Weierstraß-Institut
für Angewandte Analysis und Stochastik

Leibniz-Institut im Forschungsverbund Berlin e. V.

Preprint ISSN 2198-5855

Dynamical low–rank approximations of solutions to the Hamilton–Jacobi–Bellman equation

Martin Eigel[1], Reinhold Schneider[2], David Sommer[1]

submitted: November 30, 2021

[1] Weierstrass Institute
Mohrenstr. 39
10117 Berlin
Germany
E-Mail: martin.eigel@wias-berlin.de
david.sommer@wias-berlin.de

[2] Technische Universität Berlin
Institut für Mathematik
Straße des 17. Juni 136
10587 Berlin
Germany
E-Mail: schneidr@math.tu-berlin.de

No. 2896
Berlin 2021

2020 *Mathematics Subject Classification.* 49L20, 49M37, 93-08, 93B52, 15A69.

Key words and phrases. Dynamical low-rank approximation, feedback control, Hamilton-Jacobi-Bellman, variational Monte Carlo, tensor product approximation.

Martin Eigel acknowledges the partial support of the DFG SPP 1886 “Polymorphic Uncertainty Modelling for the Numerical Design of Structures ”. David Sommer acknowlegdes support by the ProFIT project ReLkat – Reinforcement Learning for complex automation engineering”.

Edited by
Weierstraß-Institut für Angewandte Analysis und Stochastik (WIAS)
Leibniz-Institut im Forschungsverbund Berlin e. V.
Mohrenstraße 39
10117 Berlin
Germany

Fax: +49 30 20372-303
E-Mail: preprint@wias-berlin.de
World Wide Web: http://www.wias-berlin.de/

Dynamical low–rank approximations of solutions to the Hamilton–Jacobi–Bellman equation

Martin Eigel, Reinhold Schneider, David Sommer

Abstract

We present a novel method to approximate optimal feedback laws for nonlinar optimal control basedon low-rank tensor train (TT) decompositions. The approach is based on the Dirac-Frenkel variationalprinciple with the modification that the optimisation uses an empirical risk. Compared to currentstate-of-the-art TT methods, our approach exhibits a greatly reduced computational burden whileachieving comparable results. A rigorous description of the numerical scheme and demonstrations ofits performance are provided.

1 Introduction

Feedback control is ubiquitous and indispensable in real dynamical systems. Since the controlled system can in general not be expected to follow model predictions exactly, system trajectories will eventually leave the forecasted path, meaning that any preplanned series of controls (albeit an optimal one) is based on wrong assumptions and therefore not only suboptimal, but potentially dangerous. As an illustration, one might think of an astronaut who calculated an optimal course to land on the moon but then does not modify the forecasted actuation values of their rocket-drive when atmospheric effects steer them off said course, which will leave them drifting to outer space. It is therefore vital to deploy controls based on *current* state feedback, where *current* means as frequently as possible in practical applications.

However, The problem of computing an optimal feedback control law for nonlinear optimal control problems is notoriously difficult. This is because the synthesis of such a feedback law requires solving the Hamilton-Jacobi-Bellman (HJB) equation, which is a nonlinear parabolic partial differential equation (PDE) of generally high dimension $d \gg 1$ [BC97]. Classical schemes to solve the HJB equation such as Galerkin-schemes in linear ansatz spaces suffer from the *curse of dimensionality* [KK18], i.e. an exponential complexity growth. In practice, this means that the computation of a solution is often infeasibly slow if it can be discretized and stored at all. Another severe obstacle can be the low regularity of viscosity solutions, cf [BD+97]. In this paper, our focus lies on the alleviation of the curse of dimensionality in order to enable the numerical treatment of high-dimensional control problems. We hence only consider problems where the lack of regularity is not present or not pronounced enough to prevent a sufficiently accurate approximation.

The relevance of efficient numerical methods can be seen by the fact that true feedback control methods - that is: methods solving the HJB equation - are rarely used in practice due to the necessary computational effort. Control problems arising e.g. in mechanical engineering often require new planning of controls to be computed within seconds. There hence is a tight upper limit on the time budget available for generating new controls. Therefore, most engineers deploy variations of Model Predictive Control (MPC) where open-loop controls are computed in such rapid succession that they effectively “close the loop” [CA13]. This is a conservative approach since the feedback property of the resulting controller is

DOI 10.20347/WIAS.PREPRINT.2896 Berlin 2021

obtained solely by means of the measurements at the discrete planning steps. In between two state measurements, the controller is not in feedback form.

In this work, we present a novel method to tackle nonlinear optimal control problems that (1) yields a true feedback controller and (2) has greatly reduced computational cost compared to current state-of-the-art methods. Our method is based on policy iteration, linearizing the HJB equation (which is then sometimes called the generalised Hamilton-Jacobi-Bellman or GHJB equation) and a modification of the Dirac-Frenkel variational principle. This then allows the computation of approximate solutions on a specified function manifold, for which we choose the set of multivariate polynomials with a fixed tensor train (TT) rank.

Tree based tensor networks and tensor trains in particular have already been used for successful approximations of the value function in various works, see e.g. [Fac+20; KKD19; OSS21a]. These recent results are summarized in the PhD thesis of Leon Sallandt [Sal21], which is still being finalised as this paper is written. There, the approach is based on a Lagrangian (or dynamic programming) perspective by computing the value functions for several initial states and learning the global function from these values by regression using a multi-polynomial TT model. With appropriate modifications, this approach can already be combined with regression techniques performed e.g. by machine learning methods, in particular artificial neural networks (NN). In the present paper, we follow a different approach, exploiting the Riemannian structure of the TT manifold [HRS12a; Ste16] by an empirical version of the Dirac-Frenkel principle.

