell}(t)\right\|_H}{\left\|\bar{y}^{N,\ell}(t)\right\|_H}$$

and the constant

$$C^\ell(K) = C(K) + \mathsf{Err}(t;\ell) + \frac{1}{2}\,\mathsf{Err}(t;\ell)^2 \geq C(K). \tag{5.51}$$

Note that $\mathsf{Err}(t;\ell)$ can be evaluated easily, since $\bar{y}^{N,\ell}$ and $y_{[\bar{u}^{N,\ell},t_\circ,y_\circ]}$ are known from Algorithm 10, steps 4 and 5, respectively. Thus, the constant $C^\ell(K)$ takes into account the approximation made by the POD reduced-order model. In the following theorem we provide an explicit formula for the scalar $\alpha^{N,\ell}$ which appears in the relaxed DPP. The notation $\alpha^{N,\ell}$ intends to stress that we are working with POD surrogate model. We summarize our result in the following theorem.

Theorem 5.5 *Let the constant C^ℓ be given by* (5.51) *and $N\Delta t$ denote the finite prediction horizon with $N \in \mathbb{N}$ and $\Delta t > 0$. Then the parameter $\alpha^{N,\ell}$ is given by the explicit formula:*

$$\alpha^{N,\ell}(K) = 1 - \frac{\big(\eta_N^\ell(K) - 1\big) \prod_{i=2}^N \big(\eta_i^\ell(K) - 1\big)}{\prod_{i=2}^N \eta_i^\ell(K) - \prod_{i=2}^N \big(\eta_i^\ell(K) - 1\big)} \tag{5.52}$$

with $\eta_i^\ell(K) = C^\ell(K)(1-\sigma^i(K))/(1-\sigma(K))$ *and* $\sigma(K)$ *as in* (5.19).

Theorem 5.5 informs we can compute the constant $\alpha^{N,\ell} \approx \alpha^N$ basicly in the same way of the full-model, replacing the constants C, η with C^ℓ, η^ℓ, respectively, taking into account the POD reduced-order modelling. To obtain the minimal horizon which ensures the asymptotic stability of the POD-NMPC scheme we maximize (5.52) according to the constraints $\alpha^{N,\ell} > 0$, $K > \max(0, \rho - \theta/C_V)$ and to the constraints in Table 5.0.

In Algorithm 10 we present the POD-MPC scheme. First step consists in computing the snapshots and the POD basis functions in order to build the reduced model. Then, we can compute the minimal prediction horizon N which ensures asymptotic stability for the surrogate model. After this set up, which is somehow different from Algorithm 9 we can perform the standard NMPC algorithm in the reduced space which acts faster, since the dimension of the problem is lower.

5.4. Numerical Tests

This section presents a comprehensive set of tests in order to show the performance of our proposed algorithm. We compare the results with solution given by the standard MPC scheme and the POD-MPC.
All the numerical simulations reported in this section have been made on a MacBook Pro with 1 CPU Intel Core i5 2.3 Ghz and 8GB RAM.

5.4.1. The finite difference approximation for the state equation

For $\mathcal{N} \in \mathbb{N}$ we introduce an equidistant spatial grid in Ω by $x_i = ih$, $i = 0, \dots, \mathcal{N}$, with the step size $\Delta x = 1/(\mathcal{N}+1)$. At $x_\circ = 0$ and $x_{\mathcal{N}+1} = 1$ the solution y is known due to the boundary conditions (5.1). Thus, we only compute approximations $y_i^h(t)$ for $y(x_i, t)$ with $1 \le i \le \mathcal{N}$ and $t \in [t_\circ, t_f]$. We define the vector $y^h(t) = (y_1^h(t), \dots, y_{\mathcal{N}}^h(t))^\top \in \mathbb{R}^{\mathcal{N}}$ of the unknowns. Analogously, we define $u^h = (u_1^h, \dots, u_{\mathcal{N}}^h)^\top \in \mathbb{R}^{\mathcal{N}}$, where u_i^h approximates $u(x_i, \cdot)$ for $1 \le i \le \mathcal{N}$. Adopting a classical second-order finite difference scheme we derive the following dynamical system

$$\dot{y}^h(t) = A(\theta)y^h(t) + F(y^h(t); \rho) + u(t) \quad \text{for } t \in (t_\circ, t_f], \tag{5.53a}$$

$$y^h(t_\circ) = y_\circ^h, \tag{5.53b}$$

where the matrix $A(\theta) \in \mathbb{R}^{\mathcal{N}\times\mathcal{N}}$ is given by

$$A(\theta) = \frac{\theta}{h^2}\begin{pmatrix} -2 & 1 & & & \\ 1 & -2 & 1 & & \\ & \ddots & \ddots & \ddots & \\ & & 1 & -2 & 1 \\ & & & 1 & -2 \end{pmatrix} + \frac{1}{2h}\begin{pmatrix} 0 & -1 & & & \\ 1 & 0 & -1 & & \\ & \ddots & \ddots & \ddots & \\ & & 1 & 0 & -1 \\ & & & 1 & 0 \end{pmatrix},$$

the parameter dependent, nonlinear function $F(\cdot\,;\rho) : \mathbb{R}^{\mathcal{N}} \to \mathbb{R}^{\mathcal{N}}$ is defined as

$$F(z;\rho) = \rho \begin{pmatrix} z_1^3 - z_1 \\ \vdots \\ z_{\mathcal{N}}^3 - z_{\mathcal{N}} \end{pmatrix}, \quad z = (z_1, \ldots, z_{\mathcal{N}}) \in \mathbb{R}^{\mathcal{N}},$$

and $y_\circ^h = (y_\circ(x_1), \ldots, y_\circ(x_{\mathcal{N}})^\top \in \mathbb{R}^{\mathcal{N}}$. Note that $A(\theta)$ is the discrete version of the operator $\mathcal{A}$ introduced in Section 5.2 and t_f is not infinte due to computation restrictions.

In Figure 5.1 the solution to (5.53) is plotted if the time integration is done by using the implicit Euler method. As we see from Figure 5.1, the uncontrolled solution does not

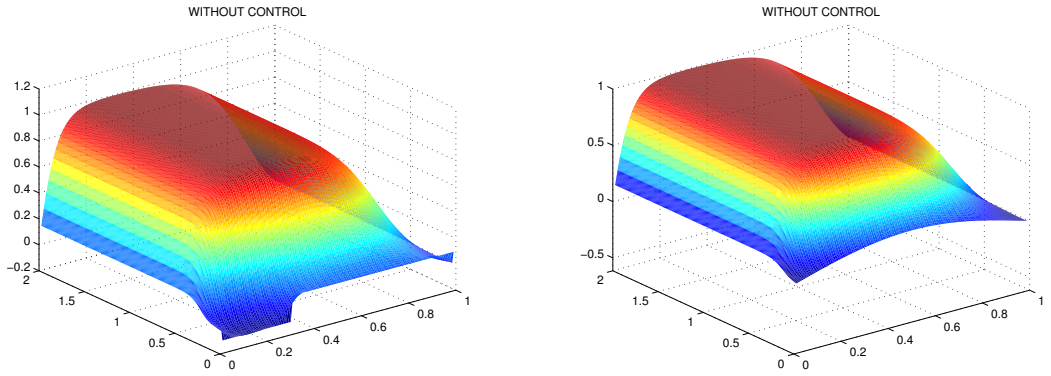

Figure 5.1.: FD state y for $y_\circ = 0.1\,\mathrm{sgn}(x - 0.3)$ (left plot) and $y_\circ = 0.2\sin \pi x$ (right plot) with $u = 0$, $(\theta, \rho) = (0.1, 11)$ and $\mathcal{N} = 99$.

tend to zero for $t \to \infty$, indeed it stabilezes at one.

