Data-Driven Recovery of Optimal Feedback Laws through Optimality Conditions and Sparse Polynomial Regression

Behzad Azmi* Dante Kalise† Karl Kunisch‡

July 17, 2020

Abstract

A data-driven approach for the computation of high-dimensional optimal feedback laws arising in deterministic nonlinear control is proposed. The approach exploits the control-theoretical link between Hamilton-Jacobi-Bellman PDEs characterizing the value function of the optimal control problems, and first-order optimality conditions via Pontryagin's Maximum Principle. The latter is used as a representation formula to recover the value function and its gradient at arbitrary points in the space-time domain through the solution of a two-point boundary value problem. After generating a dataset consisting of different state-value pairs, a hyperbolic cross polynomial model for the value function is fitted using a LASSO regression. An extended set of low and high-dimensional numerical tests in nonlinear optimal control reveal that enriching the dataset with gradient information reduces the number of training samples, and that the sparse polynomial regression consistently yields a feedback law of lower complexity.

1 Introduction

A large class of design problems including the synthesis of autopilot and guidance systems, the computation of optimal investment strategies, and the control of fluid flow phenomena, among others, can be cast as optimal control problems. In this framework, we compute a time-dependent control signal $\mathbf{u}(t)$ by minimizing a performance index

$$J(\mathbf{u}; t_0, \mathbf{x}) := \int_{t_0}^{T} \ell(\mathbf{y}(t), \mathbf{u}(t))\, dt + \mathcal{F}(T, \mathbf{y}(T)) \tag{1}$$

subject to $\mathbf{y}(t)$ being a solution of the nonlinear dynamical system in $\mathbb{R}^n$

$$\frac{d}{dt}\mathbf{y}(t) = \mathbf{f}(\mathbf{y}(t), \mathbf{u}(t))\,, \qquad \mathbf{y}(t_0) = \mathbf{x}\,. \tag{2}$$

*RICAM, Austrian Academy of Sciences, Altenbergerstraße 69, A-4040 Linz, Austria. E-mail: behzad.azmi@ricam.oeaw.ac.at

†School of Mathematical Sciences, University of Nottingham, NG7 2QL, United Kingdom. E-mail: dante.kalise@nottingham.ac.uk

‡RICAM, Austrian Academy of Sciences and Department of Mathematics, University of Graz, Heinrichstraße 36, A-8010 Graz, Austria. E-mail:karl.kunisch@uni-graz.at

The numerical realization of control laws by solving the dynamic optimization problem above is a topic at the interface between control theory, computational optimization, and numerical analysis. While this problem dates back to the birth of Calculus of Variations, it was during the second half of the 20th century when two major methodological breakthroughs shaped our understanding of optimal control theory, namely, the development of Pontryagin's Maximum Principle and the theory of Dynamic Programming (see [66] for a historical survey on this topic). On the one hand, Pontryagin's Maximum Principle (PMP) [67] yields first-order optimality conditions for (1)-(2) in the form of a two-point boundary value problem for a forward-backward coupling between state, adjoint, and control variables, denoted by $(\mathbf{y}^*(t), \mathbf{p}^*(t), \mathbf{u}^*(t))$ respectively, which in short reads

$$\begin{cases} \frac{d}{dt}\mathbf{y}^*(t) = \mathbf{f}(\mathbf{y}^*(t), \mathbf{u}^*(t))\,, \\ -\frac{d}{dt}\mathbf{p}^*(t) = \partial_{\mathbf{y}}\mathbf{f}(\mathbf{y}^*(t), \mathbf{u}^*(t))\mathbf{p}^*(t) + \partial_{\mathbf{y}}\ell(\mathbf{y}^*(t), \mathbf{u}^*(t))\,, \\ \quad \mathbf{y}^*(t_0) = \mathbf{x}\,, \qquad \mathbf{p}^*(T) = \partial_{\mathbf{y}}\mathcal{F}(T, \mathbf{y}^*(T))\,, \\ \min\limits_{u} \ \{\langle \mathbf{f}(\mathbf{y}^*(t), \mathbf{u}), \mathbf{p}^*(t)\rangle + \ell(\mathbf{y}^*(t), \mathbf{u})\} \\ \quad = \langle \mathbf{f}(\mathbf{y}^*(t), \mathbf{u}^*(t)), \mathbf{p}^*(t)\rangle + \ell(\mathbf{y}^*(t), \mathbf{u}^*(t))\,, \qquad \forall t \in (t_0, T)\,. \end{cases} \tag{TPBVP}$$

This procedure yields an optimal state-adjoint-control triple originating from the initial condition $\mathbf{x}$, in what is known as an open-loop control. On the other hand, the Dynamic Programming approach synthesizes the optimal control as

$$\mathbf{u}^*(t, \mathbf{y}) = \underset{u}{argmin} \ \{\langle \mathbf{f}(\mathbf{y}, \mathbf{u}), \nabla V(t, \mathbf{y})\rangle + \ell(\mathbf{y}, \mathbf{u})\}\,, \tag{3}$$

where $V(x, t)$ is the value function of the problem

$$V(t, \mathbf{x}) := \inf_{\mathbf{u}(\cdot)} \{J(\mathbf{u}; t, \mathbf{x}) \text{ subject to } (2)\}\,, \tag{4}$$

which in turn satisfies a first-order, nonlinear Hamilton-Jacobi-Bellman (HJB) partial differential equation of the form

$$\begin{cases} \partial_t V(t, \mathbf{x}) + \min\limits_{u} \ \{\langle \mathbf{f}(\mathbf{y}, \mathbf{u}), \nabla V(t, \mathbf{x})\rangle + \ell(\mathbf{x}, \mathbf{u})\} = 0 \qquad \text{for } (t, \mathbf{x}) \in [0, T) \times \mathcal{X}, \\ V(T, \mathbf{x}) = \mathcal{F}(T, \mathbf{x}), \end{cases} \tag{HJB}$$

to be solved over the state space of the dynamics $\mathcal{X} \subset \mathbb{R}^n$ [12, Chapter 1]. This approach expresses the optimal control as a feedback map or closed-loop form, i.e. $\mathbf{u}^* = \mathbf{u}^*(t, \mathbf{y}(t))$, yielding a control law that is optimal in the whole state space. From a practical viewpoint, optimal trajectories obtained from the solution of (TPBVP) are not robust with respect to disturbances in the control loop, raising the case for a HJB-based feedback synthesis. However, the solution of HJB PDEs, especially for high-dimensional dynamical systems, comes at a formidable computational cost, often referred in the literature as the *curse of dimensionality* [18]. Over the last years, a number of works have reported remarkable progress in the solution of high-dimensional HJB PDEs, including the use of sparse grids [42, 20], tree structure algorithms [8], max-plus methods [62, 5], polynomial approximation [53, 54], tensor decomposition techniques [50, 71, 45, 39, 65], and an evergrowing literature on artificial neural networks [48, 37, 64, 51, 59].

On the link between PMP and HJB PDEs. We follow an alternative approach for the computation of optimal feedback laws that circumvents the direct solution of the HJB PDE by exploiting its links with the PMP. There exists an extensive literature dating back to [67, Chapter 1] discussing the relation between dynamic programming and first-order optimality conditions, and we refer the reader to [12, Section 3.4] for an exhaustive revision of the main results in this topic and to the work by Barron and Jensen [13], which is the first reference studying the link between PMP and viscosity solutions of first-order HJB PDEs. In the simplest version of the statement, assuming the solution of the HJB PDE is $\mathcal{C}^2$, it can be shown that the forward-backward dynamics originating from the PMP correspond to the characteristic curves of the HJB equation. Therefore, the value $V(t_0, \mathbf{x})$ can be obtained by solving the optimality system (TPBVP) with initial condition $\mathbf{y}(t_0) = \mathbf{x}$. Moreover, the initial adjoint variable corresponds to $\mathbf{p}^*(t_0) = \nabla V(t_0, \mathbf{x})$ [32, 24, 34]. The PMP can be thus interpreted as a representation formula for the value function and its gradient at a given space-time point. This idea constitutes the basis of our work.

A data-driven method for computing optimal feedback laws. We restrict our attention to a class of smooth and unconstrained nonlinear optimal control problems where the aforedescribed link between PMP and the HJB PDE is direct, and we use it to generate a characteristic-based, causality-free method to approximate $V(t, \mathbf{x})$ and as a by-product $\mathbf{u}(t, \mathbf{x})$, without solving (HJB). To do this, we sample a set of initial conditions $\{(t^i, \mathbf{x}^i)\}_{i=1}^N$, for which we compute both $V(t^i, \mathbf{x}^i)$ and $\nabla V(t^i, \mathbf{x}^i)$ by realizing the optimal trajectory through PMP. This is done by following a reduced gradient approach [49], in which forward-backward iterative solves of (TPBVP) are combined with a gradient descent method to find the minimizer of $J(\mathbf{u}; t_0, \mathbf{x})$. Having collected an enriched dataset $\{t^i, \mathbf{x}^i, V(t^i, \mathbf{x}^i), \nabla V(t^i, \mathbf{x}^i)\}_{i=1}^N$, we fit a polynomial model for the value function

$$V_\theta(t, \mathbf{x}) = \sum_{i=1}^{q} \theta_i \Phi_i(t, \mathbf{x}) = \langle \theta, \Phi \rangle \,, \tag{5}$$

with $\Phi(t, \mathbf{x}) = (\Phi_1(t, \mathbf{x}), \dots, \Phi_q(t, \mathbf{x}))$ are elements of a suitable polynomial basis, and the parameters $\theta = (\theta_1, \dots, \theta_q)$ obtained from a LASSO regression

$$\min_{\theta \in \mathbb{R}^q} \| [\Phi; \nabla \Phi] \, \theta - [V; \nabla V] \|_2^2 + \lambda \|\theta\|_{1,\mathbf{w}} \,, \tag{6}$$

where the matrix $[\Phi; \nabla \Phi] \in \mathbb{R}^{(n+1)N \times q}$ and the vector $[V; \nabla V] \in \mathbb{R}^{(n+1)N}$ include value function and gradient data. Finally, the optimal feedback map is recovered as

$$\mathbf{u}^*(t, \mathbf{x}) = \underset{u}{\operatorname{argmin}} \;\; \{ \langle \mathbf{f}(\mathbf{x}, \mathbf{u}), \nabla V_\theta(t, \mathbf{x}) \rangle + \ell(\mathbf{x}, \mathbf{u}) \} \,. \tag{7}$$

Related literature and contributions. The idea of using open-loop solves to build a numerical representation of an optimal feedback law dates back at least to [17, 52], where a low-dimensional feedback law was constructed directly by interpolating the optimal control from a set of collocation points. More recently, this idea has been revisited and improved in [57, 56] by exploiting the extremely parallelizable structure of the open-loop solves together with a sparse grid interpolant, scaling up to 6-dimensional examples. In [63, 55] this idea has been further developed by replacing the use of a grid-based interpolant with artificial neural networks which

are trained using a dataset consisting of both the value function and gradient evaluations, presenting computational results up to dimension 30. In a similar vein, the works [31, 30, 29, 38] have proposed the use of representation formulas for HJB PDEs, ranging from the celebrated Lax-Hopf formula to variations of the PMP, in conjunction with efficient convex optimization techniques for the solution of point values of the HJB PDE on the fly. These results have been further studied in [73], and used in [74, 47] for the construction of control Lyapunov functions both with sparse grids and neural networks. Our work, while in line with the aforedescribed, proposes a different methodology which is summarized in the following ingredients:

- An enriched dataset containing both value function and gradient information, where the open-loop solves are realized through a reduced gradient approach, which is well suited for high-dimensional nonlinear optimal control problems.

- The value function, and as a consequence the feedback law, is approximated with a polynomial ansatz (5). This choice is backed by the extensive literature concerning power series approximations of the value function [7, 58, 61, 23]. In fact, it is well-known that for linear dynamics and quadratic cost functions, the value function corresponds to a quadratic form, which is contained in the span of our approximation space. In [53, 54], we have studied a Galerkin approach for HJB PDEs arising in nonlinear control and games with polynomial approximation functions. In these works we have constructed a polynomial basis limited by the total degree of the monomials, solving up to 14-dimensional tests. Here instead, borrowing a leaf from the vast literature on polynomial approximation theory [16, 27, 33, 26, 4], we consider a hierarchical basis defined through hyperbolic cross approximation, for which we report tests up to dimension 80 at moderate computational cost.

- The use of a polynomial expansion for the value function, which is linear in the coefficients, allows us to fit the value function model through a LASSO regression framework (6). This least squares problem can easily account for the use of gradient data and has a well-understood numerical realization [4, 60]. Moreover, we include an ℓ_1 penalty on the expansion coefficients, leading to the synthesis of low complexity feedback laws (7), which is crucial for a fast online computation of the feedback action.

- Our extensive numerical assessment of the methodology includes a class of genuinely nonlinear, fully coupled, high-dimensional dynamics arising in agent-based modelling [28], which ultimately connects with the control of non-local transport equations arising in mean field control [6, 40, 44].

The rest of the paper is structured as follows. In Section 2 we present the specific settings of the finite horizon, nonlinear optimal control problem under study. Section 3 describes our numerical methodology, including the numerical generation of the dataset, the polynomial ansatz for the value function and the model fit through LASSO regression. Finally, Section 4 presents an exhaustive numerical assessment of the proposed methodology, including the synthesis of high-dimensional optimal feedback laws for nonlinear PDEs and multiagent systems.