The solution obtained by the abstract Dirac-Frenkel principle can be shown to be quasi-optimal in some time interval $[0, T_{\mathsf{DF}}]$ but detoriates from the best low-rank approximation after a certain time [Lub+13]. We expect a similar behaviour in our case which may restrict the time interval in practice. Combining both approaches - abstract and empirical - is an open research question, which we aim to address in future work. Similarly, we defer the stochastic control case to a forthcoming paper, confining ourselves to deterministic control in this paper. We conjecture that the present approach is even more advantageous in the stochastic case.

The rest of the paper is organised as follows: In Section 2 we provide a short overview of the related literature, specifically the current state-of-the-art of tensor based methods to solve the HJB equation. Section 3 introduces the finite horizon optimal control problem in feedback form, which the rest of this work revolves around. In Section 4 the tensor train format, the corresponding manifold and the representation of the tangent space are introduced. These are needed to formulate the Dirac-Frenkel variational principle, which is introduced in its abstract form in Section 5. In Section 6, we combine the concepts of Sections 4 and 5 to develop our proposed DLRA method for approximately solving the HJB equation. Numerical results that illustrate the practical performance are presented and discussed in Section 7. Finally, we close in Section 8 with an outlook on future work.

2 Related work

The Bellman equation governing the value function of an optimal control (OC) problem was introduced as early as 1957 by Richard Bellman [Bel57]. Since then, numerous sophisticated methods have been introduced to approximate solutions, mostly based on the principle of dynamical programming, see e.g. [Ber05] for a broad introduction to the subject. The alternative approach, which we follow in this work, is to consider the infinitesimal version of the Bellman equation, namely the Hamilton-Jacobi-Bellman equation [BC97], which is a nonlinear parabolic PDE. In both cases, many methods rely on a fixed point iteration of the equation, which in the OC and Reinforcement Learning (RL)

literature is called *policy iteration* [How60]. Alternatives are domain splitting algorithms [FLS94], semi-Lagrangian methods [Fal87; FK14; TAK17], data-based methods using Neural Networks [Luo+14], variational iterative methods [KDK13], actor-critic methods [ZH21], tree-based methods [AS19] and tropical methods [AF18; AGL08].

For a fixed starting value, an optimal control can be obtained by open-loop approaches such as Pontryagin's maximum principle [BGP61; Pon87]. In this way, the value function can be evaluated pointwise by simply adding up the cost of that optimal control. Controls of this type have been used to find the value function e.g. in [AKK21; KW17; NGK19; OSS21a]. In this work, we use optimal open-loop controls as benchmarks to which we compare the feedback controller computed by our method.

Since any solution method for the HJB equation has to deal with the curse of dimensionality, some form of model order reduction has to take place in practical applications. Possible function approximations can be obtained by using neural networks [DLM19; IRZ21; NR21] or sparse polynomials [AKK21]. In this work we use the TT format introduced to the mathematical community by Oseledets [Ose11; OT09] for multivariate polynomials. A striking recent example of the power of the low rank TT structure for function approximation can be found in [RSN21], in which the authors use TTs with polynomial basis functions to outperform state-of-the-art NNs on the solution of parabolic PDEs by orders of magnitude, while requiring lower computational time. For further details on TTs and more general hierarchical tensor networks, we refer the reader to the survey articles [BSU16; HS14] and the standard textbooks [Hac12; Hac14]. For recent applications of TTs as value function approximators, see e.g. [KKD19; OSS21a]. Solution methods based on high-dimensional polynomials and tensor spaces have also been considered in [DKK21; KK18]. As a conjecture for future work, block sparsity of the TTs appearing in optimal control methods could be exploited to further reduce the sample complexity [TSG21].

In contrast to the aforementioned methods, our new approach is a *dynamical low rank approximation* (DLRA) [KL07; KL10] of the value function. The main idea is to approximate solutions to matrix- or tensor-valued ordinary differential equations (ODEs) by projecting the right-hand side onto the tangent space of the manifold of matrices/tensors of fixed (TT-)rank at the current approximation. In this abstract setting, the projection is usually decomposed into orthogonal parts of the tangent space after which a splitting scheme is applied, resulting in so called projector-splitting schemes [CKL21; CL20; KLW16; LOV15; Lub+13]. The obtained approximation is quasi-optimal on a finite time domain, a property known as the Dirac-Frenkel variational principle, or Dirac-Frenkel/McLachlan variational principle [McL64; Mur35]. DLR approximations to parabolic PDEs have been studied in [Bac+21; Con20], but – to the best of our knowledge – this work is the first application of DLR methods to a finite horizon optimal control problem and in particular to the nonlinear HJB equation. In order to derive an abstract DLR problem on the TT manifold, we use a Variational Monte Carlo (VMC) approach [Bay+21; EST20]. In our setting it can be understood as an empirical least squares tensor regression based on random samples.

3 The optimal control problem

Throughout this work, we consider a deterministic dynamical system

$$\dot{x}(t) = f(t, x(t)) + g(t, x(t))u(t), \quad t \in [t_0, T], \tag{1}$$

$$x(t_0) = x_0, \tag{2}$$

with initial time $t_0 \in [0, T]$, initial condition $x_0 \in \Omega \subset \mathbb{R}^d$, control $u \in L^2(0, T; \mathbb{R}^m)$, free dynamics $f : [0, T] \times \Omega \to \Omega$ and control interface $g : [0, T] \times \Omega \to \mathbb{R}^{d \times m}$. To ensure existence and uniqueness

of solutions (for admissible controls u), we assume f and g to be smooth (possibly nonlinear) functions. A total cost is associated with the triple $(t_0, x_0, u) \in [0,T] \times \Omega \times L^2(0,T;\mathbb{R}^m)$ in terms of the *cost functional*

$$\mathcal{J}(t_0, x_0, u) = \int_{t_0}^{T} c(t, x(t)) + u(t)^\intercal R(t) u(t) \mathrm{d}t + c_T(x(T)), \tag{3}$$

where the running cost $c : [0,T] \times \Omega \to \mathbb{R}_+$ and the terminal cost $c_T : \Omega \to \mathbb{R}_+$ are non-negative, coercive and smooth functionals. Moreover, $R : [0,T] \to \mathbb{R}^{m\times m}$ is continuous, $R(t)$ is positive definite for all t, and the trajectory $x(\cdot)$ is subject to (1)+(2) with the given control u. The function mapping time-state pairs to optimal future costs is called the *value function*. It is canonically defined as

$$V^* : [0,T] \times \Omega \to \mathbb{R}, \quad (t_0, x_0) \mapsto \inf_{u \in L^2(0,T;\mathbb{R}^m)} \mathcal{J}(t_0, x_0, u).$$