5.4.2. POD-MPC experiments

In our numerical examples we choose $\widehat{y} \equiv 0$, i.e., we force the state to be close to zero, and $\lambda = 0.01$ in (5.3). A finite horizon open loop strategy without terminal constraints does not give the desired trajectory stabilized in the zero-equilibrium (see Figure 5.2), it does not hold our stability request.

This section is devoted to the performance of our POD–MPC scheme presented in Algorithm 10 of Section 5.3.3. In our tests, the snapshots are computed taking the uncontrolled system, e.g. $u \equiv 0$, in (5.1) and the corresponding adjoint equation (5.27). Several hints for the computation of the snapshots in the context of MPC are given in [45]. The nonlinear term is reduced following the Discrete Empirical Interpolation Method (DEIM) (see Section 2.1.4).

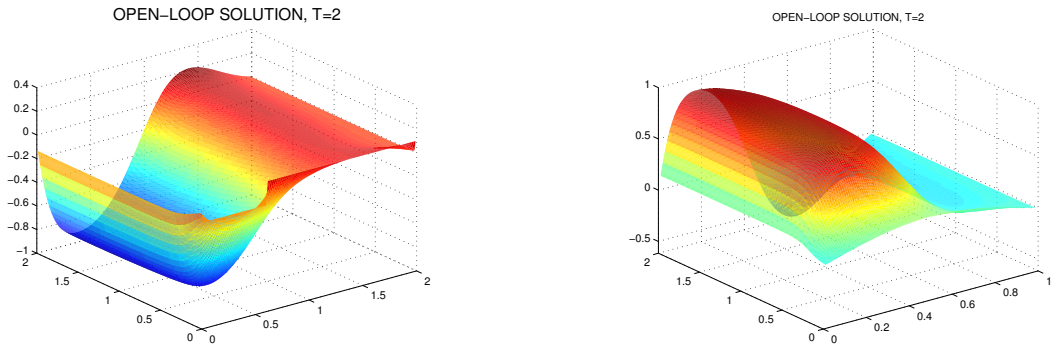

Figure 5.2.: Open-loop Solution y for $y_\circ = 0.1\,\mathrm{sgn}(x - 0.3)$, $(\theta, \rho) = (0.1, 11)$, $\mathcal{N} = 99$, $t_f = 2$ (left plot) and $y_\circ = 0.2\sin \pi x$, $(\theta, \rho) = (0.1, 11)$, $\mathcal{N} = 99$, $t_f = 2$ (right plot).

5.4.3. Test 1: Unconstrained case with smooth initial data

The first test is a simple example concerning the unconstrained optimal control problem where the parameters are given in Table 5.2.

T	Δt	Δx	θ	ρ	$y_0(x)$	u_a	u_b	N	K
0.5	0.01	0.01	1	11	$0.2\sin(\pi x)$	$-\infty$	∞	10	2.46

Table 5.2.: Test 1: Setting for the optimal control problem, minimal stabilizing horizon N and feedback constant K.

According to the computation of α^N in (5.20) related to the Relaxed Dynamic Programming Principle, the minimal horizon that guarantess asymptotic stability is $N = 10$ as shown in [10]. Even in the POD-NMPC scheme the asymptotic stability is achieved for $N = 10$, provided that $\mathsf{Err}(t; \ell) \leq 10^{-3}$ for all $t \geq t_\circ$. In Figure 5.3 we show the controlled state trajectory computed by Algorithm 9 taking $N = 3$, $N = 6$ and $N = 10$.

At the top of Figure 5.3 we show the approximation taking in the MPC algorithm the prediction horizon $N = 3$ and $N = 6$. As we can see, we do not get a stabilizing feedback for $N = 3$, whereas $N = 10$ leads to a state trajectory which tends to zero for $t \to \infty$. Note that we plot the solution only on the time interval $[0, 0.5]$ in order to have a zoom of the solution. Further, in Figure 5.3 the solution related to $u = -Ky$ is presented. As we can see, the NMPC control stabilized to the origin very soon while the control law $u = -Ky$ requires a larger time horizon. This is due to the fact we are controlling the equation with a very restrictive class of controls but, on the other hand, it is still guaranteed the stabilization by the theory the asymptotic stability. In Table 5.3 we present the error in $L^2(t_\circ, T; H)$-norm considering the solution coming from the

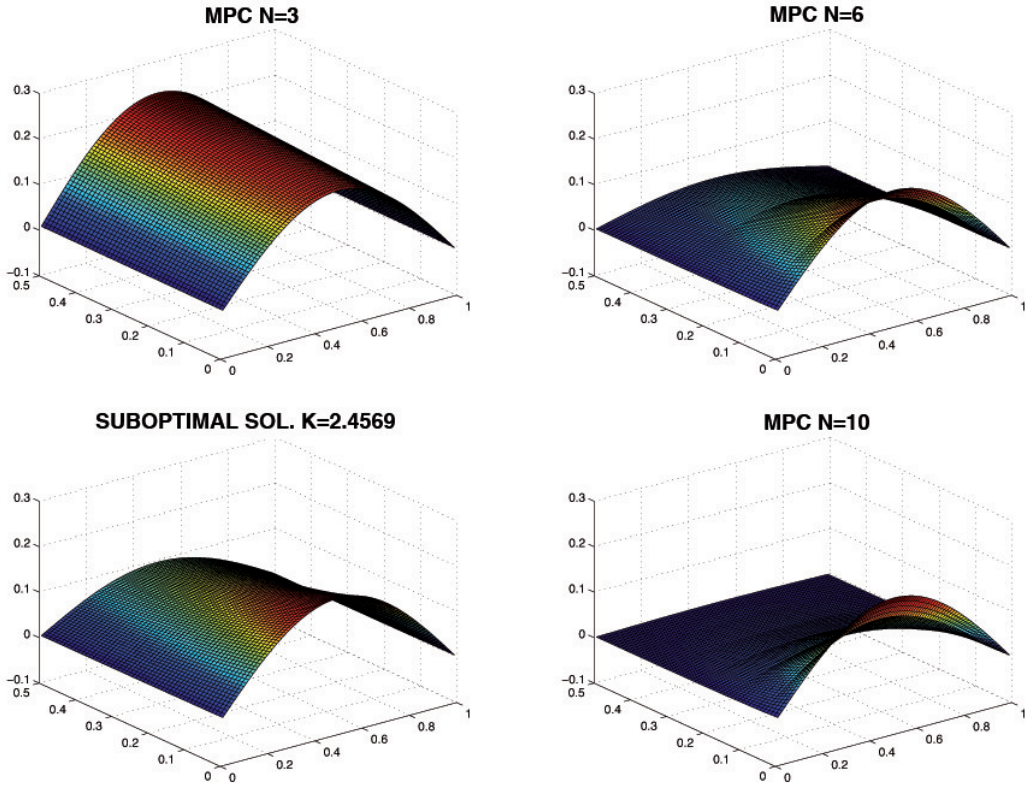

Figure 5.3.: Test 1: NMPC state with $N = 3$ and $N = 6$ (top), with $N = 10$ and with $u = -Ky$ (bottom).

Algorithm 9 as the truth solution (y^{FD} in the Table). The examples are computed with $\mathsf{Err}(t,\ell) \leq 10^{-3}$. The CPU time for the full-model turns out to be 49 seconds, in the

	$\hat{J}$	time	K	$\|y^{FD} - y\|_{L^2(t_\circ,T;H)}$
Solution with $u = -Ky$	0.0025		2.46	0.0145
Alg. 9	0.0015	49s		
Alg. 10 ($\ell = 13, \ell^{DEIM} = 15$)	0.0016	8s		0.0047
Alg. 10 ($\ell = 3, \ell^{DEIM} = 2$)	0.0016	6s		0.0058

Table 5.3.: Test 1: Evaluation of the cost functional, CPU time, suboptimal solution.