2 Formulating the Optimal Control Problem

We briefly describe the control setting used throughout the rest of the paper. Without loss of generality, we will be concerned with an unconstrained, finite-horizon, nonlinear optimal control without terminal penalty

$$\min_{u(\cdot)\in L^2(t_0,T;\mathbb{R}^m)} J(\mathbf{u};t_0,\mathbf{x}) := \int_{t_0}^{T} \ell(\mathbf{y}(t)) + \beta\|\mathbf{u}(t))\|_2^2\, dt\,, \qquad \beta > 0\,, \tag{8}$$

subject to $\mathbf{y}(t)$ being the solution to the control-affine nonlinear dynamics

$$\frac{d}{dt}\mathbf{y}(t) = \mathbf{f}(\mathbf{y}(t)) + \mathbf{g}(\mathbf{y}(t))\mathbf{u}(t)\,, \qquad \mathbf{y}(t_0) = \mathbf{x}\,, \tag{9}$$

where $\mathbf{y}(t) \in \mathbb{R}^n$ the state and $\mathbf{u}(t) \in \mathbb{R}^m$ the control variable. We assume that the running cost $\ell : \mathbb{R}^n \to \mathbb{R}$, the dynamics $\mathbf{f} : \mathbb{R}^n \to \mathbb{R}^n$, and $\mathbf{g} : \mathbb{R}^n \to \mathbb{R}^{n\times m}$, are continuously differentiable. Under these assumptions, the optimal control is characterized by a triple $(\mathbf{y}^*(t), \mathbf{p}^*(t), \mathbf{u}^*(t))$ which satisfies the state equation (9), coupled with the backward adjoint system for $\mathbf{p}^* = (p_1^*, \ldots, p_n^*)$ given by

$$-p_i^*(t) = \sum_{j=1}^{n} p_j^*(t)\left(\partial_{y_i}(f_j(\mathbf{y}^*(t)) + g_j(\mathbf{y}^*(t))\mathbf{u}^*(t)\right) + \partial_{y_i}\ell(\mathbf{y}^*(t))\,, \qquad i = 1,\ldots,n\,, \tag{10}$$

$$p_i^*(T) = 0\,, \tag{11}$$

and the optimality condition

$$\mathbf{u}^*(t) = -\frac{1}{2\beta}\mathbf{g}^t(\mathbf{y}^*(t))\mathbf{p}^*(t)\,, \qquad \forall t \in (t_0, T)\,. \tag{12}$$

Note that the latter equation is an instance of the optimality condition formally stated in (TPBVP), in the case of an unconstrained, control-affine problem with quadratic control penality. As it was discussed in the previous section, our interest is to make use of the numerical solution of open-loop optimal controls satisfying the two-point boundary value problem (9)-(12), to recover an optimal feedback law globally defined. The latter can be computed through dynamic programming arguments. Defining the value function $V(t,\mathbf{x}) : [0,T]\times\mathbb{R}^n \longrightarrow \mathbb{R}$ as

$$V(t,\mathbf{x}) := \inf_{\mathbf{u}(\cdot)}\{J(\mathbf{u};t,\mathbf{x}) \text{ subject to } (9)\}\,, \tag{13}$$

it is well-known that it corresponds to the unique viscosity solution to the Hamilton-Jacobi-Bellman PDE

$$\begin{cases} \partial_t V(t,\mathbf{x}) + \mathcal{H}(\mathbf{x}, \nabla V(t,\mathbf{x})) = 0 & \text{for } t \in (0,T), \\ V(T,\mathbf{x}) = 0, \end{cases} \tag{14}$$

where for every $(\mathbf{x},\mathbf{p}) \in \mathbb{R}^n \times \mathbb{R}^n$, the Hamiltonian $\mathcal{H}$ is defined by

$$\mathcal{H}(\mathbf{x},\mathbf{p}) := \min_{u\in R^m}\left\{\ell(\mathbf{x}) + \beta\|\mathbf{u}\|_2^2 + \mathbf{p}^t\left(\mathbf{f}(\mathbf{x}) + \mathbf{g}(\mathbf{x})\mathbf{u}\right)\right\}\,. \tag{15}$$

After solving this equation, the optimal feedback map is given by

$$u^*(t,\mathbf{x}) = \underset{u\in R^m}{argmin}\left\{\beta\|\mathbf{u}\|_2^2 + \nabla V(t,\mathbf{x})^t\left(\mathbf{g}(\mathbf{x})\mathbf{u}\right)\right\} = -\frac{1}{2\beta}\mathbf{g}^t(\mathbf{x})\nabla V(t,x). \tag{16}$$

Note that it is possible to combine (14) with (16) to obtain the HJB PDE

$$\partial_t V(t,\mathbf{x}) - \frac{1}{2\beta}\nabla V(t,\mathbf{x})^t \mathbf{g}(\mathbf{x})\mathbf{g}^t(\mathbf{x})\nabla V(t,\mathbf{x}) + \nabla V(t,\mathbf{x})^t \mathbf{f}(\mathbf{x}) + \ell(\mathbf{x}) = 0\,, \tag{17}$$

$$V(T,\mathbf{x}) = 0\,. \tag{18}$$

The numerical approximation of optimal feedback laws by solving the HJB equation above poses an overwhelmingly complex computational challenge, as it requires the solution of an $n-$dimensional nonlinear PDE. In this work, we circumvent this difficulty by interpreting system (9)-(12) as a representation formula for the solution of (17). More precisely, the computation of a given $V(t^i,\mathbf{x}^i)$ can be realized by solving (9)-(12) setting $t_0 = t_i$ and the initial condition $\mathbf{y}(t_0) = \mathbf{x}^i$, and evaluating the optimal cost (13) using the optimal triple $(\mathbf{y}^*(t), \mathbf{p}^*(t), \mathbf{u}^*(t))$. Moreover, the optimal adjoint verifies $\mathbf{p}^*(t) = \nabla V(t,\mathbf{y}^*(t))$. Therefore, the solution of (9)-(12) is a representation formula for $V(t,\mathbf{y}^*(t))$ and $\nabla V(t,\mathbf{y}^*(t))$ along the optimal state trajectory $\mathbf{y}^*(t)$ for $t \in (t_0, T)$.

Stabilization with static feedback laws. For large optimization horizon T, and $\ell(\mathbf{y}) = \|\mathbf{y}\|^2$, the cost (8) can be considered as an approximation to the asymptotic stabilization problem were $T = \infty$. This scenario will be the focus of our numerical tests. This also motivates that in the rest of the paper we will restrict our presentation to the approximation of $V(0,\mathbf{x}) = V(\mathbf{x})$ and $\nabla V(0,\mathbf{x}) = \nabla V(\mathbf{x})$, and to the associated static feedback law $u^*(0,\mathbf{x}) = u^*(\mathbf{x})$. This approximation also relates to the one done in nonlinear model predictive control, where after an open-loop solve is computed, the initial optimal control $u^*(0)$ is used to evolve the state equation for a short period , after which the open-loop optimization is re-computed with an updated initial state. It can be shown that as the prediction horizon T increases, the optimal control approaches the stationary feedback laws, see for instance [9, 46, 69, 72]. For the reader interested in obtaining the complete time-dependent optimal feedback law, we discuss at the end of Section 3 how to extend the proposed methodology.

3 Data-driven Recovery of Feedback Laws

In this section, we develop the different building blocks of the proposed approach. We first discuss how to generate the training dataset by solving a set of open-loop optimal control problems with a reduced gradient approach. Next, we build a polynomial model for the value function based on a hyperbolic cross approximation. Having set both the data and the model, we fit our model with a LASSO regression. At the end of the section, we explain how to modify the proposed framework to recover time-dependent feedback laws.

3.1 Generating a dataset with a reduced gradient approach

We begin by generating a dataset $\{\mathbf{x}^j, V(0,\mathbf{x}^j), \nabla V(0,\mathbf{x}^j)\}_{j=1}^N$ which is obtained from the solving open-loop optimal control problems of the form (8) (with $t_0 = 0$ and T sufficiently large) through the use of first-order optimality conditions (9)-(12). For this purpose we follow a reduced gradient approach with a Barzilai-Borwein update [14, 10], which is summarized as follows.

Assuming that the solution operator $\mathbf{y}(\mathbf{u}) = \mathbf{y}(\mathbf{u},\mathbf{x})$ corresponding to the state equation (9) is well-defined and continuously differentiable, we can rewrite (8)-(9) as the following unconstrained

dynamic optimization problem depending solely on the control variable $\mathbf{u}$

$$\min_{\mathbf{u}(\cdot)} \mathcal{J}(\mathbf{u}) = \min_{\mathbf{u}(\cdot)} J(\mathbf{y}(\mathbf{u}), \mathbf{u}) = \min_{(\mathbf{y}(\cdot),\mathbf{u}(\cdot))} \{J(\mathbf{y}, \mathbf{u}) : \text{ subject to } e(\mathbf{y}, \mathbf{u}) = 0\}, \tag{19}$$

where

$$e(\mathbf{y}, \mathbf{u}) := \begin{pmatrix} \frac{d}{dt}\mathbf{y}(t) - (\mathbf{f}(\mathbf{y}(t)) + \mathbf{g}(\mathbf{y}(t))\mathbf{u}(t)) \\ \mathbf{y}(0) - \mathbf{x} \end{pmatrix}. \tag{20}$$

Formally, we obtain the directional derivative of $\mathcal{J}$ at $\bar{\mathbf{u}} \in L^2(0, T; \mathbb{R}^m)$ in a direction $\delta\mathbf{u} \in L^2(0, T; \mathbb{R}^m)$ by computing

$$\mathcal{J}'(\bar{\mathbf{u}})\delta\mathbf{u} = (\mathcal{G}(\bar{\mathbf{u}}), \delta\mathbf{u}) = ((\mathbf{y}'(\bar{\mathbf{u}}))^* \partial_{\mathbf{y}} J(\bar{\mathbf{v}}) + \partial_{\mathbf{u}} J(\bar{\mathbf{v}}), \delta\mathbf{u}), \tag{21}$$

where $\bar{\mathbf{v}} := (\bar{\mathbf{y}}, \bar{\mathbf{u}})$ with $\bar{\mathbf{y}} := \mathbf{y}(\bar{\mathbf{u}})$, $\mathcal{G}$ denotes the gradient of $\mathcal{J}$, $(\cdot, \cdot)$ stands for the scalar product in the space of controls $L^2(0, T; \mathbb{R}^m)$, and the superscript $*$ corresponds to the adjoint operator. Moreover, the term $\mathbf{y}'(\bar{\mathbf{u}})$ is given by

$$\mathbf{y}'(\bar{\mathbf{u}})\delta\mathbf{u} = -(\partial_{\mathbf{y}} e(\bar{\mathbf{v}}))^{-1} \partial_{\mathbf{u}} e(\bar{\mathbf{v}})\delta\mathbf{u}. \tag{22}$$

It can be shown that $(\partial_{\mathbf{y}} e(\mathbf{y}(\mathbf{u}), \mathbf{u}))^{-1}$, defined by $(\phi, \mathbf{q}_0) \mapsto \mathbf{q}$, is the solution operator of the following linearised equation

$$\begin{cases} \frac{d}{dt}\mathbf{q}(t) - \partial_{\mathbf{y}} \left(\mathbf{f}(\mathbf{y}(t)) + \mathbf{g}(\mathbf{y}(t))\mathbf{u}(t)\right) \mathbf{q}(t) = \phi, \\ \mathbf{q}(0) = \mathbf{q}_0, \end{cases} \tag{23}$$

and that its the adjoint operator $(\partial_{\mathbf{y}} e(\mathbf{y}(\mathbf{u}), \mathbf{u}))^{-*}$ defined by $(\psi, \mathbf{p}_T) \mapsto \mathbf{p}$, is the solution operator to the following backward-in-time equation

$$\begin{cases} -\frac{d}{dt}\mathbf{p}(t) - \left(\partial_{\mathbf{y}} \left(\mathbf{f}(\mathbf{y}(t)) + \mathbf{g}(\mathbf{y}(t))\mathbf{u}(t)\right)^t \mathbf{p}(t) = \psi, \\ \mathbf{p}(T) = \mathbf{p}_T \,. \end{cases} \tag{24}$$

Putting these elements together, we are now in a position to compute the gradient $\mathcal{G}$ of $\mathcal{J}$ at $\bar{\mathbf{u}}$. Using (21) and (22), we obtain

$$\mathcal{G}(\bar{\mathbf{u}}) = \partial_{\mathbf{u}} J(\bar{\mathbf{v}}) - (\partial_{\mathbf{u}} e(\bar{\mathbf{v}}))^* (\partial_{\mathbf{y}} e(\bar{\mathbf{v}}))^{-*} \partial_{\mathbf{y}} J(\bar{\mathbf{v}}) \,, \tag{25}$$

and therefore, for almost every $t \in (0, T)$, we have

$$\mathcal{G}(\bar{\mathbf{u}})(t) = \mathbf{g}(\bar{\mathbf{y}}(t))^t \bar{\mathbf{p}}(t) + 2\beta\bar{\mathbf{u}}(t), \tag{26}$$

where $\bar{\mathbf{p}}$ is the solution to

$$\begin{cases} -\frac{d}{dt}\mathbf{p}(t) - \left(\partial_{\mathbf{y}} \left(\mathbf{f}(\bar{\mathbf{y}}(t)) + \mathbf{g}(\bar{\mathbf{y}}(t))\bar{\mathbf{u}}(t)\right)^t \mathbf{p}(t) = \partial_{\mathbf{y}} \ell(\bar{\mathbf{y}}(t)), \\ \mathbf{p}(T) = 0. \end{cases} \tag{27}$$

Having a realization of the reduced gradient, we follow the Barzilai-Borwein gradient method for finding the stationary point $\mathbf{u}^*$ of $\mathcal{J}$ (i.e., $\mathcal{G}(\mathbf{u}^*) = 0$). In this method, the stepsizes are chosen according to be either

$$\alpha_k^{BB1} := \frac{(\mathcal{S}_{k-1}, \mathcal{Y}_{k-1})}{(\mathcal{S}_{k-1}, \mathcal{S}_{k-1})}, \quad \text{or} \quad \alpha_k^{BB2} := \frac{(\mathcal{Y}_{k-1}, \mathcal{Y}_{k-1})}{(\mathcal{S}_{k-1}, \mathcal{Y}_{k-1})}, \tag{28}$$

Algorithm 1 Barzilai-Borwein two-point step-size gradient method

Input: Choose $\mathbf{u}_{-1} := 0$ and $\mathbf{u}_0 := -\mathcal{G}(0)$, tolerance $tol > 0$.

1: Set $k = 0$.
2: **while** $\|\mathcal{G}_k\| \geq tol$, **do**
3: Compute $\mathbf{y}_k(\mathbf{u}_k)$ via (9).
4: Compute $\mathbf{p}_k(\mathbf{y}_k, \mathbf{u}_k)$ via (27).
5: Compute $\mathcal{G}_k = \mathcal{G}(\mathbf{u}_k)$ using (26) with $(\mathbf{y}_k, \mathbf{p}_k, \mathbf{u}_k)$.
6: Choose

$$\alpha_k = \begin{cases} \alpha_k^{BB1} & \text{for odd } k, \\ \alpha_k^{BB2} & \text{for even } k. \end{cases}$$

7: Set $d_k = \frac{1}{\alpha_k}\mathcal{G}_k$.
8: Compute the step-size $\eta_k > 0$ based on the non-monotone linesearch given in [36].
9: Set $\mathbf{u}_{k+1} = \mathbf{u}_k - \eta_k d_k$, $k = k + 1$, and go to Step 2.

where $\mathcal{G}_k := \mathcal{G}(\mathbf{u}_k)$, $\mathcal{S}_{k-1} := \mathbf{u}_k - \mathbf{u}_{k-1}$ and $\mathcal{Y}_{k-1} := \mathcal{G}_k - \mathcal{G}_{k-1}$. With these specifications, we introduce Algorithm 1, which is used for solving the open-loop problems. Note that the formulas above are written for continuous-time dynamical systems. In practice, Algorithm 1 is expected to be used in conjunction with a suitable numerical integrator for an accurate approximation of both the state and its adjoint. Finally, fixing an initial condition $\mathbf{x}^j$, after Algorithm 1 has converged to $\mathbf{u}^*$ the dataset is completed with $V(\mathbf{x}^j) = \mathcal{J}(\mathbf{u}^*)$ and $\nabla V(\mathbf{x}^j) = \mathbf{p}^*(0)$.

3.2 Building a polynomial model for the value function

Having generated a dataset for recovering the value function associated to the optimal control problem, we now turn our attention to deriving a suitable model for regression. Our approximation of the static value function $V(\mathbf{x}) : \mathbb{R}^n \to \mathbb{R}$ follows the ideas presented in [4, 1].