If the dynamics and cost terms satisfy sufficient regularity conditions, the value function is given as the viscosity solution of the well known Hamilton-Jacobi-Bellman equation.

Theorem 1 (see e.g. [BC97; BD+97])**.** *Let $\ell(t,x,u) = c(t,x) + u^\intercal R(t) u$ and assume there are $\sigma, \delta \geq 1$ with $\sigma < \delta$, $\ell_0 > 0$. Moreover, for every compact $K \subset \mathbb{R}^d$ there exists some $f_K > 0$ such that*

$$\begin{aligned} \|f(x,u)\| &\leq f_K(1 + \|x\|^\sigma), \quad \text{for all } (x,u) \in K \times \mathbb{R}^m, \\ |\ell(x,u)| &\geq \ell_0 \|a\|^\delta \quad \text{for all } (x,u) \in \mathbb{R}^d \times \mathbb{R}^m. \end{aligned}$$

Then, the value function V^* is the unique viscosity solution of the HJB equation

$$\frac{\partial}{\partial t} V^*(t,x) + \min_{u \in \mathbb{R}^m} \left[\nabla_x V^*(t,x)^\intercal (f(t,x) + g(t,x)u) + c(t,x) + u^\intercal R(t) u \right] = 0 \tag{4}$$

$$V^*(T, \cdot) = c_T(\cdot). \tag{5}$$

Note that the HJB equation is an infinitesimal version of the Bellman equation, which we state for the sake of completeness.

Theorem 2 ([BC97; BD+97])**.** *For all $x_0 \in \Omega$ and $0 \leq t_0 \leq t_1 \leq T$, we have*

$$V^*(t_0, x_0) = \inf_{u \in L^2(t_0, t_1; \mathbb{R}^m)} \left[\int_{t_0}^{t_1} \ell(t, x(t), u(t)) \mathrm{d}t + V^*(t_1, x(t_1)) \right], \tag{6}$$

where $x(t)$ satisfies (1) *with initial condition $x(t_0) = x_0$ and control u.*

Now consider feedback controls of the form $u(t) = \alpha(t, x(t))$, where $\alpha : [0,T] \times \Omega \to \mathbb{R}^m$ is continuous on $[0,T]$ and Lipschitz in Ω. We call such functions α *admissible feedback laws* (or equivalently *admissible policies*) and we denote the set of admissible policies by $\mathcal{A}$. Next, we define the policy evaluation function $\mathcal{J}_\alpha$ via the associated cost

$$\mathcal{J}^\alpha(t_0, x_0) = \int_{t_0}^{T} c(t, x(t)) + \alpha(t, x(t))^\intercal R(t) \alpha(t, x(t)) \mathrm{d}t + c_T(x(T)). \tag{7}$$

An *optimal policy* α^* is a policy which achieves minimal costs for any starting values, i.e.

$$\mathcal{J}^{\alpha^*}(t_0, x_0) = \min_{\alpha \in \mathcal{A}} \mathcal{J}^\alpha(t_0, x_0), \quad \text{for all } (t_0, x_0) \in [0,T] \times \Omega.$$

The goal of optimal (feedback) control is to approximate such an optimal policy. If the value function is known and partially differentiable, an optimal policy can be obtained immediately, as the following theorem states.

Theorem 3 ([BC97])**.** *An optimal policy is given by*

$$\alpha^*(t,x) = -\frac{1}{2}R(t)^{-1}g(t,x)^\intercal \nabla_x V^*(t,x), \tag{8}$$

if the gradient of V^* exists.

Hence, the problem of synthesizing an optimal policy is the problem of finding the value function, which involves solving the HJB equation (4) or the Bellman equation (6).

In the following, we always assume that the conditions of Theorem 1 are satisfied so that that an optimal policy is given by the value function via (8). Hence, we can identify the value function with the policy evaluation function of that optimal policy denoted by $V^* = \mathcal{J}^{\alpha^*}$. Our goal is to approximate the value function successively on small subintervals, moving backwards in time from $t = T$ to $t = 0$. This approach is based on Bellman's principle. However, in contrast to comparable recent work [OSS21a], we use the HJB equation (4) on each subinterval instead of the Bellman equation. In particular, we define suitable approximate solutions to the HJB equation by means of the Dirac-Frenkel variational principle. While theoretical simplicity is lost to some extend, computational simplicity is gained in return. This is mainly because DLR approximations of (4) can be computed very efficiently since samples do not have to be propagated through the dynamics to evaluate the integral in (6).