POD-suboptimal approximation with only three POD and two DEIM basis functions requires 6 seconds. We can easily observe an impressive speed up of a factor eight. Moreover, the evaluation of the cost functional in the full model and the POD model provides very close values. The CPU time of the suboptimal solution is not comparable since in other simulations we have not taken into account the time needed to compute the minimal horizon N. The evaluation of the cost functional emphasizes we can take the suboptimal model as an upper bound.

5.4.4. Test 2: Constrained case with smooth initial data

In this example, in contrast to Test 1, we choose $u_a = -0.3$ and $u_b = 0$. As expected, the minimal horizon N increases compared to Test 1 (see Table 5.3). As one can see

T	Δt	Δx	θ	ρ	$y_0(x)$	u_a	u_b	N	K
0.5	0.01	0.01	1	11	$0.2\sin(\pi x)$	-0.3	0	14	1.50

Table 5.3.: Test 2: Setting for the optimal control problem, minimal stabilizing horizon N and feedback constant K.

from Figure 5.4 the NMPC state with $N = 14$ tends faster to zero than the state with $u = -Ky$.

The solution coming from the POD model is in the middle of Figure 5.4. Note that $\mathcal{E}(\ell = 3) = 0.99, \mathcal{E}(\ell = 13) = 1$, and $\mathsf{Err}(t;\ell) \leq 10^{-3}$ for any ℓ and $t \geq t_\circ$. Indeed, Table 5.4 presents the evaluation of the cost functionals for the proposed algorithms and the CPU time which shows that the speed up by the reduced order approach is about 16. Note that K in Test 2 is smaller compared to Test 1 due to the constraint of the control space. Further, the error is presented in Table 5.4.

To study the influence of $\mathsf{Err}(t;\ell)$ we present in Figure 5.5, on the left, how the optimal prediction horizon N changes according to different tolerance. The blue line corresponds to the optimal prediction horizon in Test 1, and the red one to Test 2. It turns out that, until $\mathsf{Err}(t;\ell) \leq 10^{-3}$, we can work exactly with the same horizon N we had in

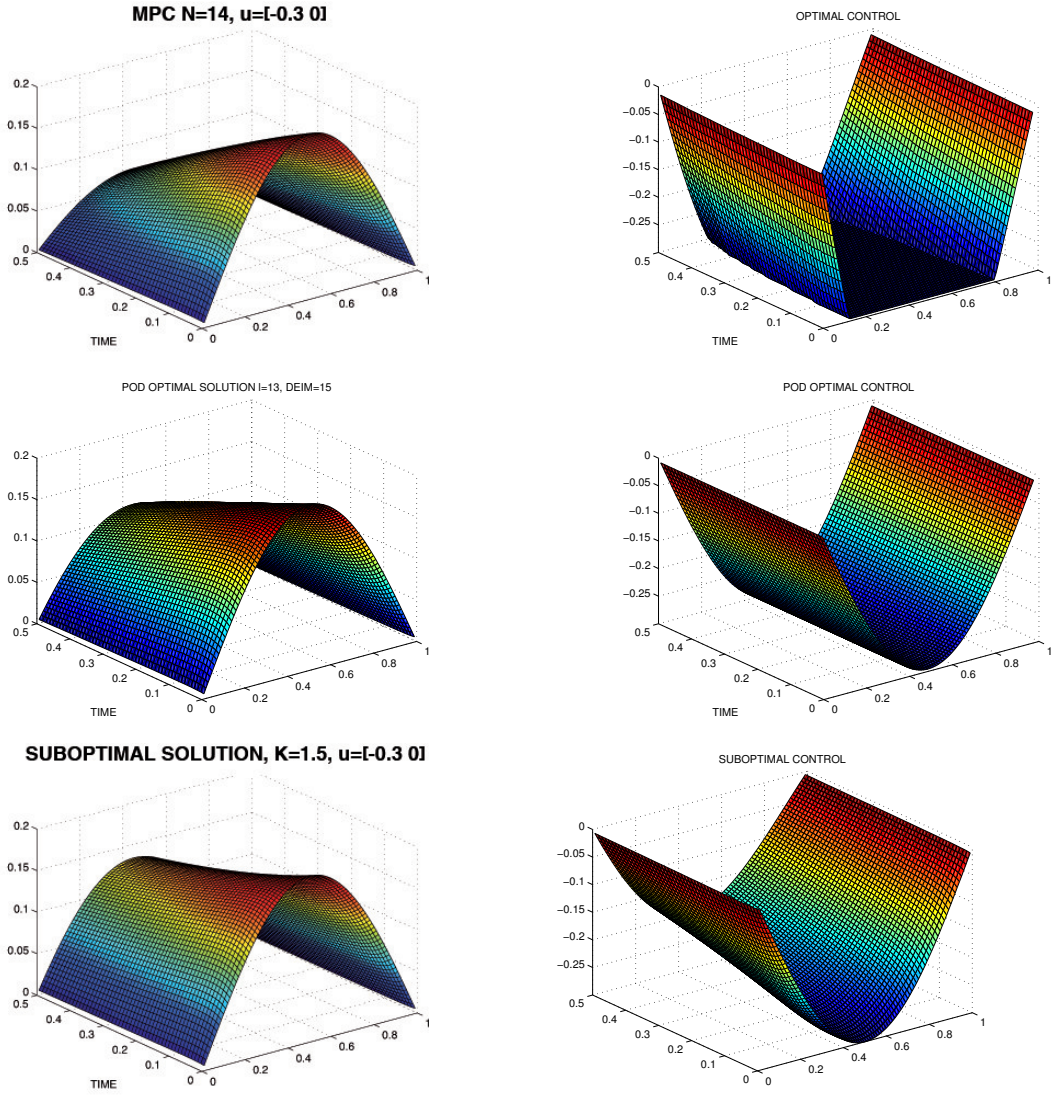

Figure 5.4.: Test 2: NMPC state with $N = 14$ and corresponding optimal control (top), POD-MPC solution with $N = 10$ and optimal control (middle), suboptimal solution with $u = -Ky$ (bottom).

	$\hat{J}$	time	K	$\|y^{FD} - y\|_{L^2(t_\circ,T;H)}$
Solution with $u = -Ky$	0.0035		1.50	0.0089
Alg. 9	0.0027	65s		
Alg. 10 ($\ell = 13$, $\ell^{DEIM} = 15$)	0.0032	5s		0.0054
Alg. 10 ($\ell = 3$, $\ell^{DEIM} = 2$)	0.0033	4s		0.0055

Table 5.4.: Test 2: Evaluation of the cost functional, CPU times, suboptimal solution.

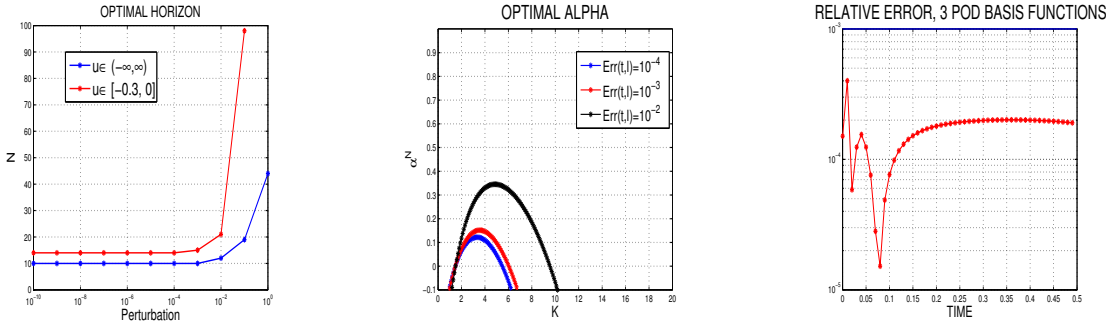

Figure 5.5.: Test 2: Optimal horizon N and $\alpha^{N,\ell}$ according to different $\mathsf{Err}(t;\ell) = 10^{-3}$, Influence of the relative error $t \mapsto \mathsf{Err}(t;\ell) = 10^{-3}$ for $\ell = 3$.

the full model. In the middle plot of Figure 5.5 there is a zoom of the function α with different values of $\mathsf{Err}(t;\ell)$. The plot on the right side of Figure 5.5 shows the relative error $\mathsf{Err}(t;\ell)$ for $0 \leq t \leq 0.5$ with $\ell = 3$.