Let $\mathcal{D} \subset \mathbb{R}^n$ be a bounded domain and $\{\Phi_\mathbf{i}\}_{\mathbf{i}\in\mathbb{N}_0^n}$ be a tensor-product orthonormal basis of $L^2(\mathcal{D})$. We consider basis which are typically polynomial, using for instance Legendre or Chebyshev polynomials. Concretely, assume that $\mathcal{D} := (-1, 1)^n$ and that $\{\phi_i\}_{i=0}^\infty$ is one-dimensional orthonormal basis of $L^2(-1, 1)$. Then, the corresponding tensor-product basis of $L^2(\mathcal{D})$ is defined by

$$\Phi_\mathbf{i}(\mathbf{x}) := \prod_{j=1}^{n} \phi_{i_j}(x_i)\,, \quad \text{with} \quad \mathbf{i} = (i_1, i_2, \ldots, i_n) \in \mathbb{N}^n, \quad \mathbf{x} = (x_1, x_2, \ldots, x_n), \tag{29}$$

where $\mathbb{N}_0 := \mathbb{N} \cup \{0\}$. Assuming that $V(\mathbf{x}) \in L^2(\mathcal{D}) \cap L^\infty(\mathcal{D})$, we can write

$$V(\mathbf{x}) = \sum_{\mathbf{i}\in\mathbb{N}_0^n} \theta_\mathbf{i}\Phi_\mathbf{i}\,, \tag{30}$$

with $\theta_\mathbf{i} = (V(\mathbf{x}), \Phi_\mathbf{i})_{L^2(\mathcal{D})}$ for every $\mathbf{i} \in \mathbb{N}_0^n$. We approximate this object by considering a truncated basis $\{\Phi_\mathbf{i}\}_{\mathbf{i}\in\mathfrak{I}}$ with a finite multi-index set $\mathfrak{I} \subset \mathbb{N}_0^n$ with cardinality $|\mathfrak{I}| = q < \infty$. Hence, we write

$$V(\mathbf{x}) = V_\theta^\mathfrak{I} + e_\mathfrak{I} = \sum_{\mathbf{i}\in\mathfrak{I}} \theta_\mathbf{i}\Phi_\mathbf{i} + \sum_{\mathbf{i}\notin\mathfrak{I}} \theta_\mathbf{i}\Phi_\mathbf{i} \tag{31}$$

with $\{\theta_{\mathbf{i}}\}_{\mathbf{i}\in\mathbb{N}_0^n} \in \ell^2(\mathbb{N}_0^n)$. In this work we are particularly interested in the case where $\mathcal{D}$ is a high-dimensional space and computing $V(\mathbf{x})$ by solving a HJB PDE is not a feasible alternative. Therefore, the selection of a basis $\mathfrak{I}$ whose cardinality scales reasonably well in high-dimensions, while maintaining an acceptable level of accuracy, is a fundamental criterion in our model selection. Figure 1 illustrates some of the typical options for generating a multi-dimensional polynomial basis. Generating a basis by directly taking the tensor product of polynomials up to a certain degree s leads to

$$\mathfrak{I}_{TP}(s) = \{\mathbf{i} = (i_1, i_2, \ldots, i_n) \in \mathbb{N}_0^n : \|\mathbf{i}\|_\infty \leq s\} \,, \tag{32}$$

with $|\mathfrak{I}_{TP}(s)| = (s+1)^n$, scaling exponentially in the dimension, limiting its applicability to $n \leq 5$ unless additional low-rank structures are assumed [39]. This exponential increase in the dimension can be mitigated by considering a total degree truncation

$$\mathfrak{I}_{TD}(s) = \{\mathbf{i} = (i_1, i_2, \ldots, i_n) \in \mathbb{N}_0^n : \|\mathbf{i}\|_1 \leq s\} \,, \tag{33}$$

with cardinality

$$|\mathfrak{I}_{TD}(s)| = \sum_{j=1}^{s} \binom{n+j-1}{j} . \tag{34}$$

This combinatorial dependence on the dimension allows to solve moderately high-dimensional problems, in our experience for $n \leq 15$ [53, 54]. In this work, we opt for a bases constructed with the *hyperbolic cross* index set, defined as

$$\mathfrak{I} = \mathfrak{I}(s) = \left\{ \mathbf{i} = (i_1, i_2, \ldots, i_n) \in \mathbb{N}_0^n : \prod_{j=1}^{n} (i_j + 1) \leq s + 1 \right\} . \tag{35}$$

While there are no explicit formulas for the cardinality of $\mathfrak{I}$, different upper bounds [2] such as

$$|\mathfrak{I}(s)| \leq \min\left\{ 2s^3 4^n, e^2 s^{2+\log_2(n)} \right\} \,, \tag{36}$$

indicate that it scales reasonably well for high-dimensional problems. For reference, in this paper we report results up to $n = 80$ at moderate computational cost. With 80 dimensions and degree 4, the tensor product basis would contain, 8.27×10^{55} elements, the total degree basis 1.93×10^6 elements, while the upper bound above for the hyperbolic cross above is 7.56×10^5. Besides the dimensionality argument, the hyperbolic cross set is also an adequate basis regarding best approximation properties [4] in conjunction with the ℓ_1 regression framework we will discuss in the following section. For the rest of the paper, we will adopt the notation $\mathbf{i}_1, \mathbf{i}_2, \ldots, \mathbf{i}_q(s)$ for an order of multi-indices in $\mathfrak{I}(s)$, and we write $\theta_{\mathfrak{I}} = \{\theta_{\mathbf{i}}\}_{\mathbf{i}\in\mathfrak{I}} = \{\theta_{\mathbf{i}_k}\}_{k=1}^q$.

3.3 Gradient-augmented regression

The last building block of our approach consists in fitting the polynomial expansion presented above with the dataset containing information of both the value function and its gradient. We first present an un-augmented linear least squares approach which will be used for numerical comparison. Based on the data generation procedure presented in section 3.1, we assume the existence of a dataset consisting of N samples

$$D = \left\{ \mathbf{x}^j, V^j \right\}_{j=1}^N , \tag{37}$$

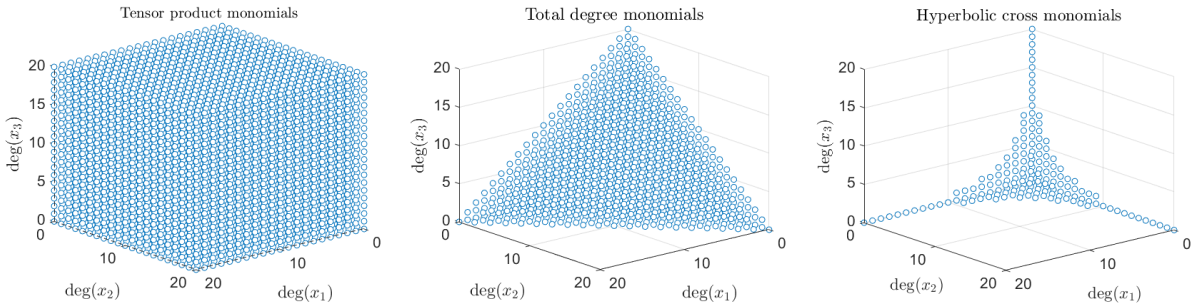


Figure 1: Alternatives for generating a high-dimensional polynomial basis. From left to right: direct tensorization of a 1-dimensional basis, truncation by total degree, hyperbolic cross approximation.

where $V^j := V(\mathbf{x}^j)$. By defining $\mathbf{V} \in \mathbb{R}^N$, $\mathbf{A} \in \mathbb{R}^{N\times q}$, and $\mathbf{e} \in \mathbb{R}^N$ by

$$\mathbf{V} := \frac{1}{\sqrt{N}} \left(V(\mathbf{x}^j)\right)_{j=1}^N, \quad \mathbf{A} := \frac{1}{\sqrt{N}} \left(\Phi_{\mathbf{i}_k}(\mathbf{x}^j)\right)_{j,k=1}^{N,q}, \quad \text{and } \mathbf{e} := \frac{1}{\sqrt{N}} \left(e_{\mathfrak{I}}(\mathbf{x}^j)\right)_{j=1}^N, \tag{38}$$

we write the following linear system to be satisfied by our model

$$\mathbf{V} = \left[V_\theta(\mathbf{x}^i)\right]_{i=1}^N = \mathbf{A}\theta_{\mathfrak{I}} + \mathbf{e}, \tag{39}$$

from which $V(\mathbf{x})$ can be approximated on the subspace $\{\text{span}(\{\Phi_{\mathbf{i}}\}_{\mathbf{i}\in\mathfrak{I}})\}$, by solving the linear least squares problem

$$\min_{\theta\in\mathbb{R}^q} \|\mathbf{A}\theta - \mathbf{V}\|_2^2. \tag{P_{ℓ_2}}$$

In order to enhance sparsity in the vector of coefficients, we resort to compressed sensing techniques and we consider the *weighted LASSO* regression

$$\min_{\theta\in\mathbb{R}^q} \|\mathbf{A}\theta - \mathbf{V}\|_2^2 + \lambda\|\theta\|_{1,\mathbf{w}}, \tag{P_{ℓ_1}}$$

where $\lambda > 0$, and the weights $\mathbf{w} := \{w_{\mathbf{i}}\}_{\mathbf{i}\in\mathbb{N}_0^n}$ with $w_{\mathbf{i}} \geq 1$ define the $\|\cdot\|_{1,\mathbf{w}}$ norm as

$$\|\theta\|_{1,\mathbf{w}} = \sum_{\mathbf{i}\in\mathbb{N}_0^n} w_{\mathbf{i}}|\theta_{\mathbf{i}}|. \tag{40}$$

Denoting by $\theta_{\ell_2} \in \mathbb{R}^q$ and $\theta_{\ell_1} \in \mathbb{R}^q$ the solutions to problems (P_{ℓ_2}) and (P_{ℓ_1}), respectively, we recover the following approximations of $V(\mathbf{x})$

$$V_{\ell_2}(\mathbf{x}) = \sum_{k=1}^{q} (\theta_{\ell_2})_{\mathbf{i}_k}\Phi_{\mathbf{i}}(\mathbf{x}) = \sum_{\mathbf{i}\in\mathfrak{I}} (\theta_{\ell_2})_{\mathbf{i}}\Phi_{\mathbf{i}}(\mathbf{x}) \quad \text{and} \quad V_{\ell_1} = \sum_{\mathbf{i}\in\mathfrak{I}} (\theta_{\ell_1})_{\mathbf{i}}\Phi_{\mathbf{i}}(\mathbf{x}). \tag{41}$$

Building on the fact that our data generation procedure retrieves the value function and its gradient, we recast the regression problems by using the augmented data [4]. We assume $V(\mathbf{x}) \in H^1(\mathcal{D})$. For the augmented dataset

$$D_{aug} = \left\{\mathbf{x}^j, V^j, V_x^j\right\}_{j=1}^N, \quad \text{with } V_x^j = \left(\frac{\partial V_T}{\partial x_1}(\mathbf{x}^j), \frac{\partial V_T}{\partial x_1}(\mathbf{x}^j), \ldots, \frac{\partial V_T}{\partial x_n}(\mathbf{x}^j)\right)^t \text{ for } j = 1, \ldots, N, \tag{42}$$

and, for every $m = 0, \dots, n$, we define

$$\mathbf{A_m} := \frac{1}{\sqrt{N}} \left(\frac{\partial \Phi_{\mathbf{i}_k}}{\partial x_m}(\mathbf{x}^j) \right)_{j,k=1}^{N,q}, \quad \mathbf{V}_m := \frac{1}{\sqrt{N}} \left(\frac{\partial V_T}{\partial x_m}(\mathbf{x}^j) \right)_{j=1}^{N}, \text{ and } \mathbf{e}_m := \frac{1}{\sqrt{N}} \left(\frac{\partial e_{\mathfrak{I}}}{\partial x_m}(\mathbf{x}^j) \right)_{j=1}^{N}, \tag{43}$$

where

$$\mathbf{A_0} := \frac{1}{\sqrt{N}} \left(\Phi_{\mathbf{i}_k}(\mathbf{x}^j) \right)_{j,k=1}^{N,q}, \quad \mathbf{V}_0 := \frac{1}{\sqrt{N}} \left(V_T(\mathbf{x}^j) \right)_{j=1}^{N}, \text{ and } \mathbf{e}_0 := \frac{1}{\sqrt{N}} \left(e_{\mathfrak{I}}(\mathbf{x}^j) \right)_{j=1}^{N}. \tag{44}$$

Then by assembling

$$\bar{\mathbf{A}} := \begin{pmatrix} \mathbf{A_0} \\ \mathbf{A_1} \\ \vdots \\ \mathbf{A_n} \end{pmatrix}, \quad \bar{\mathbf{V}} := \begin{pmatrix} \mathbf{V_0} \\ \mathbf{V_1} \\ \vdots \\ \mathbf{V_n} \end{pmatrix}, \text{ and } \bar{\mathbf{e}} := \begin{pmatrix} \mathbf{e_0} \\ \mathbf{e_1} \\ \vdots \\ \mathbf{e_n} \end{pmatrix}, \tag{45}$$

we obtain the following system of linear equations

$$\bar{\mathbf{V}} = \bar{\mathbf{A}}\theta_{\mathfrak{I}} + \bar{\mathbf{e}}\,, \tag{46}$$

for which we formulate the augmented-gradient optimization problems

$$\min_{\theta \in \mathbb{R}^q} \|\bar{\mathbf{A}}\theta - \bar{\mathbf{V}}\|_2^2, \tag{AP_{ℓ_2}}$$

and

$$\min_{\theta \in \mathbb{R}^q} \|\bar{\mathbf{A}}\theta - \bar{\mathbf{V}}\|_2^2 + \lambda \|\theta\|_{1,\mathbf{w}}. \tag{AP_{ℓ_1}}$$

Denoting by $\bar{\theta}_{\ell_2} \in \mathbb{R}^q$ and $\bar{\theta}_{\ell_1} \in \mathbb{R}^q$ the solutions to (AP_{ℓ_2}) and (AP_{ℓ_2}), respectively, we recover the following gradient-augmented approximations of $V(\mathbf{x})$

$$\bar{V}_{\ell_2}(\mathbf{x}) = \sum_{\mathbf{i} \in \mathfrak{I}} (\bar{\theta}_{\ell_2})_{\mathbf{i}} \Phi_{\mathbf{i}}(\mathbf{x}) \quad \text{and} \quad \bar{V}_{\ell_1}(\mathbf{x}) = \sum_{\mathbf{i} \in \mathfrak{I}} (\bar{\theta}_{\ell_1})_{\mathbf{i}} \Phi_{\mathbf{i}}(\mathbf{x}). \tag{47}$$

Similarly, we obtain the following representations for $\nabla_x V(\mathbf{x})$, where $\nabla_x = (\frac{\partial}{\partial x_1}, \dots, \frac{\partial}{\partial x_n})^t$

$$\nabla_x \bar{V}_{\ell_2}(\mathbf{x}) = \sum_{\mathbf{i} \in \mathfrak{I}} (\bar{\theta}_{\ell_2})_{\mathbf{i}} \nabla_x \Phi_{\mathbf{i}}(\mathbf{x}) \quad \text{and} \quad \nabla_x \bar{V}_{\ell_1}(\mathbf{x}) = \sum_{\mathbf{i} \in \mathfrak{I}} (\bar{\theta}_{\ell_1})_{\mathbf{i}} \nabla_x \Phi_{\mathbf{i}}(\mathbf{x}) \tag{48}$$

recovering the optimal feedback laws

$$\mathbf{u}_\star^*(\mathbf{x}) = -\frac{1}{2\beta} \mathbf{g}^t(\mathbf{x}) \nabla_x \bar{V}_\star(\mathbf{x})\,, \qquad \star \in \{\ell_1, \ell_2\}\,. \tag{49}$$

On the numerical realization of the weighted LASSO regression. The linear least squares problems (P_{ℓ_2}) and (AP_{ℓ_2}) can be efficiently solved by using a preconditioned conjugate gradient method. The formulations (P_{ℓ_1}) and (AP_{ℓ_1}) are instead convex, nonsmooth optimization problem which require a more elaborate treatment. In this work, we compute the solution of (P_{ℓ_1}) and (AP_{ℓ_1}) by means of the Alternating Direction Method of Multipliers (ADMM) [22]. To make matters precise, Algorithm 2 presents its implementation for problem (P_{ℓ_1}). In this

Algorithm 2 ADMM for solving weighted LASSO

Input: Choose $\theta^0, z^0, h^0 \in \mathbb{R}^q$, $\rho > 0$, and tolerance $tol > 0$.