Assume now that $0 = t_0 < t_1 < \dots t_m = T$ is an equidistant discretisation and consider a partitioning $\{[t_i, t_{i+1}]\}_{i=0}^m$ of the time interval $[0,T]$. An immediate consequence of Bellman's principle is that an optimal policy for the whole time domain $[0,T]$ must also be optimal on any subinterval $[t_i, t_{i+1}]$. Conversely, a policy that is optimal on all subintervals is also optimal on the whole interval. This enables to learn the value function by moving backwards in time and (approximately) computing the restrictions $V^*(t,x)\big|_{t\in[t_i,t_{i+1}]}$ for $i = m-1, \dots, 0$. In the following, we denote by $V_i^* = V^*\big|_{[t_i,t_{i+1}]}$ for $i = 0, \dots, m-1$ the restrictions of the value function to a particular subinterval and set $V_m^* = c_T$. Approximations of V_i^* are denoted by $\hat{V}_i$ and the approximation $\hat{V}$ of V^* on the whole time domain is defined by $\hat{V} = \hat{V}_i$ on $[t_i, t_{i+1}]$. Algorithm 1 summarizes the idea of successive backward approximation, which we deploy to approximate the value function.

Algorithm 1: Bellman-based backwards scheme to approximate the value function

Data: Time discretisation points $0 = t_0 < \dots < t_m = T$, approximation $\hat{V}_m$ of the terminal cost.
Result: Approximation $\hat{V}$ of the value function.
for $i = m-1, m-2, \dots, 0$ **do**
  Compute approximate solution $\hat{V}_i$ of the HJB-eq. (4) on $[t_i, t_{i+1}]$ with terminal condition $\hat{V}_{i+1}$.
  Set $\hat{V} = \hat{V}_i$ on $[t_i, t_{i+1}]$.
end

TT approximations of the value function by means of such a backwards scheme were already presented e.g. in [OSS21a]. In that work however, the integral formulation (6) is used exclusively, sampling trajectories $x(t)$ for given controls and adding up the costs. In contrast, the DLR approximation method used here allows to directly work with the HJB equation (4).

4 Tensor trains as function approximators

For practical computations, the approximations $\hat{V}_i$ from Algorithm 1 have to be confined to a finite-dimensional functions space. To this end, consider a set of one-dimensional basis functions $\phi_1, \dots, \phi_n$:

$\mathbb{R} \to \mathbb{R}$ and functions $v : \mathbb{R}^d \to \mathbb{R}$ of the form

$$v(x) = A\phi(x) = \sum_{i_1,\dots,i_d=1}^{n} A_{i_1,\dots,i_d} \phi_{i_1}(x_1) \cdot \ldots \cdot \phi_{i_d}(x_d), \tag{9}$$

with coefficient tensor $A \in \mathbb{R}^{n\times n\times\dots\times n}$ of order d. Usually, the basis functions ϕ_i are (orthonormal) polynomials. Consequently, v is a multivariate polynomial with a storage complexity of $\mathcal{O}(n^d)$ for its coefficient tensor. The TT format provides a possiblity to alleviate this exponential complexity by assuming some low-rank structure. A TT representation of A is any decomposition of the form

$$A_{i_1,\dots,i_d} = U^1_{i_1} \cdot \ldots \cdot U^d_{i_d}, \tag{10}$$

where

$$U^1 \in \mathbb{R}^{n\times r_1},\ U^\mu \in \mathbb{R}^{r_{\mu-1}\times n\times r_\mu} \text{ for } i = 2,\dots,d-1,\ U^d \in \mathbb{R}^{r_{d-1}\times n}$$

are called the *components* of the representation and i_μ denotes the middle index of the component, i.e. $U^\mu_{i_\mu} \in \mathbb{R}^{r_{\mu-1}\times r_\mu}$. The rank of the specific representation is given by the tuple $(r_1,\dots,r_{d-1})$. The TT-rank $\mathbf{r} = (r_1,\dots,r_{d-1})$ of A is defined as the (entry-wise) minimal rank tuple such that a TT representation (10) with the corresponding ranks exists. Such a minimal TT representation exists for any tensor. In fact, the minimal rank entry r_μ is equal to the matrix rank of the μ-th unfolding of A (for details we refer to [HRS12b]). The TT representation exhibits a storage complexity of $\mathcal{O}(nd\max(r_1,\dots,r_{d-1})^2)$, scaling only linearly in the dimension d, and hence avoiding the curse of dimensionality, provided that the ranks stay bounded. It is important to note that even for fixed rank $\mathbf{r}$, a decomposition of the form (10) is not unique. For any $\mu = 1,\dots,d-1$ we can set $U_\mu \to U_\mu S$ and $U_{\mu+1} \to S^{-1}U_{\mu+1}$ for invertible $S \in \mathbb{R}^{r_\mu\times r_\mu}$ without changing the tensor. A unique representation is then given by requiring *left-* and *right-orthogonality* of the components in the sense of the following definition.

Definition 1. *For a component $U_\mu \in \mathbb{R}^{r_{\mu-1}\times n\times r_\mu}$, define the left and right unfolding*

$$L(U^\mu) \in \mathbb{R}^{r_{\mu-1}n\times r_\mu}, \qquad R(U^\mu) \in \mathbb{R}^{r_{\mu-1}\times r_\mu n},$$

by suitable matrix reshaping (for details regarding the order, see e.g. [Ste16]). A component U^μ is called left- or right-orthogonal if

$$L(U^\mu)^\intercal L(U^\mu) = I_d \in \mathbb{R}^{r_\mu\times r_\mu}, \quad \text{or} \quad R(U^\mu)R(U^\mu)^\intercal = I_d \in \mathbb{R}^{r_{\mu-1}\times r_{\mu-1}},$$

respectively. A TT representation $A_{i_1,\dots,i_d} = U^1_{i_1} \cdot \ldots \cdot U^d_{i_d}$ of a tensor A is called μ-orthogonal if $U^1,\dots,U^{\mu-1}$ are left orthogonal and $U^{\mu+1},\dots,U^d$ are right orthogonal. In that case, U^μ is called the core of the representation.

Left and right orthogonality of all but one component imposes $\sum_{\mu=1}^{d-1} r_\mu^2$ additional conditions on the representation. Hence, the μ-orthogonal TT representation of A is unique for any μ.