One of the big advantages of feedback control is the stabilization under perturbation of the system. The perturbation of the initial condition is a typical example which comes from many applications in fact, often the measurements may not be correct. For a given noise distribution $\delta = \delta(x)$ we consider a perturbation the following form:

$$y_0(x) = \big(1 + \delta(x)\big) y_\circ(x) \quad \text{for } x \in \Omega.$$

The study of the asympotic stability does not change: we can compute the minimal prediction horizon as before. As we can see in Figure 5.6 the POD-NMPC algorithm is able to stabilize with a noise of $|\delta(x)| \leq 30\%$.

5.4.5. Test 3: Constrained case with smooth initial data

Now we decrease the diffusion term and, as a consequence, the prediction horizon N increases; see Table 5.6 and middle plot of Figure 5.6. Even if the prediction horizon is very large, the proposed Algorithm 10 accelerates the approximation of the problem. Note that, in our example, the diffusion term is still relevant such that we can work with only 2 POD basis functions. The CPU time in the full model is 84 seconds, whereas

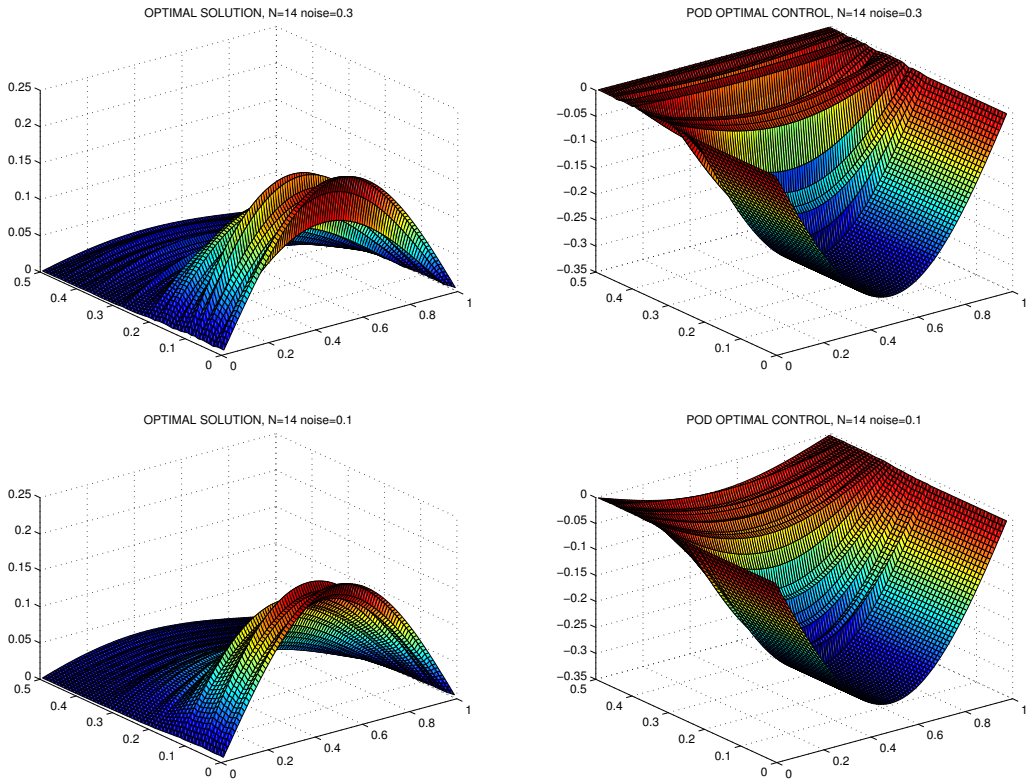

Figure 5.6.: Test 5: POD optimal solution with $\ell = 3, \ell^{DEIM} = 2$. Noise=30% (top), 10% (bottom).

T	Δt	Δx	θ	ρ	$y_0(x)$	u_a	u_b	N	K
0.5	0.01	0.01	$1/\sqrt{2}$	10	$0.2\sin(\pi x)$	-1	0	30	5

Table 5.6.: Test 3: Setting for the optimal control problem.

with a low-rank model, such as $\ell = 2$ we obtained the solution in five seconds and an impressive speed up factor of 16. Even with a more accurate POD model we have a very good speed up factor of nine. The evaluation of the cost functional is given in Table 5.6. In Figure 5.7, we present the optimal solution with $N = 30$ and the optimal

	$\hat{J}$	time	K	$\|y^{FD} - y\|_{L^2(t_\circ,T;H)}$
Suboptimal solution ($u = -Ky$)	0.0021		5	0.0208
Algorithm 9	0.0016	84s		
Algorithm 10 ($\ell = 16$, $\ell^{DEIM} = 16$)	0.0017	9s		0.0092
Algorithm 10 ($\ell = 2$, $\ell^{DEIM} = 3$)	0.0018	5s		0.0093

Table 5.6.: Test 3: Evaluation of the cost functional and CPU time.

control beetwen $[-1, 0]$. As expected the control -1 is acting mostly at the begining, then the optimal control is decreasing close to 0. At the bottom of Figure 5.7 we put the suboptimal solution coming from the POD model reduction and the corresponding control with 16 POD basis functions and 16 DEIM basis elements. The error between the NMPC state and the POD-MPC state is less than 0.01.

5.4.6. Test 4: Constrained case with non-smooth initial data

In the last test we focus on a different initial condition and different control constraints. The parameters are presented in Table 5.7. The minimal horizon N which ensures asymp-

T	Δt	Δx	θ	ρ	$y_0(x)$	u_a	u_b	N	K
0.5	0.01	0.01	1/2	5	$\mathrm{sgn}(x - 0.3)$	-1	1	43	9.99

Table 5.7.: Test 4: Setting for the optimal control problem.

totic stability is $N = 43$. Table 5.7 emphazises again the performance of the POD-NMPC method with an acceleration 12 times faster of the full model. The evaluation of the

	$\hat{J}$	time	K	$\|y^{FD} - y\|_{L^2(t_\circ,T;H)}$
Solution with $u = -Ky$	4.7e-4		9.99	0.006
Alg. 9	4.1e-4	50s		
Alg. 10 ($\ell = 17$, $\ell^{DEIM} = 19$)	4.4e-4	12s		0.0034
Alg. 10 ($\ell = 3$, $\ell^{DEIM} = 4$)	4.4e-4	4s		0.0035

Table 5.7.: Test 4: Cost functional, CPU time and suboptimal solution.