1: Set $k = 0$.
2: **while** $\|\theta^k - z^k\| \geq tol$ and $\|\rho(h^k - h^{k-1})\| \geq tol$ **do**
3: $\quad \theta^{k+1} = \left(2\mathbf{A}\mathbf{A}^t + \rho\mathbf{I}\right)^{-1}\left(2\mathbf{A}^t\mathbf{V} + \rho(z^k - h^k)\right)$
4: $\quad z^{k+1} = \mathrm{Prox}_{\frac{\lambda}{\rho}\|\cdot\|_{1,\mathbf{w}}}(\theta^{k+1} + h^k)$.
5: $\quad h^{k+1} = h^k + \theta^{k+1} - z^{k+1}$.
6: $\quad$ Set $k = k + 1$ and go to Step 2.

algorithm, $\mathbf{I}$ stands for the identity matrix and the proximal operator $\mathrm{Prox}_{\frac{\lambda}{\rho}\|\cdot\|_{1,\mathbf{w}}}$ is explicitly given by a soft-thresholding type operator [15, Chapter 6]

$$\mathrm{Prox}_{\frac{\lambda}{\rho}\|\cdot\|_{1,\mathbf{w}}}(\mathbf{x}) = \left([|x_i| - \frac{\lambda w_i}{\rho}]_+ \,\mathrm{sgn}(x_i)\right)_{i=1}^{q} \quad \text{for } \mathbf{x} = (x_1, \ldots, x_q)\,, \tag{50}$$

where $[\cdot]_+$ denotes the positive part. The application of Algorithm 2 for problem (AP_{ℓ_1}) is directly done by replacing $\mathbf{A}$ and $\mathbf{V}$ by $\bar{\mathbf{A}}$ and $\bar{\mathbf{V}}$, respectively.

3.4 Recovering time-dependent feedback laws.

The present computational framework can be extended to recover time-dependent value functions and feedback laws. While we argue that the primary object of study in deterministic optimal control of physical systems is the synthesis of static feedback laws, there exist applications in finance and operations research where the computation of time-dependent feedback controls is of great interest [41]. As discussed at the end of Section 2, the solution of the optimal control problem for a given initial condition $(t^j, \mathbf{x}^j)$ generates data for $V(t, \mathbf{y}^*(t))$ and $\nabla V(t, \mathbf{y}^*(t))$ with $t \in (t^j, T)$, along the optimal trajectory departing from $\mathbf{y}^*(t^j) = \mathbf{x}^j$. From an approximation viewpoint, this space-time data can be use to fit a model for $V(t, \mathbf{x})$. The simplest option is to approximate $V(t, \mathbf{x})$ along the space-time cylinder treating time in the same way as $\mathbf{x}$, that is

$$V(t, \mathbf{x}) \approx V_\theta(\tilde{\mathbf{x}}) = \sum_{\mathbf{i} \in \mathfrak{I}} \theta_{\mathbf{i}} \Phi_{\mathbf{i}}(\tilde{\mathbf{x}})\,, \qquad \tilde{\mathbf{x}} = (t, \mathbf{x}) \in \mathbb{R}^{n+1}\,. \tag{51}$$

The computational cost of this augmented representation is related to the new polynomial basis and the increase of $|\mathfrak{I}(s)|$ in $\mathbb{R}^n$ to $\mathbb{R}^{n+1}$. An alternative to this treatment is to establish a time-marching structure for t, and embed this time dependence in θ, similar to the method of lines for parabolic PDEs [70]. Formally, we write

$$V(t, \mathbf{x}) \approx V_\theta(t, \mathbf{x}) = \sum_{\mathbf{i} \in \mathfrak{I}} \theta_{\mathbf{i}}(t) \Phi_{\mathbf{i}}(\mathbf{x})\,, \tag{52}$$

where an additional approximation for θ is required. It is reasonable to assume that the artificial dataset from the numerical optimal control solutions will be provided as a time series with a uniform time discretization parameter τ, so we use a piecewise constant approximation for $\theta_{\mathbf{i}}(t)$

$$\theta_{\mathbf{i}}(t) \approx \theta_{k,\mathbf{i}}\,, \qquad \text{for } t \in [(k-1)\tau, k\tau)\,, \qquad k = 1, \ldots, N_T \tag{53}$$

where $N_T = T/\tau$, and the space-time approximation of $V(t, \mathbf{x})$ becomes

$$V_\theta(t, \mathbf{x}) = \sum_{k=1}^{N_T} \sum_{\mathbf{i} \in \mathfrak{I}} \theta_{k,\mathbf{i}} \Phi_{\mathbf{i}}(\mathbf{x}) \,, \tag{54}$$

which can be further simplified in the absence of terminal penalties in the cost functional since in this case $V(T, \mathbf{x}) = 0$. Thus, the computational increase is linear with respect to the cost associated to the static feedback law. A high-order discretization in time can be used to reduce the number of time nodes, however, it is necessary to always maintain a linear structure in θ.

4 Numerical Tests

In this section we assess the proposed methodology for recovering optimal feedback laws in three different tests. After presenting the practical aspects of our numerical implementation, we study the control of a nonlinear, low-dimensional oscillator. Then, we study large-dimensional dynamics arising in optimal control of nonlinear parabolic PDEs and non-local agent-based dynamics. In these tests, we focus on studying the effects of the sparse regression and the selection of weights in the ℓ_1 penalty, the gradient-augmented recovery, the selection of a suitable polynomial basis, and the effectiveness of the recovered control law. Both sampling and regression algorithms were implemented in MATLAB R2014b, and the numerical tests were run in a MacBook Pro with 2.9 GHz Dual-Core Intel Core i5 and memory 16 GB 1867 MHz DDR3.

4.1 Practical aspects

Generating the samples. For each test we fixed an $n - dimensional$ hyperrectangle as the domain for sampling initial condition vectors $\{\mathbf{x}^j\}_{j=1}^N \in \mathbb{R}^n$. These initial vectors were generated using Halton quasi-random sequences[1] in dimension n. Then for every $i \in \{1, \dots, N\}$, we compute the value function $V^j = V(0, \mathbf{x}^j)$ by solving the open-loop optimal control problem (8)-(9). Every optimal control problem was solved in the reduced form by using Algorithm 1 with $tol = 10^{-5}$ as discussed in Section 3.1. Note that the computational burden associated to solving an optimal control problem for each initial condition of the ensemble can be alleviated by directly parallelising this task. Further, for each control problem, the gradient of the value function was obtained by evaluating the solution $\mathbf{p}^*$ of the adjoint equation (10) at initial time $t_0 = 0$, so that $\nabla V_x^j(\mathbf{x}^j) = \nabla V(0, \mathbf{x}^j) = \mathbf{p}^*(0)$. This quantity is obtained as a by-product of solving the optimal control problem at not additional cost.

Training and validation. We split the sampling dataset $\{\mathbf{x}^j, V^j, V_x^j\}_{j=1}^N$ into two sets: a set of training indices $\mathcal{I}_{tr}$ which is used for regression, and a set of validation indices $\mathcal{I}_{val}$, with $\mathcal{I}_{val} \cup \mathcal{I}_{tr} \subset \{1, \dots, N\}$. Without loss of generality, we assume that $\mathcal{I}_{tr} = \{1, \dots, N_d\}$ and $\mathcal{I}_{val} = \{N_d + 1, \dots, N\}$ for $N \in \mathbb{N}$ with $N_d < N$. The linear least square problems (P_{ℓ_2}) and (AP_{ℓ_2}) were solved using a preconditioned conjugate gradient method, and the algorithm was terminated when the norm of residual was less than 10^{-8}. For the LASSO regressions (P_{ℓ_1}) and (AP_{ℓ_1}), we employed Algorithm 2 with $tol = 10^{-5}$. To analyse the generalization error of the

[1] `https://www.mathworks.com/help/stats/generating-quasi-random-numbers.html`

approximated value function $\bar{V}(\mathbf{x})$ with respect to the exact value $V(0,\mathbf{x})$, we use the following relative errors:

$$
\begin{aligned}
Err_{L^2}(\bar{V}) &= \left(\frac{\sum\limits_{j\in\mathcal{I}_{val}} |\bar{V}(\mathbf{x}^j) - V(0,\mathbf{x}^j)|^2}{\sum\limits_{j\in\mathcal{I}_{val}} |V(0,\mathbf{x}^j)|^2} \right)^{\frac{1}{2}}, \\
Err_{H^1}(\bar{V}) &= \left(\frac{\sum\limits_{j\in\mathcal{I}_{val}} \left(|\bar{V}(\mathbf{x}^j) - V(0,\mathbf{x}^j)|^2 + \sum_{i=1}^n \left| \frac{\partial \bar{V}(\mathbf{x}^j)}{\partial x_i} - \frac{\partial V(0,\mathbf{x}^j)}{\partial x_i} \right|^2 \right)}{\sum\limits_{j\in\mathcal{I}_{val}} \left(|V(0,\mathbf{x}^j)|^2 + \sum_{i=1}^n \left| \frac{\partial V(0,\mathbf{x}^j)}{\partial x_i} \right|^2 \right)} \right)^{\frac{1}{2}}.
\end{aligned}
\tag{55}
$$

Weights in the ℓ_1 norm. Regarding the selection of weights for the $\|\cdot\|_{1,\mathbf{w}}$ norm, we consider expressions of the form

$$
w_i = v_i^{\alpha} \quad \text{for } \alpha > 0, \tag{56}
$$

where the terms $v_{\mathbf{i}}$ depend on the polynomial basis chosen for regression. In the case of Legendre and Chebyshev polynomials, we proceed as in [68], which we summarize in the following. We consider tensorized Legendre polynomials on $\mathcal{D} = [-1,1]^n$ of the from

$$
\mathbf{L}_{\mathbf{i}}(\mathbf{x}) := \prod_{j=1}^{n} L_{i_j}(x_i) \text{ with } \quad \mathbf{i} = (i_1, i_2, \ldots, i_n) \in \mathbb{N}^n, \quad \mathbf{x} = (x_1, x_2, \ldots, x_n), \tag{57}
$$

with L_k defined as the univariate orthonormal Legendre polynomials of degree k. In this case, the Legendre polynomials form a basis for the real algebraic polynomials on $\mathcal{D}$ and are orthogonal with respect to the tensorized uniform measure on $\mathcal{D}$. Moreover, due to the fact that $\|L_k\|_{L^\infty(-1,1)} \leq \sqrt{k}$, we can write that

$$
\|\mathbf{L}_{\mathbf{i}}\|_{L^\infty(\mathcal{D})} \leq \prod_{j=1}^{n} (1+i_j)^{\frac{1}{2}}, \tag{58}
$$

from where we take the hyperbolic cross weights $v_{\mathbf{i}} = \prod_{j=1}^n (1+i_j)^{\frac{1}{2}}$. For tensorized Chebyshev polynomials on $\mathcal{D}$

$$
\mathbf{C}_{\mathbf{i}}(\mathbf{x}) := \prod_{j=1}^{n} C_{i_j}(x_i) \text{ with } \quad \mathbf{i} = (i_1, i_2, \ldots, i_n) \in \mathbb{N}^n, \quad \mathbf{x} = (x_1, x_2, \ldots, x_n), \tag{59}
$$

with $C_k(x) = \sqrt{2}\cos((k-1)\arccos(x))$, the uniform bound $\|C_k\|_{L^\infty(-1,1)} \leq \sqrt{2}$ holds, leading to $\|\mathbf{C}_{\mathbf{i}}\|_{L^\infty(\mathcal{D})} \leq 2^{\frac{\|\mathbf{i}\|_0}{2}}$. The latter is a valid alternative for setting $v_{\mathbf{i}}$, however we note that the bound (58) also holds in this case, so we choose $v_{\mathbf{i}} = \prod_{j=1}^n (1+i_j)^{\frac{1}{2}}$. To fit with these settings, in our numerical experiments we rescale the sampling set of initial conditions $\mathcal{D}$ to the unit hypercube $[-1,1]^n$.

4.2 Test 1: Van der Pol oscillator

We consider the optimal control of the Van der Pol oscillator expressed as

$$\min_{\mathbf{u}\in L^2(0,T;\mathbb{R})} \int_0^T y_1^2(t) + y_2^2(t) + \beta \mathbf{u}^2(t) dt \tag{60}$$

subject to

$$\begin{cases} \partial_t y_1 = y_2, \\ \partial_t y_2 = -y_1 + y_2(1 - y_1^2) + u, \\ (y_1(0), y_2(0)) = (x_1, x_2), \end{cases} \tag{61}$$

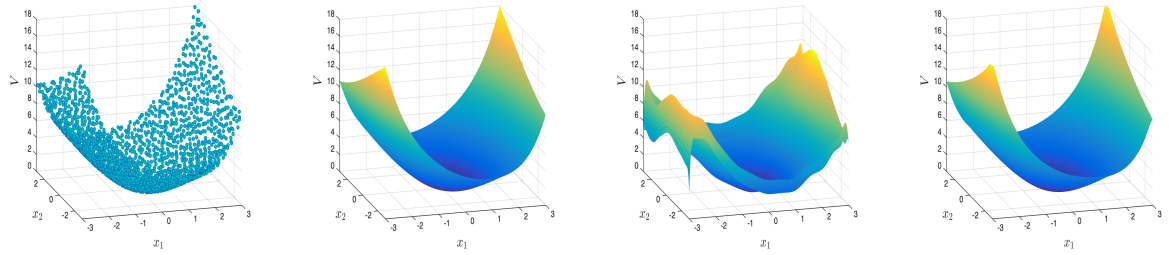
where we set $\mathbf{x} := (x_1, x_2)$, $\beta = 0.1$, and $T = 3$. A dataset $\{\mathbf{x}^j, V^j, V_x^j\}_{j=1}^N$ with $N = 2000$ is prepared by solving open-loop problems for different values of quasi-randomly chosen initial vectors from the domain $\mathcal{D} = [-3, 3]^2$. The temporal discretization is done by the Crank-Nicolson time stepping method with step-size $\Delta t = 10^{-4}$. Here we set $s = 16$ in the Hyperbolic cross index set $\mathfrak{I}(s)$ given in (35). In this case we have $q \equiv |\mathfrak{I}(s)| = 52$. That is, we use 52 polynomial basis functions to approximate the value function $V(\mathbf{x}) := V(0, \mathbf{x})$ corresponding to the optimal control problem (60)-(61).

We computed the solutions θ_{ℓ_2}, $\bar{\theta}_{\ell_2}$, θ_{ℓ_1} and $\bar{\theta}_{\ell_1}$ to the problems (P_{ℓ_2}) , (AP_{ℓ_2}), (P_{ℓ_1}), and (AP_{ℓ_1}), respectively, analysing different sizes for the training dataset N_d, the choice of ℓ_1 weights encoded through α in (56), and polynomial bases. A compact summary of these results is given in Table 1. The first column gives the errors of the value function and the nonzero components in its expansion without relying on gradient information and without sparsification. In the second column gradient information is added and the errors decrease for the same number of training samples ($N_d = 40$). For the third column the sparsity enhancing functional is added and approximately the same errors are obtained with significantly fewer nonzero components in the expansion.