For a given TT rank $\mathbf{r}$, we define the set

$$\mathcal{M}_\mathbf{r} = \{A \in \mathbb{R}^{n\times\dots\times n} : A \text{ has TT rank } \mathbf{r}\}.$$

It is noteworthy that $\mathcal{M}_\mathbf{r}$ is a smooth manifold in $\mathbb{R}^{n\times\dots\times n}$ [HRS12b]. With a chosen suitable basis $\{\phi_1,\dots,\phi_n\}$, we define a set of function approximations

$$\mathfrak{F}_\mathbf{r} = \left\{v : \mathbb{R}^d \to \mathbb{R} : v \text{ admits a representation (9) with } A \in \mathcal{M}_\mathbf{r}\right\}.$$

Note that by identification of a function with its coefficient tensor, $\mathfrak{F}_{\mathbf{r}}$ forms a smooth manifold in the n^d-dimensional linear space $\mathfrak{F} = \bigcup_{\mathbf{r}} \mathfrak{F}_{\mathbf{r}}$ in the same way that $\mathcal{M}_{\mathbf{r}}$ forms a smooth manifold in $\mathbb{R}^{n\times\ldots\times n}$. In order to do perform an optimisation on $\mathfrak{F}_{\mathbf{r}}$, or $\mathcal{M}_{\mathbf{r}}$, respectively, we require a representation of the tangent space $\mathcal{T}_U(\mathcal{M}_{\mathbf{r}})$ of $\mathcal{M}_{\mathbf{r}}$ in U. Throughout this work, we use the following representation.

Theorem 4 ([HRS12b] or [Ste16])**.** *Let $U \in \mathcal{M}_{\mathbf{r}}$ be d-orthogonal. The tangent space $\mathcal{T}_U(\mathcal{M}_{\mathbf{r}})$ of $\mathcal{M}_{\mathbf{r}}$ in the point U is given by $\tau(X)$, where*

$$\begin{aligned} X &= U_1^\ell \times \ldots \times U_{d-1}^\ell \times \mathbb{R}^{r_{d-1}\times n\times r_d}, \\ U_\mu^\ell &= \{W^\mu \in \mathbb{R}^{r_{\mu-1}\times n\times r_\mu} : L(U^\mu)^\intercal L(W^\mu) = 0 \in \mathbb{R}^{r_\mu\times r_\mu}\}, \end{aligned}$$

and

$$\begin{aligned} &\tau : X \longrightarrow \mathcal{T}_U(\mathcal{M}_{\mathbf{r}}), \quad \tau(W^1,\ldots,W^d) = \delta U \\ &\delta U_{i_1,\ldots,i_d} = \sum_{\mu=1}^{d} U^1_{i_1}\cdot\ldots\cdot U^{\mu-1}_{i_{\mu-1}} W^\mu_{i_\mu} U^{\mu+1}_{i_{\mu+1}}\cdot\ldots\cdot U^d_{i_d}. \end{aligned} \tag{11}$$

The tangent space has the same dimension as the underlying manifold. The previously mentioned ambiguity in the representation is now eliminated due to the *gauging conditions* $L(U^\mu)^\intercal L(W^\mu) = 0$ in U^ℓ_μ.

Corollary 1. *Each of the spaces U^ℓ_μ has dimension $r_{\mu-1}nr_\mu - r_\mu^2$ and hence the tangent space has dimension*

$$n_X \coloneqq \dim(X) = \sum_{\mu=1}^{d} r_{\mu-1}nr_\mu - \sum_{\mu=1}^{d-1} r_\mu^2.$$

Using the representation (11) for elements δU of the tangent space of U, a simple form for the sum $U + \delta U$ can be obtained.

Lemma 1 (see [Ste16])**.** *Let $U \in \mathcal{M}_{\mathbf{r}}$ be d-orthogonal and denote its component tensors by $U_1,\ldots,U_d$. Let $\delta U \in \mathcal{T}_U(\mathcal{M}_{\mathbf{r}})$ be given by $(\delta U_1,\ldots,\delta U_d) \in X$. Then,*

$$\begin{aligned} U(i_1,\ldots,i_d) + \delta U(i_1,\ldots,i_d) = \begin{bmatrix}\delta U_1(i_1) & U_1(i_1)\end{bmatrix} \begin{bmatrix} U_2(i_2) & 0 \\ \delta U_2(i_2) & U_2(i_2)\end{bmatrix} \cdots \\ \cdots \begin{bmatrix} U_{d-1}(i_{d-1}) & 0 \\ \delta U_{d-1}(i_{d-1}) & U_{d-1}(i_{d-1})\end{bmatrix} \begin{bmatrix} U_d(i_d) \\ U_d(i_d) + \delta U_d(i_d)\end{bmatrix}. \end{aligned}$$

This can easily be verified by multiplying out the matrix products. In particular, the sum $U + \delta U$ has at most TT-rank $2\mathbf{r}$.

5 The Dirac-Frenkel variational principle

The Dirac-Frenkel variational principle [Mur35] provides a principled way to approximate tensor valued ODEs of the form

$$\dot{A}(t) = F(t, A(t)), \tag{12}$$
$$A(0) = A_0, \tag{13}$$

where $A(t) \in \mathbb{R}^{n\times\dots\times n}$, on the manifold $\mathcal{M}_{\mathbf{r}}$. More precisely, given an approximation $Y_0 \in \mathcal{M}_{\mathbf{r}}$ of the initial condition A_0, an approximation $Y(t) \in \mathcal{M}_{\mathbf{r}}$ of $A(t)$ is defined as the solution of the TT-valued ODE

$$\dot{Y}(t) = \operatorname*{arg\,min}_{\vartheta\in\mathcal{T}_{\mathcal{M}_{\mathbf{r}}}(Y(t))} \|\vartheta - F(t, Y(t))\|, \tag{14}$$

$$Y(0) = Y_0. \tag{15}$$

The minimum in (14) is attained by the orthogonal projection of the right-hand side onto the tangent space, leading to

$$\dot{Y}(t) = P_{\mathcal{T}_{\mathcal{M}_r}(Y(t))}F(t, Y(t)). \tag{16}$$

In this abstract setting, error bounds can be derived, which we quote for the sake of completeness.