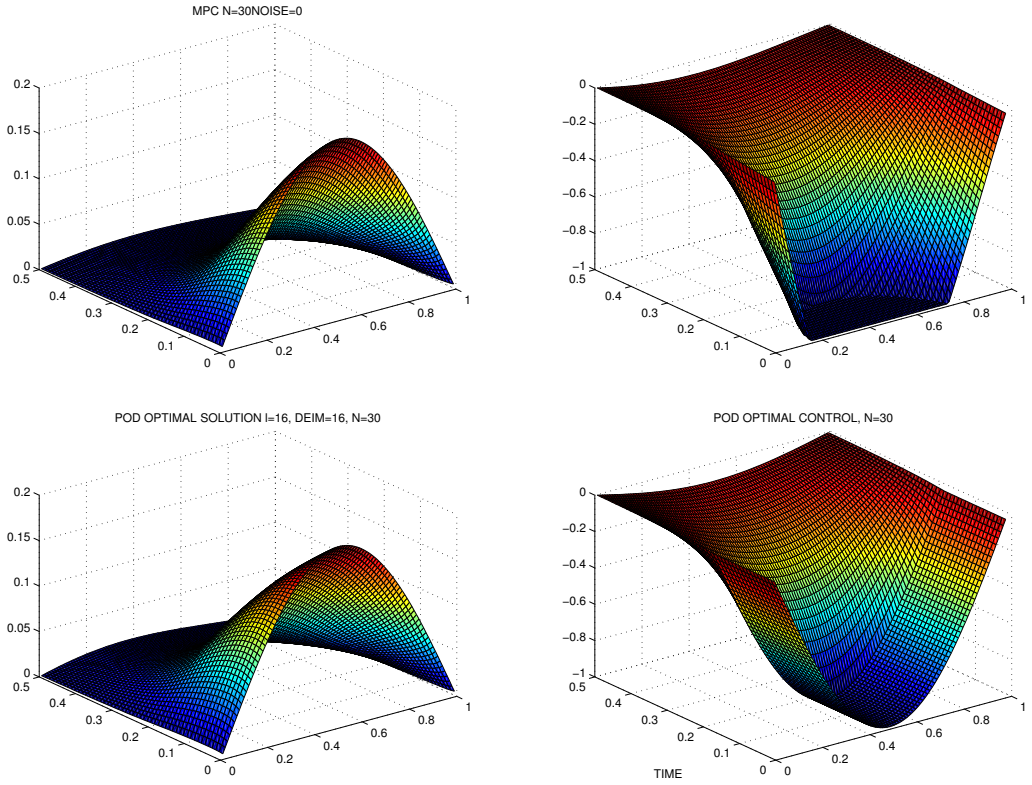

Figure 5.7.: Test 3: Optimal solution with MPC and prediction horizon $N = 30$ (top). Suboptimal solution and control from POD model reduction (bottom).

cost functional gives the same order in all the simulation we provide. In Figure 5.8 we present the NMPC state for $N = 43$ (left plot), the POD-NMPC state with $N = 43$, $\ell = 3$, $\ell^{DEIM} = 4$ (middle plot) and the increase of the optimal horizon N according to the perturbation $\mathsf{Err}(t;\ell)$. In Figure 5.8 on the left the optimal solution, on the right side we show the optimal horizon N increase according to the perturbation $\mathsf{Err}(t;\ell)$. Figure

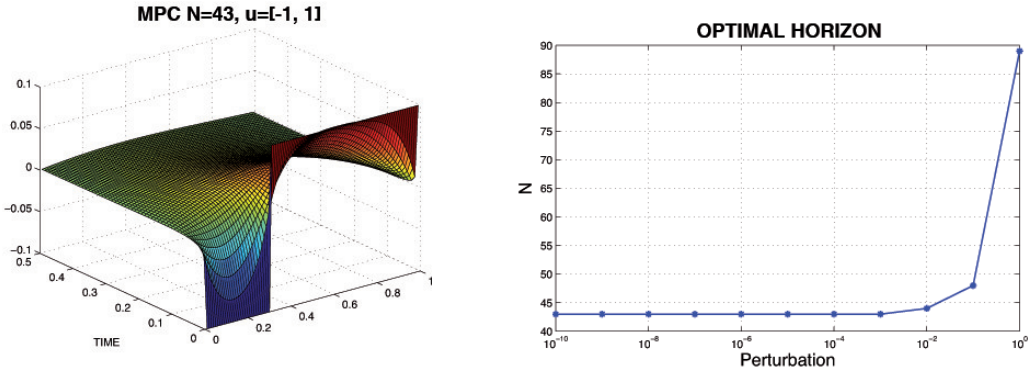

Figure 5.8.: Test 4: Optimal solution with MPC and prediction horizon $N = 40$ (left). Optimal horizon with different $\mathsf{Err}(t;\ell)$ (right).

5.9 shows the POD approximation with 3 POD basis functions and 4 DEIM basis. The error between the MPC approximation and the POD-MPC is 0.0192.

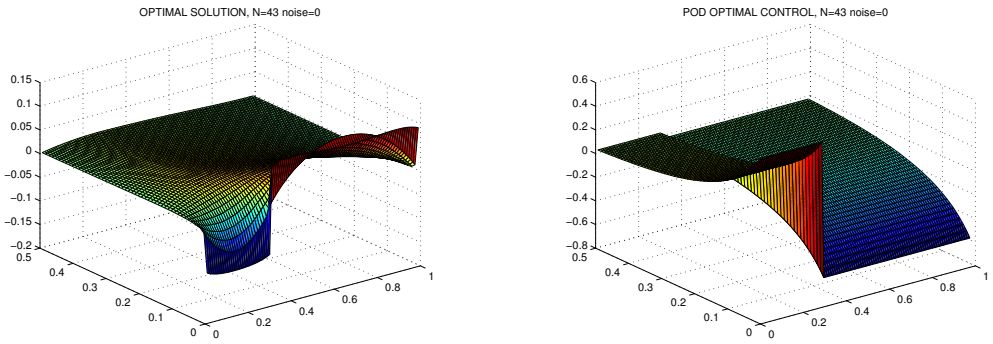

Figure 5.9.: Test 4: POD optimal solution and control with $\ell = 3, \ell^{DEIM} = 4$.

The error between the NMPC state and the POD-MPC state is 0.0035 when $\mathcal{E}(\ell = 3) = 0.99$, whereas for $\mathcal{E}(\ell = 17) = 1$ the error is 0.0034.

6. Conclusions and future directions

This thesis has demonstrated the crucial importance of Dynamic Programming Principle and POD model order reduction for optimal control problems with PDE constraints.

VI-PI approximation scheme
In this thesis we have presented an accelerated algorithm for the solution of static HJB equations arising in different optimal control problems. The proposed method considers a pre-processing value iteration procedure over a coarse mesh with relaxed stopping criteria, which is used to generate a good initial guess for a policy iteration algorithm. This leads to accelerated numerical convergence with respect to the successive approximation method, with a speedup ranging in average from $4\times$ to $8\times$. We have assessed the performance of the new scheme via an extensive set of numerical tests focusing on infinite horizon and minimum time problems, providing numerical evidence of the reliability of the method in tests with increasing complexity. We have also included an application related to optimal control of partial differential equations, which is an area where difficulties related to the high dimension of the discretized problem naturally arise. Positive aspects of the proposed scheme are its wide applicability spectrum (in general for static HJB), and its insensitivity with respect to the complexity of the discretized control set. Nonetheless, for some non-trivial targets, special care is needed in order to ensure that the coarse pre-processing step will actually lead to an improved behavior of the policy iteration scheme. Certainly, several directions of research remain open. The aim of Chapter 3 was to present the numerical scheme and provide a numerical assessment of its potential. Future work should focus on tuning the algorithm in order to achieve an optimal performance; for instance, in order to make a fair comparison with the value iteration algorithm, the policy iteration step was also performed via a successive approximation scheme, whereas better results could be obtained by using a more efficient solver. Other possible improvements would be related to multigrid methods, high-order schemes and domain decomposition techniques. An area of application that remains unexplored is the case of differential games, where Hamilton-Jacobi-Isaacs equations need to be solved. Results presented in [21] indicate that the extension is far from being trivial since a convergence framework is not easily guaranteed and the policy iteration scheme requires some modifications.