	Err_{L^2}	Err_{H^1}	Nonzero components
V_{ℓ_2} for $N_d = 40$, $\lambda = 0.002$	1.46×10^{-1}	1.17	52/52
$\bar{V}_{\ell_2}$ for $N_d = 40$, $\lambda = 0.01$	9.38×10^{-3}	3.25×10^{-2}	52/52
$\bar{V}_{\ell_1}$ for $N_d = 40$, $\lambda = 0.01$	1.20×10^{-2}	2.05×10^{-2}	19/52

Table 1: Test 1. Numerical Results for Legendre polynomial basis and $\alpha = 1$ for $\bar{V}_{\ell_1}$. Including gradient information and sparsification leads to less error and fewer components in the expansion with a reduced number of training samples.

To illustrate the approximation of the value function $V(\mathbf{x})$ by the different regression formulations, we consider Figure 2(a). This figure displays the scatter plot associated to the training and validation data $\{\mathbf{x}^j, V^j\}_{j=1}^N$ with $N = 2000$. Figure 2(b) shows the approximation on the bases of (AP_{ℓ_1}) for the Legendre polynomial basis, with $\lambda = 0.01$, $\alpha = 1$, and $N_d = 40$. It clearly outperforms the approximation on the basis of (P_{ℓ_2}) given in Figure 2(c), again with $N_d = 40$. To achive a similar result without including gradient information would require to increase the size of the training set to $N_d = 120$, as shown in Figure 2(d). Sparse regression with gradient-augmented information provides an accurate reduced complexity approximation with fewer training samples.



(a) Scatter plot of sampling data (b) $\bar{V}_{\ell_1}$ for $N_d = 40$ (c) V_{ℓ_2} for $N_d = 40$ (d) V_{ℓ_2} for $N_d = 120$

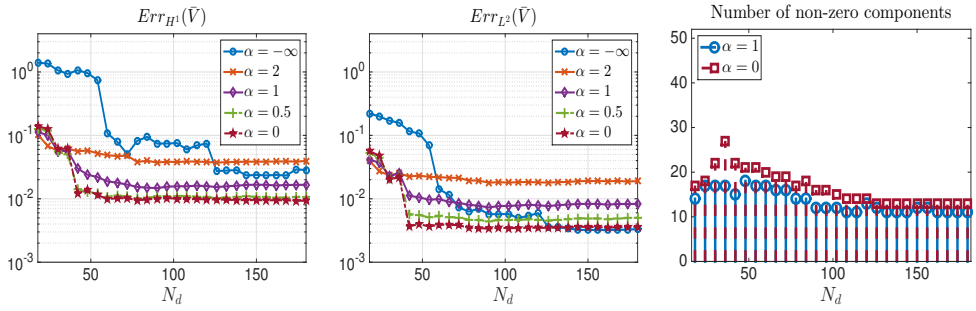
Figure 2: Test 1. (a) Training and validation dataset (b) Sparse regression with gradient-augmented information and $N_d = 40$ training points (c) Linear least squares without gradient-augmented information, $N_d = 40$ (d) Linear least squares without gradient information with $N_d = 120$. Sparse regression with gradient-augmented information provides an accurate reduced complexity approximation with fewer training samples.

We next turn to Figures 3–6 where the errors according to (55) are plotted on a logarithmic scale ($\log_{10}$) with respect to the number of samples N_d used for training. Here $|\mathcal{I}_{val}| = 1800$ validation samples were used. For problem (P_{ℓ_1}) and (AP_{ℓ_1}) we chose the sparse penalty parameter $\lambda = 0.002$ and $\lambda = \{0.01, 0.02\}$, respectively. Choosing λ larger for (AP_{ℓ_1}) than for (P_{ℓ_1}) allows to approximately balance the contributions for the data and the regularization terms in the cost functionals of these two problems. The cardinality of the non-zero coefficients of θ_{ℓ_1} and $\bar{\theta}_{\ell_1}$ is determined by defining components as nonzero if its absolute value is bigger than double machine extended precision 10^{-20}. Let us next make some observations on these results.

As expected, the error decreases with the training size N_d, up to a certain threshold. The best possible fit for the chosen order $s = 16$ ($q = 52$) of the polynomial approximation is reached at about $N_d = 50$ and $N_d = 75$ for Legendre and Chebyshev polynomials, respectively, without the use of gradient information, see Figures 3 and 4. These error levels are reached much earlier when we include gradient information, as shown in Figures 5 and 6. The influence of the ℓ_1 weights, expressed in terms of α, is not very pronounced. Note that $\alpha = -\infty$ corresponds to no regularisation, whereas $\alpha = 0$ corresponds to a constant weight. In the case that $N_d \lll 52$, the system is highly under-determined for $\alpha = -\infty$, which goes along with a large error. For small N_d, the choice $\alpha = 2$ can be favoured over the choice $\alpha = 0$, with the latter giving best results for N_d sufficiently large.

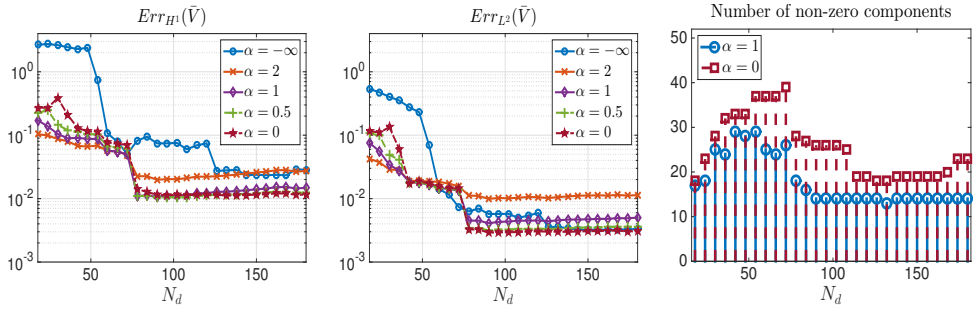
In the last column of these plots the cardinality of nonzero coefficients for $\bar{\theta}_{\ell_1}$ is depicted. It typically increases with N_d up to a certain threshold, and roughly stays constant thereafter, at less than 50 percent of the total number of free coefficients. Increasing λ promotes sparsity, as expected, see Figures 5 -6, (c) and (f).

Figures 3 and 5 correspond to the Legendre polynomial basis, while Figures 4 and 6 are obtained with Chebyshev polynomials. The results are quite similar in terms of the asymptotic (w.r.t. N_d) behavior of the errors. The number of non-zero components is higher for the Chebyshev than for the Legrendre polynomial expansion. By comparing Figures 5 (resp. Figure 6) with 3 (resp. Figure 4), we can see that for obtaining the same precision of approximation we need to consider more samples. As expected, for the case of gradient-augmented approximation we obtained better results for the H_1-errors for small N_d.



(a) $\lambda = 0.002$ (b) $\lambda = 0.002$ (c) $\lambda = 0.002$

Figure 3: Test 1. Numerical results for the polynomial approximation without gradient informations using the Legendre polynomial basis.



(a) $\lambda = 0.002$ (b) $\lambda = 0.002$ (c) $\lambda = 0.002$

Figure 4: Test 1. Numerical results for the polynomial approximation without gradient informations using the Chebyshev polynomial basis.

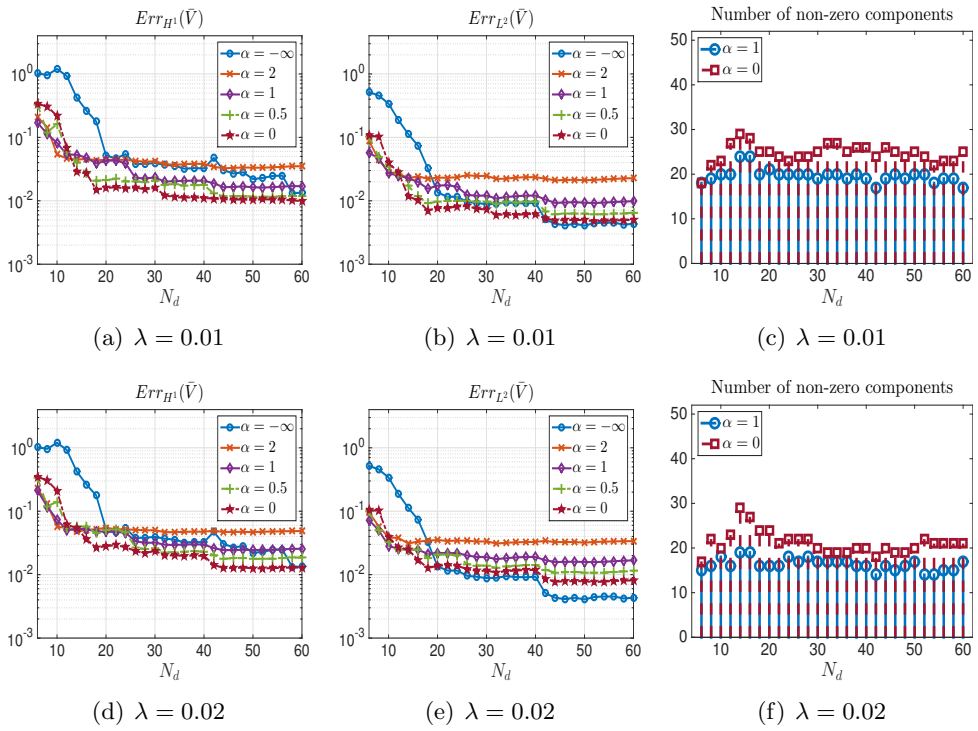
Moreover, we approximated the optimal control by using (48) and (49). To be more precise, we compute the following feedback law:

$$\mathbf{u}_\theta(\mathbf{x}) = -\frac{1}{2\beta}\sum_{\mathbf{i}\in\mathfrak{I}}\theta_{\mathbf{i}}\nabla_x\Phi_{\mathbf{i}}(\mathbf{x}). \tag{62}$$

where $\nabla_x\Phi_{\mathbf{i}}$ stands for the gradient of the polynomial basis. We applied this feedback law for the choices θ_{ℓ_2}, $\bar{\theta}_{\ell_2}$, and $\bar{\theta}_{\ell_1}$ for two initial vectors $(2,1)$ and $(2,-1)$. The evolution of the norm for the states controlled by these feedback laws, compared to the optimal state, and the uncontrolled state is illustrated in Figures 7(a) and 8(a). Figures 7(b) and 8(b) depict the evolution of the absolute value of the controls. Clearly, the controls $\mathbf{u}_{\bar{\theta}_{\ell_2}}$ and $\mathbf{u}_{\bar{\theta}_{\ell_1}}$ on the basis of $\bar{V}_{\ell_2}$ and $\bar{V}_{\ell_1}$ approximate well the challenging behaviour of the optimal control, and they outperform $\mathbf{u}_{\theta_{\ell_2}}$ obtained by V_{ℓ_2}.

Another advantage of using a sparse regression is the synthesis of a feedback law of reduced complexity. This is particularly relevant for the implementation of feedback laws in a real-time environment, where the number of calculations in the control loop needs to be minimized. In general, a feedback law expressed in the form

$$\mathbf{u}_\theta(\mathbf{x}) = -\frac{1}{2\beta}\mathbf{g}^t(\mathbf{x})\sum_{\mathbf{i}\in\mathfrak{I}}\theta_{\mathbf{i}}\nabla_x\Phi_{\mathbf{i}}(\mathbf{x})\,, \qquad \text{with } \; g(\mathbf{x})\in\mathbb{R}^{n\times m}\,, \nabla\Phi_i(\mathbf{x})\in\mathbb{R}^n\,, \tag{63}$$



(a) $\lambda = 0.01$ (b) $\lambda = 0.01$ (c) $\lambda = 0.01$

(d) $\lambda = 0.02$ (e) $\lambda = 0.02$ (f) $\lambda = 0.02$

Figure 5: Test 1. Numerical results for the gradient-augmented polynomial approximation using the Legendre polynomial basis.

requires $\mathcal{O}((mn^2+n)q)$ floating-point operations, where q is the number of non-zero components in the expansion. Thus, the operation count decreases linearly with the level of sparsity. Going back to Table 1, this implies a reduction of 63% in the number of operations with respect to an ℓ_2-based controller.

4.3 Test 2: Controlled Allen-Cahn equation

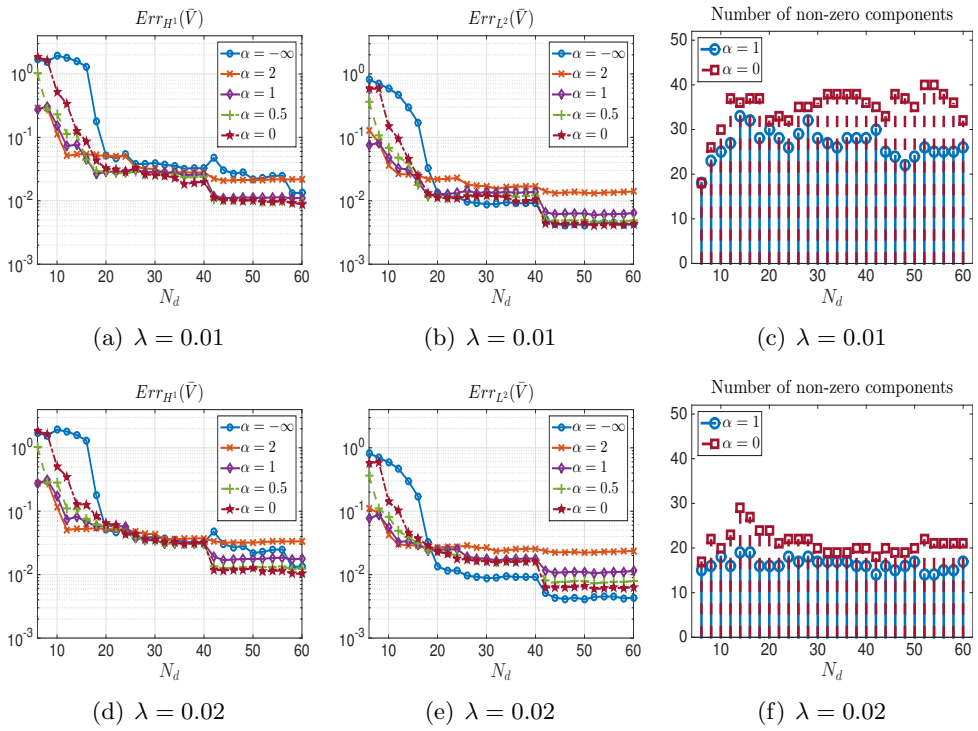
In the following test, we consider the PDE-constrained optimal control problem

$$\min_{\mathbf{u}\in L^2(0,T;\mathbb{R}^2)} \int_0^T (\|y(t)\|^2_{L^2(0,1)} + \beta|\mathbf{u}(t)|^2_{\ell^2})dt \tag{64}$$

subject to

$$\begin{cases} \partial_t y - \nu\partial_x^2 y - y(1-y^2) = \sum_{i=1}^3 u_i(t)\mathbf{1}_{\omega_i} & \text{in } (0,T)\times\Omega, \\ \partial_x y(t,1) = \partial_x y(t,0) = 0 & \text{in } (0,T), \\ y(0,y) = y_0 & \text{in } \Omega, \end{cases} \tag{65}$$

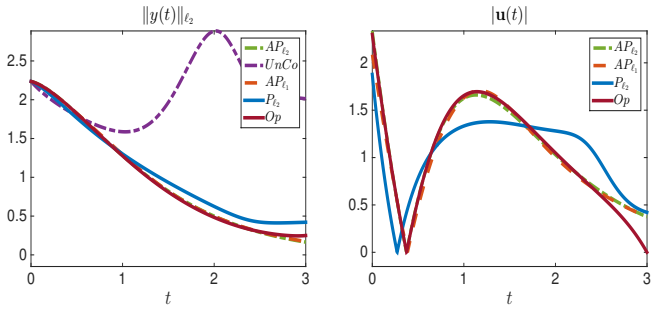
with the 3-d control vector $\mathbf{u}(t) := [u_1(t), u_2(t), u_3(t)] \in L^2(0,4;\mathbb{R}^3)$, $\Omega = (-1,1)$, $\nu = 0.1$, $\beta = 0.01$ and $T = 4$. The control signals act through $\mathbf{1}_{\omega_i} = \mathbf{1}_{\omega_i}(x)$, which denote the indicator functions with supports $\omega_1 = (-0.7,-0.4)$, $\omega_2 = (-0.2,0.2)$, and $\omega_3 = (0.4,0.7)$. Due to the infinite-dimensional nature of the state equation this problem does not fall directly into the optimal control setting considered in this paper. We first perform an approximation of (65) in



(a) $\lambda = 0.01$ (b) $\lambda = 0.01$ (c) $\lambda = 0.01$

(d) $\lambda = 0.02$ (e) $\lambda = 0.02$ (f) $\lambda = 0.02$

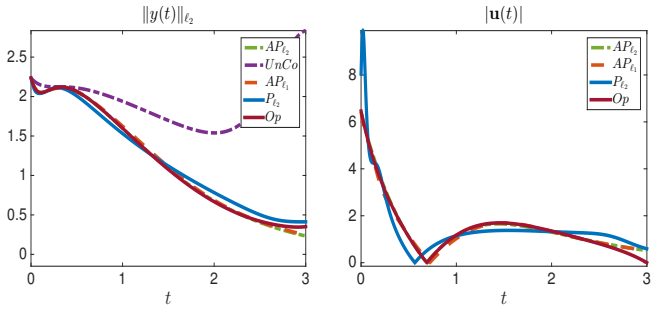
Figure 6: Test 1. Numerical results for the gradient-augmented polynomial approximation using the Chebyshev polynomial basis.