Theorem 5. *[Lub+13] Suppose that $\dot{A}(t) \leq \mu$ and that a continuously differentiable best approximation $X(t) \in \mathcal{M}_{\mathbf{r}}$ to $A(t)$ exists for $t \in [0, T]$. Let $\delta > 0$ be such that the smallest nonzero singular value of every matrix unfolding of $X(t)$ is greater or equal to ρ, and assume that the best-approximation error is bounded by $\|X(t) - A(t)\| \leq c\rho$ for $t \in [0, T]$ with a constant c depending only on the dimension d. Then, the approximation error of the dynamical low-rank approximation defined by* (20) *with initial value $Y(0) = X(0)$ is bounded by*

$$\|Y(t) - X(t)\| \leq 2\beta e^{\beta t}\int_0^t \|X(s) - A(s)\|\mathrm{d}s,$$

with $\beta = C\mu\rho - 1$ for $t \in [0, T]$, as long as the right-hand side remains bounded by $c\rho$. The constant C is only dependent on d and is given in [Lub+13].

In recent years there have been numerous works on the numerical treatment of ODEs of this type, see [KLW16; KL07; LO13] for an introduction in the matrix case and [CKL21; CL20; LOV15] for more recent tensor-based research directions. Generally, these methods rely on a splitting of the projector $P_{\mathcal{M}_{\mathbf{r}}(Y(t))}$ into orthogonal parts of the tangent space, so-called *projector splitting algorithms*. The norm $\|.\|$ governing (14) and hence the projector is usually the Frobenius norm. This is in contrast to our work, where $\|.\|$ is an empirical norm[1]. Carrying over results from the treatment of the abstract Dirac-Frenkel principle to the empirical case (specifically the projector splitting schemes) is an important direction of future work, that we do not yet address in this paper.

6 Dynamical low-rank approximation of the HJB equation

Based on the preceding review of tools that we require, we now return to the HJB equation (4) on $[t_i, t_{i+1}]$ with terminal condition $\hat{V}_{i+1}(t_{i+1}, \cdot)$. The goal is to obtain an approximation $\hat{V}_i$ of the value function on the current interval. Inserting (8) into the HJB (4) leads to a coupled problem:

Find V such that

$$\frac{\partial}{\partial t}V(t,x) + \nabla_x V(t,x)^\intercal(f(t,x) + g(t,x)\alpha(t,x)) + c(t,x) + \alpha(t,x)^\intercal R(t)\alpha(t,x) = 0, \tag{17}$$

$$V(t_{i+1}, \cdot) = \hat{V}_{i+1}(t_{i+1}, \cdot), \tag{18}$$

[1] the details of which are provided in the next chapter

where α satisfies

$$\alpha(t,x) = -\frac{1}{2}R(t)^{-1}g(t,x)^\intercal \nabla_x V(t,x). \tag{19}$$

To compute $\hat{V}_i$, we use a fixed point iteration of the coupled problem, iteratively solving (17)+(18) for fixed α and then updating α via (19). This procedure is known as *policy iteration* in the optimal control literature. We depict a conceptual summary in algorithm 2. If the solutions to (17) are exact, it converges under mild assumptions on dynamics and cost terms [SL79]. In order to track the convergence of the scheme under approximations, we introduce on $L^2((t_i,t_{i+1});L^2(\Omega,\rho))$ and $L^2((0,T);L^2(\Omega,\rho))$ the norms

$$\|v\|_i^2 = \int_{t_i}^{t_{i+1}} \|v(t,\cdot)\|^2_{L^2(\Omega,\rho)}\mathrm{d}t, \qquad \|v\|^2 = \sum_{i=0}^{m-1}\|v\|_i^2,$$

and stop the iteration once the $\|.\|_i$-difference of two consecutive approximations becomes smaller than a specified threshold.

Algorithm 2: Policy iteration on subinterval

Data: Interval $[t_i,t_{i+1}]$, terminal condition $\hat{V}_{i+1}$, admissible policy α, error tolerance δ.
Result: Approximation $\hat{V}_i$.
while norm change of $\hat{V}_i > \delta$ **do**
  Compute approximate solution $\hat{V}_i$ of (17)+(18) on $[t_i,t_{i+1}]$.
  Update $\alpha \propto \nabla_x \hat{V}_i$ according to (19).
end

It remains to be shown how to compute the approximations $\hat{V}_i$. To ease notation, without loss of generality we consider the interval $[t_0,t_1]$ instead of $[t_i,t_{i+1}]$ for the remainder of this chapter. We construct $\hat{V}_0$ as a dynamical low-rank approximation of (17) in the tensor train format.