POD and HJB coupling
We have also proposed a computational approach for the control of linear and semilinear partial differential equations based on the coupling between a POD-Galerkin method and the Dynamic Programming approach. The examples show that even with a low number of POD basis functions we can compute a rather accurate solution by means of an

adaptive technique. This is still an essential requirement when dealing with the Dynamic Programming approach, which suffers from the curse-of-dimensionality although recent developments in the methods used for HJB equations will allow to increase this bound in the next future (for example by applying patchy techniques). The method requires the use of an accuracy indicator for the POD approximation and of the residual to control the global error. So the time interval is splitted into a union of sub-intervals where the POD basis functions will not change. The adaptive technique is not very expensive with respect to the global solution of the control problem. Although at present this technique has been implemented just for 1-dimensional problems, the results seem to indicate that this could be a reasonable approach also for 2-dimensional problems and we will try to proceed in this direction. Then we want to extend this approach to multidimentional problems trying to overcome the actual limits on the number of POD basis functions.

MPC methods

Finally, we have proposed a new numerical method for optimal control problems which tries to stabilize a one dimensional semilinear parabolic equation by means of Model Predictive Control. We presented asymptotic stability conditions where the control space is bounded for a suboptimal problem coming from a particular class of feedback controls. Since the CPU time of the full dimensional algorithm may increase with the dimension of the prediction horizon, we have presented a deep study of the suboptimal model which comes from POD model reduction. We have given an *a-priori* error estimate for the computation of the prediction horizon of the suboptimal model. Therefore, the new reduced model approach turns out to be very efficient with respect to the full dimensional problem. Although the algorithm is applied to a one dimensional problem, the theory is rather general and applied to more dimensioanl equations, not only with POD model reduction but any method. Another important issue is due to the computation of the snapshots, one could adopt different strategies to obtain the surrogate model. Then, it would be very interesting to investigate the stability property of MPC algorithm, computing step by step a less accurate control. Another interesting direction is to extend the technique to more general problems and increase the state dimensions.

A. Appendix

In this Appendix we recall a convergence result concerning Newton's method and a theorem on the perturbation of Ordinary Differential Equation.

A.1. Newton's theorem

Here, we show a convergence result for Newton's method. More details can be found in [33]. Assume we have to solve a nonlinear operator equation

$$F(x) = 0,$$

wherein $F : D \subset X \to Y$ for Banach spaces X, Y. Let F be at least once continuously differentiable. Suppose we have a starting guess x_0 of the unknown solutions x^* at hand. Then the succesive linearization leads to the general Newton method

$$F'(x_k)\Delta x_k = -F(x_k), \quad x_{k+1} = x_k + \Delta x_k, \quad k = 0, 1, \dots \tag{A.1}$$

In the following theorem we provide some conditions to ensure the convergence of the method for a given intial guess x_0.

Theorem A.1 *Let $F : D \to Y$ be a continuously Fréchet differentiable mapping with $D \subset X$ open and convex. For a starting point $x_0 \in D$ let $F'(x_0)$ be invertibile. Assume that*

$$\|F'(x_0)^{-1}F(x_0)\| \leq \alpha \tag{A.2}$$

$$\|F'(x_0)^{-1}(F'(y) - F'(x))\| \leq \bar{\omega}\|y - x\| \quad x, y \in D \tag{A.4}$$

$$h_0 := \alpha\bar{\omega} \leq \frac{1}{2} \tag{A.5}$$

$$B(x_0, \rho) \subset D \qquad \rho := \frac{1 - \sqrt{1 - 2h_0}}{\bar{\omega}} \tag{A.6}$$

Then te sequence $\{x_k\}$ obtained from the ordinary Newton iteration is well-defined, remains in $B(x_0, \rho)$, and converges to some x^* with $F(x^*) = 0$. For $h_0 < \frac{1}{2}$, the convergence is quadratic.

A.2. Continuous dependence on data

Let us recall the main results for ODEs concerning continuous dependence on parameter:

Theorem A.2 *Consider the pair of differential equations*

$$\begin{aligned} \dot{x} &= f(t,x) & \dot{y} &= g(t,y) \\ x(t_0) &= x_0 & y(t_0) &= y_0 \end{aligned}$$

where f and g are assumed to be continuous in some domain $\Omega \subset [0,T] \times \mathbb{R}^n$, and for all $(t,x) \in \Omega$

$$\|f(t,x) - g(t,x)\| \leq \varepsilon$$

and

$$\|x_0 - y_0\| \leq \delta$$

for some $\varepsilon, \delta > 0$ small. Furthermore, assume that f satisfies the Lipschitz condition

$$\|f(t,x) - f(t,y)\| \leq L\|x - y\|.$$

Then, there exists an interval (m, M) such that

$$\|x(t) - y(t)\| \leq \left(\delta + \frac{\varepsilon}{L}\right) e^{L|t-t_0|} - \frac{\varepsilon}{L} \tag{A.6}$$

for all $t \in (m, M)$.

List of Algorithms

Bibliography

[1] A. Alla, M. Falcone, *An adaptive POD approximation method for the control of advection-diffusion equations,* In K. Kunisch, K. Bredies, C. Clason, G. von Winckel, (eds) *Control and Optimization with PDE Constraints,* International Series of Numerical Mathematics, **164**, Birkhäuser, Basel, 2013, 1-17.

[2] A. Alla, M. Falcone, *A Time-Adaptive POD Method for Optimal Control Problems,* to appear in the Proceedings of the 1st IFAC Workshop on Control of Systems Modeled by Partial Differential Equations,

[3] A. Alla, M. Falcone, D. Kalise, *An Efficient Policy Iteration Algorithm for Dynamic Programming Equations* to appear PAMM, 2013.

[4] A. Alla, M. Falcone, D. Kalise, *An Efficient Policy Iteration Algorithm for Dynamic Programming Equations*, submitted to SIAM J. of Scientific Computing, 2013.

[5] A. Alla, M. Falcone, D. Kalise, *An accelerated value/policy iteration scheme for the solution of DP equations,* submitted to ENUMATH 2013

[6] A. Alla, S. Volkwein, *Asymptotic Stability of POD based Model Predictive Control for a semilinear parabolic PDE,* submitted to ACOM: *Model Reduction of Parametrized Systems*

[7] G. Allaire, *Shape optimization by the homogenization method,* Applied Mathematical Sciences **146**, Springer Verlag, 2002.

[8] F. Allgöwer, H. Chen, *A quasi-infinite horizon nonlinear model predictive control scheme with guaranteed stability,* Automatica, **34** (1998), 1205-1217.

[9] F. Allgöwer, R. Findeisen, Z.K. Nagy, *Nonlinear Model Predictive Control: From Theory to Application,* J. Chin. Inst. Chem. Engrs., **35** (2004), 299-315.

[10] N. Altmüller, L. Grüne, K. Worthmann, *Receding Horizon optimal control for the wave equation,* Proceedings of the 49th IEEE Conference on Decision and Control, Atlanta, Georgia, (2010), 3427 - 3432.

[11] K. Alton, I. M. Mitchell, *An ordered upwind method with precomputed stencil and monotone node acceptance for solving static convex Hamilton-Jacobi equations,* J. Sci. Comput., **51** (2012), 313–348.

[12] A.C. Antoulas, *Approximation of large-scale Dynamical Systems,* SIAM, Philadelphia, 2005.

[13] J.A. Atwell, B.B. King, *Proper Orthogonal Decomposition for Reduced Basis Feedback Controllers for Parabolic Equations*, Matematical and computer modelling. **33** (2001), 1-19.

[14] M. Bardi, M. Falcone, *An approximation scheme for the minimum time function*, SIAM J. Control Optim., **28** (1990), 950-965.

[15] M. Bardi, I. Capuzzo Dolcetta, *Optimal control and viscosity solutions of Hamilton-Jacobi-Bellman equations,* Birkhäuser, Basel, 1997.

[16] G. Barles, *Solutions de visocité des equations de Hamilton-Jacobi,* Springer, Berlin, 1994.

[17] M. Barrault, Y. Maday, N.C. Nguyen, A.T. Patera, *An empirical interpolation method: application to efficient reduced-basis discretization of partial differential equations,* Comptes Rendus Mathematiques, **339** (2004), 667-672.

[18] R. Bellman, Dynamic Programming, Princeton University Press, Princeton, NJ, 1957.