(a) State (b) Control

Figure 7: Test 1. Evolution of $\|y(t)\|_{\ell_2}$ and $|\mathbf{u}(t)|$ for $\mathbf{x} = (2, -1)$. Here *UnCo* stands for the uncontrolled trajectory, and *Op* refers to the exact optimal trajectory.

space by doing a pseudospectral collocation using Chebyshev spectral elements with 18 degrees of freedom as in [54]. This approximates the PDE control dynamics as an 18-dimensional nonlinear dynamical system. The resulting ODE system was treated numerically by the Crank-Nicolson time stepping method with step-size $\Delta t = 0.005$. Subsequently, a dataset $\{\mathbf{x}^j, V^j, V_x^j\}_{j=1}^N$ with $N = 9000$ (including samples for training and validation) was generated by solving open-loop problem (64)-(65) for different values of quasi-randomly chosen initial vectors from the hypercube $[-10, 10]^{18}$.



(a) State (b) Control

Figure 8: Test 1. Evolution of $\|y(t)\|_{\ell_2}$ and $|\mathbf{u}(t)|$ for $\mathbf{x} = (2, 1)$. Here *UnCo* stands for the uncontrolled trajectory, and *Op* refers to the exact optimal trajectory.

For this example we used the two different values $s = 4$ and $s = 8$ in the hyperbolic cross index set $\mathfrak{I}(s)$. For these choices we have $|\mathfrak{I}(4)| = 226$ and $|\mathfrak{I}(8)| = 1879$, resulting in 226 (resp. 1897) polynomial basis functions to approximate the value functions $V := V(0, \cdot)$. We computed the solutions θ_{ℓ_2}, $\bar{\theta}_{\ell_2}$, θ_{ℓ_1} and $\bar{\theta}_{\ell_1}$ to problems (P_{ℓ_2}) , (AP_{ℓ_2}), (P_{ℓ_1}), and (AP_{ℓ_1}) for the different choices of N_d, α, and polynomial basis. For problems (P_{ℓ_1}) and (AP_{ℓ_1}) we show results with $\lambda = 0.01$ and $\lambda = 0.008, 0.04$, respectively, using Chebyshev polynomials. Similarly as in Test 1, we report the level of non-sparsity of θ_{ℓ_1}, $\bar{\theta}_{\ell_1}$. The errors (55) are shown for a validation set with $|\mathcal{I}_{val}| = 5000$ samples. These results are depicted in Figures 9 and 10.

Overall, these results allow to draw the same conclusions as for the previous test. In particular, as N_d increases, the validation errors are getting smaller. Again, comparing Figures 9 and 10 the gradient-augmented results reach the lowest errors with significantly smaller datasets N_d than the gradient-free ones. Figures 9-10 also confirm that the errors for $s = 8$ are smaller compared to $s = 4$. Naturally, for the set of polynomials basis associated to $s = 8$, we need to increase the training data in comparison to $s = 4$. Comparing rows 1 and 2 in Figure 9, we observe that decreasing λ results in an increase on the number of non-zero components and in a decrease of the errors.

We approximated the optimal control using (49) and (48), resulting in the feedback law

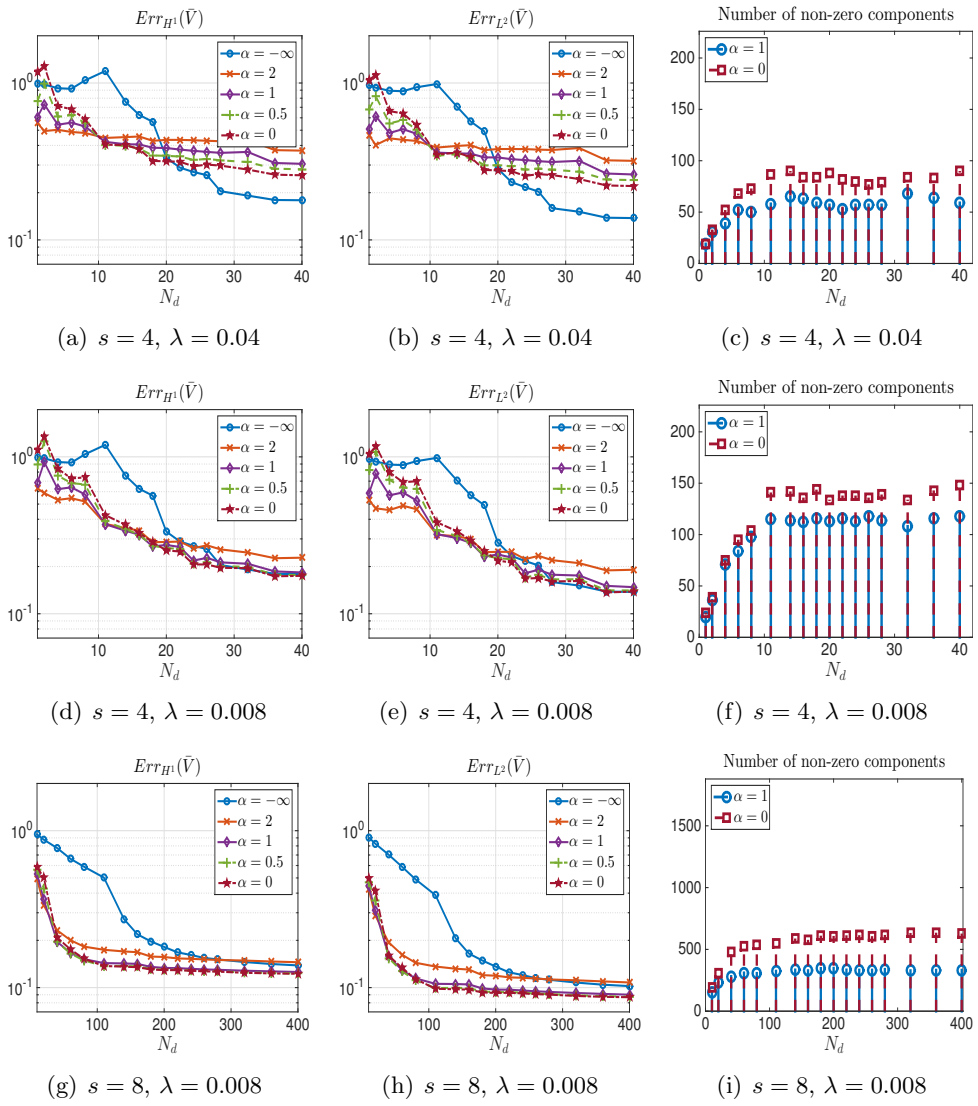
$$\mathbf{u}_\theta(\mathbf{x}) = -\frac{1}{2\beta}\mathbf{g}^t \sum_{\mathbf{i}\in\mathfrak{I}} \theta_{\mathbf{i}} \nabla_x \Phi_{\mathbf{i}}(\mathbf{x}) \,, \tag{66}$$

where $\mathbf{g} := [\mathbf{1}_{\omega_1}|\mathbf{1}_{\omega_2}|\mathbf{1}_{\omega_3}] \in \mathbb{R}^{n\times 3}$. We applied this feedback law for the choices θ_{ℓ_2}, $\bar{\theta}_{\ell_2}$, and $\bar{\theta}_{\ell_1}$ on the initial condition

$$y_0(x) = (x - 1)(x + 1) + 5. \tag{67}$$

We report the results for the Chebyshev polynomial basis with $s = 4$ and thus $q = 226$. For the case $\bar{\theta}_{\ell_1}$ we set $\lambda = 0.008$ and $\alpha = 1$. The evolution of the norm of the resulting controlled, optimal, and uncontrolled states are depicted in Figures 11(a), and the associated controls in 11(b).

Figure 12(a) depicts the uncontrolled state. It converges to the stable equilibrium given by the constant function with value 1. The state controlled by $\mathbf{u}_{\bar{\theta}_{\ell 1}}$ is illustrated in Figure 12(b). Here the state tends to 0 as expected.



(a) $s = 4$, $\lambda = 0.04$ (b) $s = 4$, $\lambda = 0.04$ (c) $s = 4$, $\lambda = 0.04$

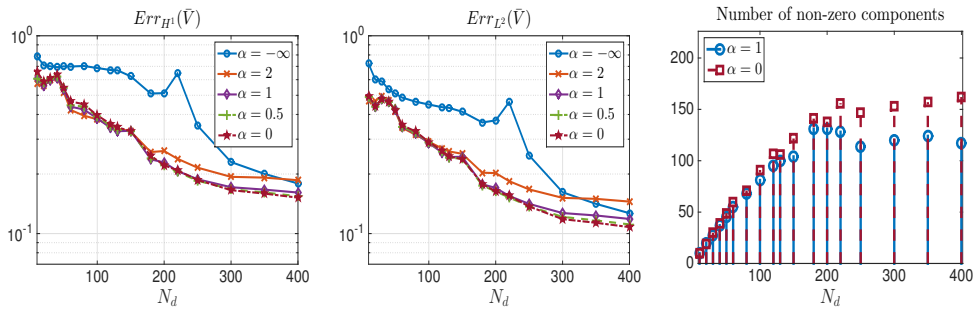
(d) $s = 4$, $\lambda = 0.008$ (e) $s = 4$, $\lambda = 0.008$ (f) $s = 4$, $\lambda = 0.008$

(g) $s = 8$, $\lambda = 0.008$ (h) $s = 8$, $\lambda = 0.008$ (i) $s = 8$, $\lambda = 0.008$

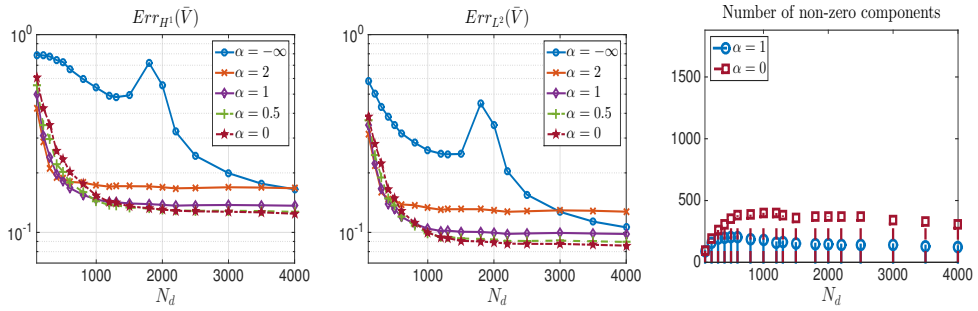
Figure 9: Test 2. Numerical results for the gradient-augmented polynomial approximation using the Chebyshev polynomial basis

4.4 Test 3: Optimal consensus control in the Cucker-Smale model

We conclude with a thorough discussion of a high-dimensional, non-linear, non-local optimal control problem related to consensus control of agent-based dynamics [11, 21, 25]. We study the Cucker-Smale model [35] for consensus control with N_a agents with states $(y_i, v_i) \in \mathbb{R}^d \times \mathbb{R}^d$ for $i = 1, \ldots, N_a$, where y_i and v_i stand for the position and velocity of the i-th agent, respectively,



(a) $s = 4$, $\lambda = 0.01$ (b) $s = 4$, $\lambda = 0.01$ (c) $s = 4$, $\lambda = 0.01$



(d) $s = 8$, $\lambda = 0.01$ (e) $s = 8$, $\lambda = 0.01$ (f) $s = 8$, $\lambda = 0.01$

Figure 10: Test 2. Numerical results for the polynomial approximation without gradient informations using the Chebyshev polynomial basis.

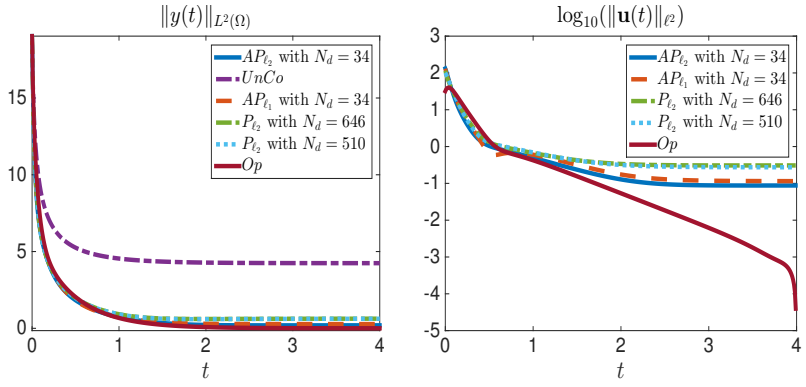
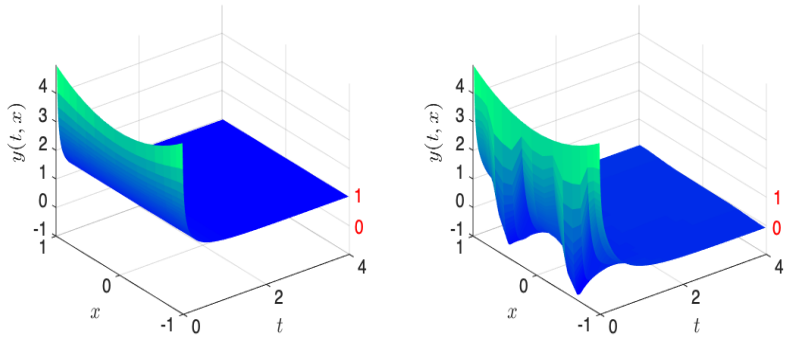


Figure 11: Test 2. Evolution of $\|y(t)\|_{L^2(\Omega)}$ and $\log_{10}(\|\mathbf{u}(t)\|)$ for the choices $\lambda = 0.008$, $\alpha = 1$, and Chebyshev polynomial basis.

and $d \in \mathbb{N}$ is the dimension of the physical space. Then dynamics of the agents are governed by

$$\begin{aligned} \frac{dy_i}{dt} &= w_i\,, \\ \frac{dv_i}{dt} &= \frac{1}{N_a}\sum_{j=1}^{N_a} \frac{v_j - v_i}{1 + \|y_i - y_j\|^2} + u_i\,, \qquad i = 1, \dots, N_a\,, \\ y_i(0) &= x_i\,, \quad v_i(0) = w_i\,. \end{aligned} \tag{68}$$



(a) Uncontrolled (b) Controlled

Figure 12: Test 2. The uncontrolled state and the controlled state by $\mathbf{u}_{\bar{\theta}_{\ell 1}}$ for $N_d = 34$, $\lambda = 0.008$, $\alpha = 1$, and $\alpha = 1$.