Let the terminal condition $\hat{V}_1 \in \mathfrak{F}_{\mathbf{r}}$ and consider for given α the following problem:

Find V such that

$$\frac{\partial}{\partial t}V(t,\cdot) = \operatorname*{arg\,min}_{\vartheta \in \mathcal{T}_{V(t,\cdot)}(\mathfrak{F}_{\mathbf{r}})} \|\vartheta - \tilde{F}(t,V(t,\cdot))\|^2_{L^2(\Omega,\rho)}, \tag{20}$$

$$V(t_1,\cdot) = \hat{V}_1(t_1,\cdot), \tag{21}$$

where $\tilde{F}(t,V(t,\cdot))(\cdot) = -\nabla_x V(t,\cdot)^\intercal(f(t,\cdot)+g(t,\cdot)\alpha(t,\cdot)) - c(t,\cdot) + \alpha(t,\cdot)^\intercal R(t)\alpha(t\cdot)$. Note that this essentially means that the time derivative of V_0 is approximated in the tangent space of the current solution. By a simple time inversion $t \to t_0 + (t_1 - t)$, the terminal condition can be turned into an initial condition. Crucially, any solution to (20) stays on the manifold $\mathfrak{F}_{\mathbf{r}}$ and can therefore be identified with a time-dependent coefficient tensor $A(t) \in \mathcal{M}_{\mathbf{r}}$ via $V(t,x) = A(t)\phi(x)$. Denoting the coefficient tensor of $\hat{V}_1$ by $\hat{A}_1$, we see that the abstract problem (20)+(21) is equivalent to the TT-valued ODE

$$\dot{A}(t) = \operatorname*{arg\,min}_{B \in \mathcal{T}_{A(t)}(\mathcal{M}_{\mathbf{r}})} \|B\phi - F(t,A(t)\phi)\|^2_{L^2(\Omega,\rho)}, \tag{22}$$

$$A(t_0) = A_1, \tag{23}$$

where F arises from time inversion of $\tilde{F}$.

In general, the L^2-integral on the right-hand side of (22) is difficult to compute. Nevertheless, we can easily carry out a pointwise evaluation of the basis functions ϕ as well as the other terms in $F(t, A(t)\phi)$. In practice, we hence replace the exact L^2-norm with a Monte Carlo approximation

$$\|v\|^2_{L^2(\Omega,\rho,M)} = \frac{1}{M}\sum_{k=1}^{M} |v(x_k)|^2, \qquad x_k \sim \rho,$$

for $v \in L^2(\Omega, \rho)$. This turns the right-hand side of the ODE into an empirical risk minimisation. We eventually arrive at

$$\dot{A}(t) = \operatorname*{arg\,min}_{B\in\mathcal{T}_{A(t)}(\mathcal{M}_{\mathbf{r}})} \frac{1}{M}\sum_{k=1}^{M} |B\phi(x_k) - F(t, A(t)\phi(x_k))(x_k)|^2, \qquad t \in (t_0, t_1) \tag{24}$$

$$A(t_0) = \hat{A}_1. \tag{25}$$

Statistical bounds for the error of the empirical minimiser in (24) compared to the best L^2-approximation $\Phi^*(t) = \arg\min_{\Phi\in L^2(\Omega,\rho)} \|\Phi - F(t, A(t)\phi(\cdot))(\cdot)\|^2_{L^2(\Omega,\rho)}$ are given in [Eig+19].

A crucial observation is that the minimisation on the right-hand side is a linear problem since the optimisation is over the linear tangent space. Implementation details on how the minimum in (24) for a given t can be computed are given in Appendix A. Since the fit is linear, issues of local minima are avoided which for instance occur in the alternating linear scheme (ALS) [HRS12b] and other nonlinear optimisation methods. Alternating methods can still be applied here to divide the problem into smaller sub-problems and reduce the computational burden, leading (in their simplest form) to an effective Lie-Trotter type splitting of the right hand side. A more detailed examination of this topic is however beyond the scope of this paper and might be addressed in future work.

The numerical realisation of (24) poses an additional hurdle. While the true solution $A(t)$ always stays on the manifold $\mathcal{M}_{\mathbf{r}}$, it is straightforward to see that any one step with a numerical integrator, e.g. a Runge-Kutta method, leads to leaving it. This is due to the fact that by Lemma 1 any sum $U + \delta U$ where $U \in \mathcal{M}_{\mathbf{r}}$ and $\delta U \in \mathcal{T}_U(\mathcal{M}_{\mathbf{r}})$ has rank $2\mathbf{r}$ in general. We therefore need to retract back onto the manifold after each step of the integrator by truncating the ranks appropriately. To make this precise, let $t_0 = t^{(0)} < t^{(1)} < t^{(2)} < \ldots < t^{(L)} = t_1$ be a *micro-discretisation* of the *macro-interval* $[t_0, t_1]$ with equidistant step size τ and define a numerical approximation A_ℓ of $A(t^{(\ell)})$ by the explicit Euler scheme

$$\begin{aligned} A_0 &= \hat{A}_1, \\ A_{\ell+1} &= \mathcal{R}(A_\ell + \tau\Delta A_\ell), \qquad \ell = 0\ldots, L-1. \end{aligned}$$

Here, ΔA_ℓ is the solution to the minimisation problem on the right-hand side of (24) if A_ℓ is substituted for $A(t)$, the addition $A_\ell + \tau\Delta A_\ell$ is performed like in Lemma 1, and $\mathcal{R}$ denotes the rank-truncation of a TT with rank $2\mathbf{r}$ back to a tensor of rank $\mathbf{r}$. This truncation is performed by a TT-SVD with fixed rank [OT09]. Once all A_ℓ are obtained in this way, we define $\hat{V}_0(t, x)$ by linear interpolation, i.e.

$$\hat{V}_0(t, x) = A_\ell\phi(x) + \frac{t - t^{(\ell)}}{\tau}(A_{\ell+1} - A_\ell)\phi(x) \quad \text{for} \quad t \in [t^{(\ell)}, t^{(\ell+1)}],$$

or by simply always setting it to

$$\hat{V}_0(t, x) = A_\ell\phi(x) \quad \text{for} \quad t \in [t^{(\ell)}, t^{(\ell+1)}].$$