[19] P. Benner, E. Quintana-Ortí, *Solving stable gerenaralized Lyapunov equations with the matrix sign function,* Numerical Algorithms, **20** (1999), 75-100.

[20] D. Bini, M. Capovani, O. Menchi, *Metodi Numerici per l'Algebra Lineare*, Zanichelli, 1988.

[21] O. Bokanowski, S. Maroso, H. Zidani, *Some convergence results for Howard's algorithm,* SIAM Journal on Numerical Analysis **47** (2009), 3001–3026.

[22] O. Bokanowski, E. Cristiani, H. Zidani, *An efficient data structure to solve front propagation problems,* J. Sci. Comput., **42** (2010), 251-273.

[23] N.D. Botkin, K. Hoffman, V. Turova, *Stable numerical schemes for solving Hamilton Jacobi Bellman Isaacs equations*, SIAM J. of Scientific Computing, **33** (2011), 992-1007.

[24] S. Cacace, E. Cristiani, M. Falcone, A. Picarelli, *A patchy dynamic programming scheme for a class of Hamilton-Jacobi-Bellman equations,* SIAM J. of Scientific Computing, **34** (2012), 2625-2649.

[25] S. Cacace, E. Cristiani, M. Falcone, *Can local single pass methods solve any stationary Hamilton-Jacobi-Bellman equations?*, preprint, 2013 available on `arXiv:1301.6775`.

[26] I. Capuzzo Dolcetta, M. Falcone, *Discrete dynamic programming and viscosity solutions*, Annales de l'Institut Henry Poincaré- Analyse non-lineaire, **6** (supplement) (1989), 161-184.

[27] E. Carlini, M. Falcone, R. Ferretti, *An efficient algorithm for Hamilton-Jacobi equations in high dimension.* Computing and Visualization in Science, **7** (2004), 15-29.

[28] T. Cazenave, A. Haraux, *An Introduction to Semilinear Evolution Equation,* Oxford Science Publications, 1998.

[29] S. Chaturantabut, D.C. Sorensen, *Discrete Empirical Interpolation for NonLinear Model Reduction,* SIAM J. of Scientific Computing, **32** (2010), 2737-2764.

[30] M.G.Crandall, P.L. Lions, *Two approximations of solutions of Hamilton -Jacobi equations,* Math. Comput. **43** (1984), 1-19.

[31] R.F. Curtain, H.J. Zwart, *An introduction to infinite-dimensional linear systems theory,* Springer-Verlag, 1995.

[32] R. Dautray, J.L. Lions, *Mathematical Analysis and Numerical Methods for Science and Technology* Volume 5: Evolution Problems I, Springer, Berlin, 2000.

[33] P. Deuflhard, *Newton Methods for Nonlinear Problems. Affine Invariance and Adaptive Algorithms.* Springer Series in Computational Mathematics, **35**, Springer, Berlin, 2004.

[34] L.C. Evans, *Partial Differential Equations,* American Math. Society, Providence, Rhode Island, 2008.

[35] R. Findeisen, F. Allgöwer, *An Introduction to Nonlinear Model Predictive Control* In C.W. Scherer and J.M. Schumacher, editors, Summerschool on *The Impact of Optimization in Control*, Dutch Institute of Systems and Control, DISC, 2001.

[36] R. Findeisen, F. Allgöwer, *The quasi-infinte horizon approach to nonlinear model predictive control,* In A. Zinober and D. Owens, editors, *Nonlinear and Adaptive Control, Lecture Notes in Control and Information Sciences,* Springer-Verlag, Berlin, 2002, 89-105.

[37] M. Falcone, *A numerical approach to the infinite horizon problem of deterministic control theory*, Applied Mathematics and Optimization, **15** (1987), 1-13.

[38] M. Falcone, *Numerical solution of Dynamic Programming equations*, Appendix A in the volume M. Bardi and I. Capuzzo Dolcetta, *Optimal control and viscosity solutions of Hamilton–Jacobi–Bellman equations*, Birkhäuser, Boston, 1997, 471-504.

[39] M. Falcone, R. Ferretti, *Discrete time high-order schemes for viscosity solutions of Hamilton-Jacobi-Bellman equations*, Numer. Math., **67** (1994), 315-344.

[40] M. Falcone, R. Ferretti, *Semi-Lagrangian Approximation Schemes for Linear and Hamilton-Jacobi Equations,* SIAM, 2013.

[41] M. Falcone, T. Giorgi, *An approximation scheme for evolutive Hamilton-Jacobi equations*, In W.M. McEneaney, G. Yin and Q. Zhang (eds.), *Stochastic Analysis, Control, Optimization and Applications: A Volume in Honor of W.H. Fleming*, Birkhäuser, 1999, 289-303.

[42] M. Falcone, P. Lanucara and A. Seghini, *A splitting algorithm for Hamilton-Jacobi-Bellman equations,* Applied Numerical Mathematics, **15** (1994), 207-218.

[43] W. H. Fleming, R.W. Rishel, *Deterministic and stochastic optimal control,* Springer-Verlag, New York, 1975.

[44] M. Gerdts, *Optimal Control of ODEs and DAEs,* DeGruyter-Verlag Berlin, 2006.

[45] J. Ghiglieri, S. Ulbrich, *Optimal Flow Control Based on POD and MPC and an Application to the Cancellation of Tollmien-Schlichting Waves,* to appear in Optimization Methods and Software, 2012.

[46] S. Gugercin, A. Antoulas, *A survey of model reduction by balanced truncation and some new results* Internat. J. Control, **77** (2004), 748-766.

[47] G. Golub, C.F. Van Loan, *Matrix Computation* Johns Hopkins University Press, 1996.

[48] R. Gonzáles, C. Sagastizábal, *Un algorithme pour la résolution rapide d'équations discrètes de Hamilton-Jacobi-Bellman*, C.R. Acad. Sci. Paris, Sér. I, **311** (1990), 45-50.

[49] M. Grepl, *Certified reduced basis method for nonaffine linear time-varying and nonlinear parabolic partial differential equations,* Math. Models Methods Appl. Sci., **22** (2012), 1-40.

[50] L. Grüne, *An adaptive grid scheme for the discrete Hamilton-Jacobi-Bellman equation*, Numer. Math., **75** (1997), 319-337.

[51] L. Grüne, J. Panneck, M. Seehafer, K. Worthmann, *Analysis of unconstrained nonlinear MPC schemes with time varying control horizon,* SIAM Journal on Control and Optimization, **48** (2010), 4938 - 4962.

[52] L. Grüne, J. Pannek, *Nonlinear Model Predictive Control,* Springer London, 2011. Automatica, **49** (2013), 725-734.

[53] M. Gubisch, S. Volkwein, *Proper Orthogonal Decomposition for Linear-Quadratic Optimal Control,* submitted 2013, available on `http://kops.ub.uni-konstanz.de`

[54] M. Hinze, R. Pinnau, M. Ulbrich, S. Ulbrich, *Optimization with PDE Constraints. Mathematical Modelling: Theory and Applications,* **23**, Springer Verlag, 2009.

[55] M. Hinze, S. Volkwein *Proper orthogonal decomposition surrogate models for nonlinear dynamical systems: error estimates and suboptimal control,* In P. Benner, V. Mehrmann, D. C. Sorensen (eds.), *Reduction of Large-Scale Systems,* Lecture Notes in Computational Science and Engineering, **45** (2005), 261-306.

[56] P. Holmes, J.L. Lumley, G. Berkooz, C.W. Romley, *Turbulence, Coherent Structures, Dynamical Systems and Symmetry,* Cambridge Monographs on Mechanics, Cambridge University Press, 2nd edition, 2012.

[57] R.A. Howard. *Dynamic programming and Markov processes,* Wiley, New York, 1960.