The consensus control problem consists of finding a control $\mathbf{u}(t) := (u_1(t), \dots, u_{N_a}(t)) \in \mathbb{R}^{d \times N_a}$ which steers the system towards the consensus manifold

$$v_i = \bar{v} = \frac{1}{N_a} \sum_{j=1}^{N_a} v_j \,, \quad \forall i = 1, \dots, N_a \,. \tag{69}$$

Asymptotic consensus emergence is conditional to the cohesiveness of the initial state $\mathbf{x}_0 = (x_1, \dots, x_{N_a})$ and $\mathbf{w}_0 = (w_1, \dots, w_{N_a})$ [35]. To remove this dependence on the initial state, we cast this problem as an optimal control problem by defining the following cost functional

$$J(\mathbf{u}; \mathbf{x}_0, \mathbf{w}_0) := \int_0^T \sum_{i=1}^{N_a} \frac{1}{N_a} \|v_i(t) - \bar{v}\|^2 + \beta \|u_i(t)\|^2 \, dt, \tag{70}$$

and formulating the optimal control problem

$$\min_{\mathbf{u} \in L^2(0,T;\mathbb{R}^{d \times N_a})} \{J(\mathbf{u}; \mathbf{x}_0, \mathbf{v}_0) \text{ subject to } (68)\}. \tag{$OC(\hat{\mathbf{x}}_0)$}$$

For the sake of completeness, in this case the adjoint system is given by [11]

$$-\frac{dp_{y_i}}{dt} = \frac{1}{N_a} \sum_{j \neq i} \frac{-2(p_{v_j} - p_{v_i})}{(1 + \|y_j - y_i\|^2)^2} [(y_i - y_j) \otimes (v_i - v_j)] \,, \tag{71}$$

$$-\frac{dp_{v_i}}{dt} = p_{y_i} + \frac{1}{N_a} \sum_{j \neq i} \frac{p_{v_j} - p_{v_i}}{1 + \|y_i - y_j\|^2} + \frac{2}{N} (v_i - \bar{v})^t \qquad i = 1, \dots, N_a \,, \tag{72}$$

$$p_{y_i}(T) = 0, \quad p_{v_i}(T) = 0 \,, \tag{73}$$

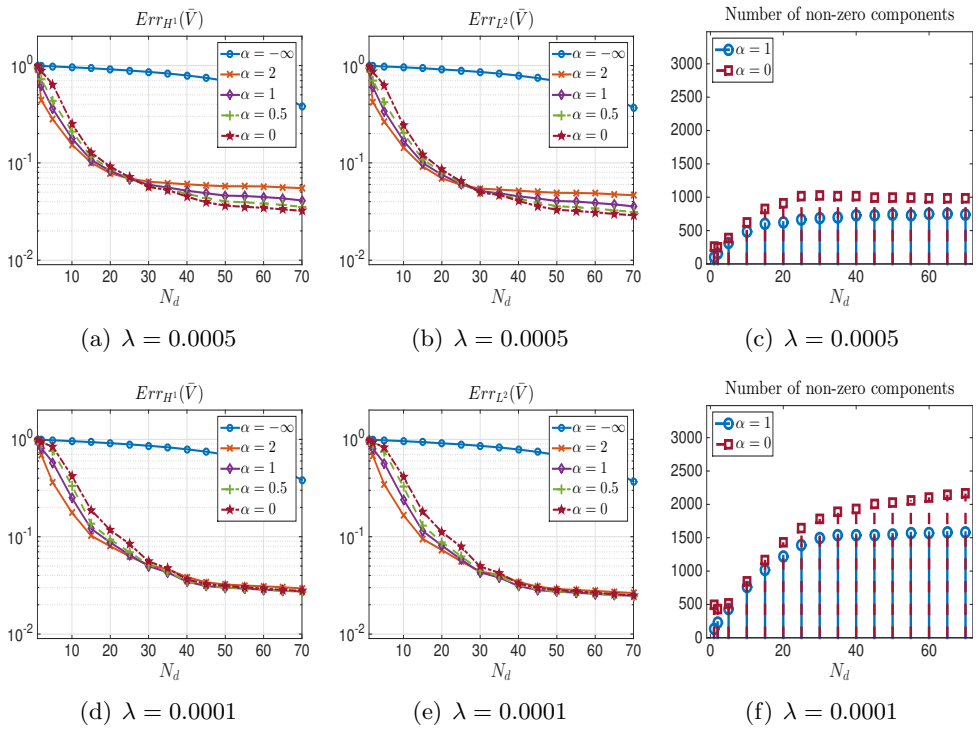
and the optimality condition reads

$$p_{v_i}(t) + 2\beta u_i^*(t) = 0 \qquad \forall \, t \in (0, T) \,, \, i = 1, \dots, N \,. \tag{74}$$

We denote the augmented initial state $\hat{\mathbf{x}}_0 = (\mathbf{x}_0, \mathbf{w}_0)$, and we approximate the value function $V(\hat{\mathbf{x}}) = V(0, \hat{\mathbf{x}})$. We set $N_a = 20$, $d = 2$, $T = 10$, and $\beta = 0.01$, and we compute a

dataset $\{\hat{\mathbf{x}}^j, V^j, V^j_{\hat{x}}\}_{j=1}^N$ with $N = 10^4$. For every j, the initial vectors $\hat{\mathbf{x}}^j \in \mathbb{R}^{80}$ were chosen quasi-randomly from the hypercube $[-3, 3]^{80}$. The dataset was computed by solving open-loop problems with a time discretization using the fourth order Runge-Kutta method with step-size $\Delta t = 0.01$.

The choice $s = 4$ in the hyperbolic cross index set $\mathfrak{I}(s)$, results in $|\mathfrak{I}(4)| = 3481$ polynomial basis functions for V. As in the previous examples, we computed the solutions θ_{ℓ_2}, $\bar{\theta}_{\ell_2}$, θ_{ℓ_1} and $\bar{\theta}_{\ell_1}$ for different values of N_d and λ. We report results for the errors and levels of non-sparsity for the case of Legendre polynomials as basis functions, $\lambda = 5 \times 10^{-4}, 10^{-4}$ and $|\mathcal{I}_{val}| = 5000$, in Figures 13 and 14, with and without gradient information, respectively. For this 80-dimensional problem, the observations from the previous examples are confirmed, and the gradient-augmented sparse regression requires two orders of magnitude less of training samples to achieve the same error levels of the gradient-free counterpart.



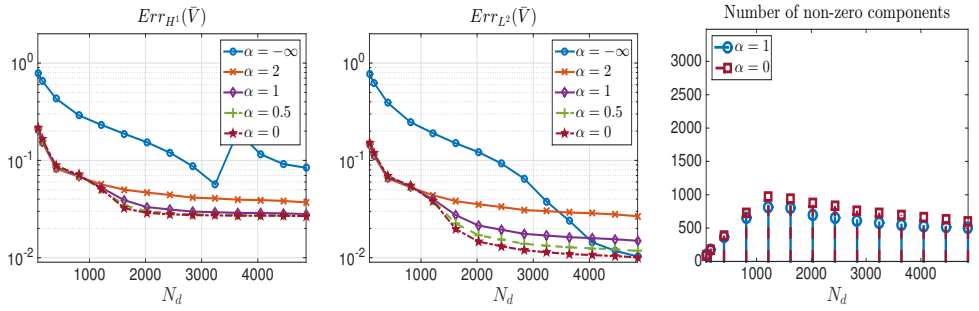
(a) $\lambda = 0.0005$ (b) $\lambda = 0.0005$ (c) $\lambda = 0.0005$

(d) $\lambda = 0.0001$ (e) $\lambda = 0.0001$ (f) $\lambda = 0.0001$

Figure 13: Test 3. Numerical results for the gradient-augmented polynomial approximation using the Legendre polynomial basis.

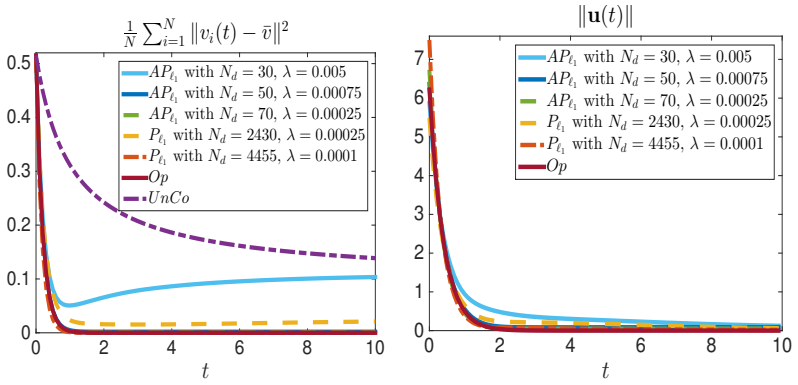
Furthermore, we computed the approximation of the optimal control according to (66) with $\mathbf{g} = [\mathbf{0}; I]^t$ for different setting of $\theta = \theta_{\ell_1}$ and $\theta = \bar{\theta}_{\ell_1}$, and present stabilization results in Figure 15.

Figure 16(a) shows the uncontrolled dynamics of the agents for a specific initial state $\hat{\mathbf{x}}_0$. The dynamics of the optimal state and approximations, corresponding to $OC(\hat{\mathbf{x}}_0)$, are plotted in Figures 16(b), 16(c), and 16(d). Colored trajectories are uncontrolled, and red trajectories represent the controlled evolution. By comparing these Figures, it can be seen that the dynamics of the optimal state and its approximation obtained by $\mathbf{u}_{\bar{\theta}_{\ell_1}}$ for $N_d = 70$, $\lambda = 2.5 \times 10^{-4}$ are almost identical. Note that the controlled trajectories in these two subplots 16(b) and 16(c)



(a) $\lambda = 0.0005$ (b) $\lambda = 0.0005$ (c) $\lambda = 0.0005$

Figure 14: Test 3. Numerical results of the polynomial approximation using the Legendre basis without gradient information.



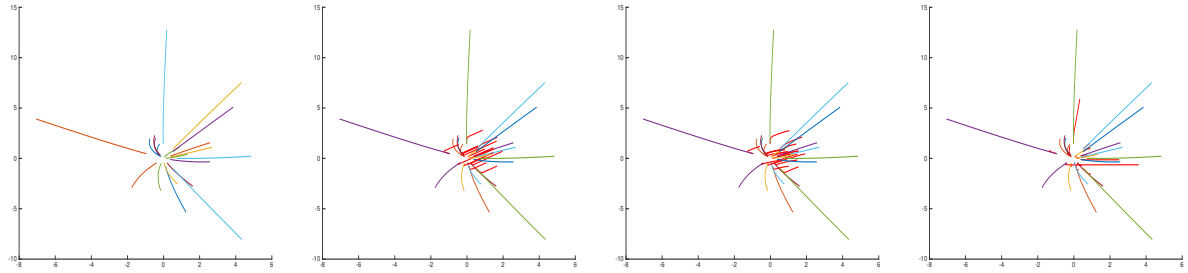
(a) State (b) Control

Figure 15: Test 3. (a) Evolution of the consensus variance $\frac{1}{N_a}\sum_{i=1}^{N_a} \|v_i(t) - \bar{v}\|^2$ for different control laws and (b) the norm $\|\mathbf{u}(t)\|$ of the associated control signals. Here *Op* stands for the exact optimal control, and *UnCo* for the uncontrolled solution.

achieve consensus, unlike 16(d), where the feedback obtained with a sparse regression without gradient information does not stabilize the dynamics despite the large training dataset. This observation is also supported by Table 2 and Figure (15). Table 2 shows that the smallest validation error is achieved for the gradient-augmented sparse regression of the value function with only 70 training samples, with approximately 20% of nonzero components. Figure (15) depicts the evolution of the tracking term $\frac{1}{N_a}\sum_{i=1}^{N_a} \|v_i(t) - \bar{v}\|^2$ and the norm of the control of every feedback law. From this figure, we can see that the control $\mathbf{u}_{\bar{\theta}_{\ell_1}}$ associated to $N_d = 70$, $\lambda = 2.5 \times 10^{-4}$ delivers the best approximation for the optimal control of $OC(\hat{\mathbf{x}}_0)$ among the different control laws.

Concluding remarks

We have presented a sparse polynomial regression framework for the approximation of feedback laws arising in nonlinear optimal control. The main ingredients of our approach include: the generation of a gradient-augmented dataset for the value function associated to the control



(a) Uncontrolled State (b) Optimal state (c) AP_{ℓ_1} with $N_d = 70$, $\lambda = 2.5 \times 10^{-4}$ (d) P_{ℓ_1} with $N_d = 2430$, $\lambda = 2.5 \times 10^{-4}$

Figure 16: Test 3. Controlled trajectories (in red) generated by different control laws. In this high-dimensional problem ($n = 80$), the sparse, gradient-augmented regression for V (c) yields an feedback law which approaches the optimal trajectory (b) with few training samples $N_d = 70$.

	Err_{L^2}	Err_{H^1}	Nonzero components
$\bar{V}_{\ell_1}$ for $N_d = 50$, $\lambda = 7.5 \times 10^{-4}$	5.40×10^{-2}	6.37×10^{-2}	393/3481
$\bar{V}_{\ell_1}$ for $N_d = 70$, $\lambda = 2.5 \times 10^{-4}$	3.56×10^{-2}	4.11×10^{-2}	738/3481
V_{ℓ_1} for $N_d = 2430$, $\lambda = 2.5 \times 10^{-4}$	7.46×10^{-2}	9.38×10^{-2}	656/3481

Table 2: Test 3. Validation errors for different regressions.

problem by means of PMP solves, a hyperbolic cross polynomial ansatz for recovering the value function and its feedback law, and a sparse optimization method to fit the model. Through a series of numerical tests, we have shown that the proposed approach can approximate high-dimensional control problems at moderate computational cost. The gradient-augmented dataset reduces the number of open-loop solves required to recover the optimal control, and the sparse regression provides a feedback law of reduced complexity, which is an appealing feature for real-time implementations. The effectiveness of the proposed methodology suggests different research directions. First, the deep neural network ansatz proposed in [63] can be combined with a sparsity-promoting loss function along the lines of our work. In the light of recent results discussed in [3], it is a pertinent question to find whether deep neural networks or polynomial approximants are more effective ansatz for the value function. As we have previously mentioned, there are different control-theoretical arguments which support the case for having a polynomial approximation of the value function. A second direction of research is related to the extension of the presented results in the context to time-dependent, and second-order stochastic control problems where the representation formula given by the PMP is replaced by a backward stochastic differential equation [43]. Finally, the ideas proposed in this work regarding sparse polynomial regression can be implemented in the context of approximate dynamic programming in reinforcement learning [19].