[58] L. Iapichino, S. Volkwein, *Greedy sampling of distributed parameters in the reduced-basis method by numerical optimization,* submitted, 2013.

[59] R. Kalaba, *On nonlinear differential equations, the maximum operation and monotone convergence,* J. of Math. Mech., **8** (1959), 519-574.

[60] E. Kammann, F. Tröltzsch, S. Volkwein, *A method of a-posteriori error estimation with application to proper orthogonal decomposition,* ESAIM: Mathematical Modelling and Numerical Analysis, **47** (2013), 555-581.

[61] C.T. Kelley, *Iterative method for Optimization,* SIAM, 1999.

[62] G. Kossioris, C. Makridakis, P. Souganidis, *Finite volume schemes for Hamilton Jacobi equations,* Numer. Math. **83** (1999), 427 - 442.

[63] K. Kunisch, S. Volkwein, *Control of Burgers' Equation by a Reduced Order Approach using Proper Orthogonal Decomposition,* Journal of Optimization Theory and Applications, **102** (1999), 345- 371.

[64] K. Kunisch, S. Volkwein, *Galerkin proper orthogonal decomposition methods for parabolic problems* Numer. Math. **90** (2001), 117-148.

[65] K. Kunisch, S. Volkwein, *Galerkin Proper Orthogonal Decomposition Methods for a general equation in Fluid Dynamics*, SIAM J. Num. Anal., **40** (2002), 492-515.

[66] K. Kunisch, S. Volkwein. *Optimal snapshot location for computing POD basis functions* ESAIM: Mathematical Modelling and Numerical Analysis **44** (2010), 509-529.

[67] K. Kunisch, S. Volkwein, L. Xie, *HJB-POD Based Feedback Design for the Optimal Control of Evolution Problems.* SIAM J. on Applied Dynamical Systems, **4** (2004), 701-722.

[68] K. Kunisch, L. Xie, *POD-Based Feedback Control of Burgers Equation by Solving the Evolutionary HJB Equation*, Computers and Mathematics with Applications, **49** (2005), 1113-1126.

[69] K. Kunisch, L. Xie, *Suboptimal feedback control of flow separation by POD model reduction,* SIAM Computational Science and Engineering, **12** (2006), 233-250.

[70] I.T. Jolliffe, *Principal Component Analysis,* Springer-Verlag, 1986.

[71] J.L. Lions, *Optimal control of systems governed by partial differential equations,* Springer-Verlag, New York 1971.

[72] M. Loéve, *Probability theory.* Vol. II, 4th ed. Graduate Texts in Mathematics **46**, Springer-Verlag, 1978.

[73] H. Maurer, *Numerical Solution of singular control problems using multiple shooting techniques,* J. of Optimization Theory and Applications, **18** (1976), 235-257.

[74] N.C. Nguyen, G. Rozza, A.T. Patera. *Reduced basis approximation and a posteriori error estimation for time dependent viscous Burgers equation.* Calcolo, **46** (2009), 157-185.

[75] G. Pannocchia, J.B. Rawlings, S.J. Wright, *Conditions under which suboptimal nonlinear MPC is inherently robust,* In 18th IFAC World Congress, Milan, Italy, Sep. 2011.

[76] A.T. Patera, G. Rozza, *Reduced Basis Approximation and A Posteriori Error Estimation for Paramtrized Partial Differential Equations.* MIT Pappalardo Graduate Monographs in Mechanical Engineering, 2006.

[77] M. Pollatschek, B. Avi-Itzhak, *Algorithms for Stochastic Games with Geometrical Interpretation,* Management Sci **15** (1969), 399-415.

[78] L. Pontryagin, V. Boltyanskii, R. Gamkrelize, E. Mishenko, *The Mathematical Theory of Optimal Processes,* Wiley, 1962.

[79] M.L. Puterman, S.L. Brumelle, *On the convergence of Policy iteration in stationary Dynamic,* Porgramming. Math. of Operation Research, **4** (1979), 60-69.

[80] A. Quarteroni, *Modellistica Numerica per Problemi Differenziali,* Springer, Milano, 2008.

[81] M.L. Rapun, J.M. Vega. *Reduced order models based on local POD plus Galerkin projection.* J. Comput. Phys., **229** (2010), 3046-3063.

[82] J.B. Rawlings, D.Q. Mayne *Model Predictive Control: Theory and Design,* Nob Hill Publishing, LLC, 2009.

[83] M. Reed, B. Simon, *Methods of Modern Mathematical Physics I: Functional Analysis,* Academic Press, New York, 1980.

[84] T. Reis, T. Stykel, *A Survey on Model Reduction of Coupled Systems* W.H.A. Schilders, H. van der Vorst, J. Rommes (eds), *Model Order Reduction: Theory, Research Aspects and Applications,* Mathematics in Industry, **13**, Springer-Verlag, Berlin/Heidelberg, (2008), 133-155.

[85] G. Rozza, K. Veroy, *On the stability of reduced basis techniques for Stokes equations in parametrized domains.* Computer Methods in Applied Mechanics and Engineering, **196** 2007, 1244-1260.

[86] G. Rozza, D.B.P. Huynh, A.T. Patera, *Reduced basis approximation and a posteriori error estimation for affinely parametrized elliptic coercive partial differential equations* Archive of Computational Methods in Engineering, **15** (2008),229-275.

[87] E.W. Sachs, M. Schu, *A-priori error estimates for reduced order models in finance.*ESAIM: Mathematical Modelling and Numerical Analysis, **47** (2013), 449-469.

[88] S.Salsa, *Equazioni a derivate parziali*, Springer, 2004.

[89] M.S. Santos, *Numerical solution of dynamic economic models*, in Handbook of Macroeconomics, J.B. Taylor and M. Woodford eds., Elsevier Science, Amsterdam, The Netherlands, 2009, 311-386.

[90] M.S. Santos and J. Rust, *Convergence properties of policy iteration*, SIAM J. Control Optim., **42** (2004), 2094-2115.

[91] A. Seeck, *Iterative lösungen der Hamilton-Jacobi-Bellman-Gleichung bei unendlichen Zeithorizont*, Diplomarbeit, Universität Kiel, 1997.

[92] J.A. Sethian, *Level set methods and fast marching methods*, Cambridge University Press, 1999.

[93] J.A. Sethian, A. Vladimirsky, *Ordered upwind methods for static Hamilton-Jacobi equations: theory and algorithms*, SIAM J. Numer. Anal., **41** (2003), 325-363.

[94] J.R. Singler, *New POD expressions, error bounds, and asymptotic results for reduced order models of parabolic PDEs,* Preprint 2013, available on `http://web.mst.edu/ singlerj`

[95] L. Sirovich, *Turbulence and the dynamics of coherent structures. Parts I-II,* Quarterly of Applied Mathematics, **XVL** (1987), 561-590.

[96] F. Tröltzsch, *Optimal Control of Partial Differential Equations: Theory, Methods and applications,* Graduate Studies in Mathematics, **112**, American Mathematical Society, 2010.

[97] F. Tröltzsch. *Optimal Control of Partial Differential Equations: Theory, Methods and applications*, AMS, 2010.

[98] F. Tröltzsch, S. Volkwein, *POD a-posteriori error estimates for linear-quadratic optimal control problems,* Computational Optimization and Applications, **44** (2009), 83-115.

[99] S. Volkwein, *Lagrange-SQP techniques for the control constrained optimal boundary control for the Burgers equation,* Computational Optimization and Applications, **26** (2003), 253-284.

[100] S. Volkwein, *Model Reduction using Proper Orthogonal Decomposition: Theory and Reduced-Order Modelling,* Lecture Notes, Universität Konstanz, 2012. See `www.math.uni-konstanz.de/numerik/personen/volkwein/index.php`.

[101] K. Zhou, J.C Doyle and K. Glover, *Robust and Optimal Control.* Prentice Hall, Upper Saddle River, New Jersey, 1996.