Acknowledgements. Dante Kalise was supported by a public grant as part of the Investissement d'avenir project, reference ANR-11-LABX-0056-LMH, LabEx LMH. Karl Kunisch was supported in part by the ERC advanced grant 668998 (OCLOC) under the EU's H2020 research program.

References

[1] B. Adcock, A. Bao, and S. Brugiapaglia. Correcting for unknown errors in sparse high-dimensional function approximation. *Numer. Math.*, 142(3):667–711, 2019.

[2] B. Adcock, S. Brugiapaglia, and C. G. Webster. Compressed sensing approaches for polynomial approximation of high-dimensional functions. In *Compressed sensing and its applications*, Appl. Numer. Harmon. Anal., pages 93–124. Birkhäuser/Springer, Cham, 2017.

[3] B. Adcock and N. Dexter. The gap between theory and practice in function approximation with deep neural networks, 2020. arXiv preprint: 2001.07523.

[4] B. Adcock and Y. Sui. Compressive Hermite interpolation: sparse, high-dimensional approximation from gradient-augmented measurements. *Constr. Approx.*, 50(1):167–207, 2019.

[5] M. Akian, S. Gaubert, and A. Lakhoua. The max-plus finite element method for solving deterministic optimal control problems: basic properties and convergence analysis. *SIAM J. Control Optim.*, 47(2):817–848, 2008.

[6] G. Albi, Y.-P. Choi, M. Fornasier, and D. Kalise. Mean field control hierarchy. *Appl. Math. Optim.*, 76(1):93–135, 2017.

[7] E. Al'brekht. On the optimal stabilization of nonlinear systems. *Journal of Applied Mathematics and Mechanics*, 25(5):1254, 1961.

[8] A. Alla, M. Falcone, and L. Saluzzi. An efficient DP algorithm on a tree-structure for finite horizon optimal control problems. *SIAM J. Sci. Comput.*, 41(4):A2384–A2406, 2019.

[9] B. Azmi and K. Kunisch. On the stabilizability of the Burgers equation by receding horizon control. *SIAM J. Control Optim.*, 54(3):1378–1405, 2016.

[10] B. Azmi and K. Kunisch. Analysis of the Barzilai-Borwein step-sizes for problems in Hilbert spaces. *J. Optim Theory Appl.*, 185:819–844, 2020.

[11] R. Bailo, M. Bongini, J. A. Carrillo, and D. Kalise. Optimal consensus control of the cucker-smale model. *IFAC-PapersOnLine*, 51(13):1 – 6, 2018.

[12] M. Bardi and I. Capuzzo-Dolcetta. *Optimal control and viscosity solutions of Hamilton-Jacobi-Bellman equations*. Systems & Control: Foundations & Applications. Birkhäuser Boston, Inc., Boston, MA, 1997. With appendices by Maurizio Falcone and Pierpaolo Soravia.

[13] E. N. Barron and R. Jensen. The Pontryagin maximum principle from dynamic programming and viscosity solutions to first-order partial differential equations. *Trans. Amer. Math. Soc.*, 298(2):635–641, 1986.

[14] J. Barzilai and J. M. Borwein. Two-point step size gradient methods. *IMA J. Numer. Anal.*, 8(1):141–148, 1988.

[15] A. Beck. *First-order methods in optimization*, volume 25 of *MOS-SIAM Series on Optimization*. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA; Mathematical Optimization Society, Philadelphia, PA, 2017.

[16] J. Beck, R. Tempone, F. Nobile, and L. Tamellini. On the optimal polynomial approximation of stochastic PDEs by Galerkin and collocation methods. *Math. Models Methods Appl. Sci.*, 22(9):1250023, 33, 2012.

[17] S. C. Beeler, H. T. Tran, and H. T. Banks. Feedback control methodologies for nonlinear systems. *J. Optim. Theory Appl.*, 107(1):1–33, 2000.

[18] R. Bellman. *Adaptive control processes: A guided tour*. Princeton University Press, Princeton, N.J., 1961.

[19] D. P. Bertsekas. *Reinforcement Learning and Optimal Control*. Athena Scientific, Belmont, MA, 2019.

[20] O. Bokanowski, J. Garcke, M. Griebel, and I. Klompmaker. An adaptive sparse grid semi-Lagrangian scheme for first order Hamilton-Jacobi Bellman equations. *J. Sci. Comput.*, 55(3):575–605, 2013.

[21] M. Bongini, M. Fornasier, and D. Kalise. (Un)conditional consensus emergence under perturbed and decentralized feedback controls. *Discrete Contin. Dyn. Syst.*, 35(9):4071–4094, 2015.

[22] S. Boyd, N. Parikh, E. Chu, B. Peleato, J. Eckstein, et al. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine learning*, 3(1):1–122, 2011.

[23] T. Breiten, K. Kunisch, and L. Pfeiffer. Taylor expansions of the value function associated with a bilinear optimal control problem. *Annales de l'Institut Henri Poincaré C, Analyse Non Linéaire*, 2019. Accepted for publication.

[24] P. Cannarsa and H. Frankowska. Some characterizations of optimal trajectories in control theory. *SIAM J. Control Optim.*, 29(6):1322–1347, 1991.

[25] M. Caponigro, M. Fornasier, B. Piccoli, and E. Trélat. Sparse stabilization and control of alignment models. *Math. Models Methods Appl. Sci.*, 25(3):521–564, 2015.

[26] A. Chkifa, A. Cohen, G. Migliorati, F. Nobile, and R. Tempone. Discrete least squares polynomial approximation with random evaluations - application to parametric and stochastic elliptic pdes. *ESAIM: M2AN*, 49(3):815–837, 2015.

[27] A. Chkifa, A. Cohen, and C. Schwab. Breaking the curse of dimensionality in sparse polynomial approximation of parametric PDEs. *J. Math. Pures Appl. (9)*, 103(2):400–428, 2015.

[28] Y.-P. Choi, D. Kalise, J. Peszek, and A. A. Peters. A collisionless singular Cucker-Smale model with decentralized formation control. *SIAM J. Appl. Dyn. Syst.*, 18(4):1954–1981, 2019.

[29] Y. T. Chow, J. Darbon, S. Osher, and W. Yin. Algorithm for overcoming the curse of dimensionality for time-dependent non-convex Hamilton-Jacobi equations arising from optimal control and differential games problems. *J. Sci. Comput.*, 73(2-3):617–643, 2017.

[30] Y. T. Chow, J. Darbon, S. Osher, and W. Yin. Algorithm for overcoming the curse of dimensionality for certain non-convex Hamilton-Jacobi equations, projections and differential games. *Ann. Math. Sci. Appl.*, 3(2):369–403, 2018.

[31] Y. T. Chow, J. Darbon, S. Osher, and W. Yin. Algorithm for overcoming the curse of dimensionality for state-dependent Hamilton-Jacobi equations. *J. Comput. Phys.*, 387:376–409, 2019.

[32] F. H. Clarke and R. B. Vinter. The relationship between the maximum principle and dynamic programming. *SIAM J. Control Optim.*, 25(5):1291–1311, 1987.

[33] A. Cohen, R. Devore, and C. Schwab. Analytic regularity and polynomial approximation of parametric and stochastic elliptic PDE's. *Anal. Appl. (Singap.)*, 9(1):11–47, 2011.

[34] E. Cristiani and P. Martinon. Initialization of the shooting method via the hamilton-jacobi-bellman approach. *Journal of Optimization Theory and Applications*, 146(2):321–346, Aug 2010.

[35] F. Cucker and S. Smale. Emergent behavior in flocks. *IEEE Trans. Automat. Control*, 52(5):852–862, 2007.

[36] Y.-H. Dai and H. Zhang. Adaptive two-point stepsize gradient algorithm. *Numer. Algorithms*, 27(4):377–385, 2001.

[37] J. Darbon, G. P. Langlois, and T. Meng. Overcoming the curse of dimensionality for some hamilton–jacobi partial differential equations via neural network architectures, 2019. arXiv preprint: 1910.09045.

[38] J. Darbon and S. Osher. Algorithms for overcoming the curse of dimensionality for certain Hamilton-Jacobi equations arising in control theory and elsewhere. *Res. Math. Sci.*, 3:Paper No. 19, 26, 2016.

[39] S. Dolgov, D. Kalise, and K. Kunisch. Tensor Decompositions for High-dimensional Hamilton-Jacobi-Bellman Equations, 2019. arXiv preprint: 1908.01533.

[40] M. Fornasier and F. Solombrino. Mean-field optimal control. *ESAIM: COCV*, 20(4):1123–1152, 2014.

[41] P. A. Forsyth. A Hamilton-Jacobi-Bellman approach to optimal trade execution. *Appl. Numer. Math.*, 61(2):241–265, 2011.

[42] J. Garcke and A. Kröner. Suboptimal feedback control of PDEs by solving HJB equations on adaptive sparse grids. *J. Sci. Comput.*, 70(1):1–28, 2017.

[43] A. Gnoatto, A. Picarelli, and C. Reisinger. Deep xva solver – a neural network based counterparty credit risk management framework, 2020. arXiv preprint: 2005.02633.

[44] S. N. Gomes and G. A. Pavliotis. Mean field limits for interacting diffusions in a two-scale potential. *J. Nonlinear Sci.*, 28(3):905–941, 2018.

[45] A. Gorodetsky, S. Karaman, and Y. Marzouk. High-dimensional stochastic optimal control using continuous tensor decompositions. *The International Journal of Robotics Research*, 37(2-3):340–377, 2018.

[46] L. Grüne and A. Rantzer. On the infinite horizon performance of receding horizon controllers. *IEEE Trans. Automat. Control*, 53(9):2100–2111, 2008.

[47] L. Grüne. Computing lyapunov functions using deep neural networks, 2020. arXiv preprint: 2005.08965.

[48] J. Han, A. Jentzen, and W. E. Solving high-dimensional partial differential equations using deep learning. *Proc. Natl. Acad. Sci. USA*, 115(34):8505–8510, 2018.

[49] R. Herzog and K. Kunisch. Algorithms for PDE-constrained optimization. *GAMM-Mitt.*, 33(2):163–176, 2010.

[50] M. B. Horowitz, A. Damle, and J. W. Burdick. Linear hamilton jacobi bellman equations in high dimensions. In *53rd IEEE Conference on Decision and Control*, pages 5880–5887, 2014.

[51] K. Ito, C. Reisinger, and Y. Zhang. A neural network-based policy iteration algorithm with global h^2-superlinear convergence for stochastic games on domains. *Foundations of Computational Mathematics*, 2020.

[52] K. Ito and J. Schroeter. Reduced order feedback synthesis for viscous incompressible flows. *Mathematical and Computer Modelling*, 33(1):173 – 192, 2001. Computation and control VI proceedings of the sixth Bozeman conference.

[53] D. Kalise, S. Kundu, and K. Kunisch. Robust feedback control of nonlinear PDEs by numerical approximation of high-dimensional Hamilton-Jacobi-Isaacs equations. *SIAM J. Appl. Dyn. Syst.*, 19(2):1496–1524, 2020.

[54] D. Kalise and K. Kunisch. Polynomial approximation of high-dimensional Hamilton-Jacobi-Bellman equations and applications to feedback control of semilinear parabolic PDEs. *SIAM J. Sci. Comput.*, 40(2):A629–A652, 2018.

[55] W. Kang, Q. Gong, and T. Nakamura-Zimmerer. Algorithms of data development for deep learning and feedback design, 2019. arXiv preprint:1912.00492.

[56] W. Kang and L. Wilcox. *A Causality Free Computational Method for HJB Equations with Application to Rigid Body Satellites*. 2015. AIAA Guidance, Navigation, and Control Conference.

[57] W. Kang and L. C. Wilcox. Mitigating the curse of dimensionality: sparse grid characteristics method for optimal feedback control and HJB equations. *Comput. Optim. Appl.*, 68(2):289–315, 2017.

[58] A. Krener, C. Aguilar, and T. Hunt. Series solutions of HJB equations. In K. Hüper and J. Trumpf, editors, *Mathematical System Theory – Festschrift in Honor of Uwe Helmke on the Occasion of his Sixtieth Birthday*, pages 247–260. CreateSpace, 2013.

[59] K. Kunisch and D. Walter. Semiglobal optimal feedback stabilization of autonomous systems via deep neural network approximation, 2020. arXiv preprint:2002.08625.

[60] L. Laurent, R. Le Riche, B. Soulier, and P.-A. Boucard. An overview of gradient-enhanced metamodels with applications. *Archives of Computational Methods in Engineering*, 26(1):61–106, Jan 2019.

[61] D. Lukes. Optimal regulation of nonlinear dynamical systems. *SIAM Journal on Control*, 7(1):75–100, 1969.

[62] W. M. McEneaney. A curse-of-dimensionality-free numerical method for solution of certain HJB PDEs. *SIAM J. Control Optim.*, 46(4):1239–1276, 2007.

[63] T. Nakamura-Zimmerer, Q. Gong, and W. Kang. Adaptive deep learning for high-dimensional hamilton-jacobi-bellman equations, 2019. arXiv preprint:1907.05317.

[64] N. Nüsken and L. Richter. Solving high-dimensional hamilton-jacobi-bellman pdes using neural networks: perspectives from the theory of controlled diffusions and measures on path space, 2020. arXiv preprint:2005.05409.

[65] M. Oster, L. Sallandt, and R. Schneider. Approximating the stationary hamilton-jacobi-bellman equation by hierarchical tensor products, 2019. arXiv preprint:1911.00279.

[66] H. J. Pesch and M. Plail. The maximum principle of optimal control: a history of ingenious ideas and missed opportunities. *Control Cybernet.*, 38(4A):973–995, 2009.

[67] L. S. Pontryagin, V. G. Boltyanskii, R. V. Gamkrelidze, and E. F. Mishchenko. *The mathematical theory of optimal processes*. Translated from the Russian by K. N. Trirogoff; edited by L. W. Neustadt. Interscience Publishers John Wiley & Sons, Inc. New York-London, 1962.

[68] H. Rauhut and R. Ward. Interpolation via weighted ℓ_1 minimization. *Appl. Comput. Harmon. Anal.*, 40(2):321–351, 2016.

[69] M. Reble and F. Allgöwer. Unconstrained model predictive control and suboptimality estimates for nonlinear continuous-time systems. *Automatica J. IFAC*, 48(8):1812–1817, 2012.

[70] E. Sarmin and L. Chudov. On the stability of the numerical integration of systems of ordinary differential equations arising in the use of the straight line method. *USSR Computational Mathematics and Mathematical Physics*, 3(6):1537 – 1543, 1963.

[71] E. Stefansson and Y. P. Leong. Sequential alternating least squares for solving high dimensional linear hamilton-jacobi-bellman equation. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3757–3764, 2016.

[72] K. Worthmann, M. Reble, L. Grüne, and F. Allgöwer. The role of sampling for stability and performance in unconstrained nonlinear model predictive control. *SIAM J. Control Optim.*, 52(1):581–605, 2014.

[73] I. Yegorov and P. M. Dower. Perspectives on characteristics based curse-of-dimensionality-free numerical approaches for solving hamilton–jacobi equations. *Appl. Math. Optim.*, Jul 2018.

[74] I. Yegorov, P. M. Dower, and L. Grüne. Synthesis of control lyapunov functions and stabilizing feedback strategies using exit-time optimal control, 2019. arXiv preprint:1906.02703.