Now, if $\hat{V}_0^{\text{old}}$ is the approximation from the previous policy iteration step, one could compute the empirical approximation to the $\|.\|_0$-norm

$$\begin{aligned}\|\hat{V}_0 - \hat{V}_0^{\text{old}}\|_{0,L,M}^2 &= \frac{1}{L-1}\sum_{\ell=0}^{L-1} \|\hat{V}_0(t^{(\ell)},\cdot) - \hat{V}_0^{\text{old}}(t^{(\ell)},\cdot)\|_{L^2(\Omega,\rho,M)}^2 \\ &= \frac{1}{(L-1)M}\sum_{\ell=0}^{L-1}\sum_{k=1}^{M} |\hat{V}_0(t^{(\ell)},x_k) - \hat{V}_0^{\text{old}}(t^{(\ell)},x_k)|^2\end{aligned}$$

and stop the iteration once this norm difference becomes smaller than the threshold δ. However, since we are first and foremost interested in obtaining a nearly optimal control α, we instead add the change in the controls $\alpha \propto \nabla_x \hat{V}_0$ and $\alpha^{\text{old}} \propto \nabla_x \hat{V}_0^{\text{old}}$ and stop the iteration once

$$\frac{1}{(L-1)M}\sum_{\ell=0}^{L-1}\sum_{k=1}^{M} |\hat{V}_0(t^{(\ell)},x_k) - \hat{V}_0^{\text{old}}(t^{(\ell)},x_k)|^2 + \|\alpha(t^{(\ell)},x_k) - \alpha^{\text{old}}(t^{(\ell)},x_k)\|^2 < \delta. \quad (26)$$

The reason for this is that the L^2-norm is agnostic to errors in the gradients, which may arise due to overfitting. By requiring (26), we demand that not only $\hat{V}_0$ but also the relevant part of the gradient $\nabla_x \hat{V}_0$ converges. In that sense, the left-hand side of (26) can be seen as an empirical approximation of an H^1-norm of $\hat{V}_0 - \hat{V}_0^{\text{old}}$, where the norms for the gradients are now weighted by R and g to represent only the gradient parts relevant for the control.

7 Numerical tests

This chapter is concerned with numerical experiments that illustrate the performance of the proposed DLR approximation[2]. We consider a problem of the form

$$\dot{x} = Ax + \mathrm{nl}(x) + gu,$$

where $x \in \mathbb{R}^d$, $A \in \mathbb{R}^{d\times d}$, $g \in \mathbb{R}^d$, u is scalar and nl is a smooth nonlinear function with $\mathrm{nl}(0) = 0$. In particular, the optimal control problem is derived from a modified one dimensional heat equation

$$\begin{aligned}\frac{\partial}{\partial t}x(s,t) &= \sigma\frac{\partial^2}{\partial s^2}x(s,t) + x(s,t)^3 + g(s)u(t), && \text{for } (s,t) \in [-1,1]\times(0,T),\\ x(s,0) &= \tilde{x}_0(s), && \text{for } s \in [-1,1],\\ \frac{\partial}{\partial s}x(-1,t) &= \frac{\partial}{\partial s}x(1,t) = 0 && \text{for } t \in (0,T),\end{aligned}$$

with unstable reaction term $x(s,t)^3$, diffusion $\sigma > 0$, scalar control u and initial state $\tilde{x}_0$. Note that due to the instability introduced by the reaction term, this problem is generally more difficult to control than most other canonically treated examples like viscous Burgers' type equations, Allen-Kahn or degenerate Zeldovich equations [KK18; OSS21a] since the quadratic regulator usually provides a strong and mostly stable controller for these types of problems. This however is not the case for the nonlinear reaction problem defined above. We hence omit the mentioned alternative examples and just note that our method can be applied with them as well, although the difference to the linear quadratic regulator would turn out to be small.

[2] All computations are carried out on an Intel Xeon Gold 6154 CPU 3.00GHz, openSUSE Leap 15.2 distribution.

Our goal is to find a control u such that the quadratic cost functional

$$\tilde{J}(0, x_0, u) = \int_0^T \|x(\cdot, t)\|^2_{L^2(\Omega)} + \gamma u(t)^2 \,\mathrm{d}t + c_T \|x(\cdot, T)\|_{L^2(\Omega)},$$

is minimal with $\gamma, c_T > 0$. A semi-discretization of the PDE with finite differences at d equidistant points $-1 = s_1 < \ldots < s_d = 1$ leads to a an ODE of the form

$$\dot{x} = Ax + x^3 + gu, \tag{27}$$
$$x(0) = x_0, \tag{28}$$

with $x_0 = (\tilde{x}_0(s_1), \ldots, \tilde{x}_0(s_n))^\intercal$, $x(t) \in \mathbb{R}^d$, $g = (g(s_1), \ldots, g(s_d))^\intercal \in \mathbb{R}^d$ and $A \in \mathbb{R}^{d\times d}$ is given by

$$A = \frac{\sigma}{h^2} \begin{pmatrix} -2 & 2 & & & \\ 1 & -2 & 1 & & \\ & \ddots & \ddots & \ddots & \\ & & 1 & \ddots & 1 \\ & & & 2 & -2 \end{pmatrix}, \quad h = s_1 - s_0 = \frac{2}{d-1}.$$

The x-dependent term in the cost functional can be approximated using a simple quadrature rule with nodes $s_1, \ldots, s_d$ (here, we use the rectangle rule with an additional node at the last grid point s_d). This yields the new cost functional

$$J(0, x_0, u) = \int_0^T x(t)^\intercal Q x(t) + \gamma u(t)^2 \,\mathrm{d}t + c_T x(T)^\intercal Q x(T), \tag{29}$$

where

$$Q = h \begin{pmatrix} 1 & & \\ & \ddots & \\ & & 1 \end{pmatrix},$$

and $x(t)$ is understood to be the solution of $\dot{x} = Ax + x^3 + gu$ with starting value x_0. The control problem is now to find a control u for the nonlinear system (27) such that (29) is minimal for every starting value x_0.

To specify the control problem, we choose the parameters $\sigma = 1$, $\gamma = 0.1$, $c_T = 1$ and $g = \chi_{[-0.4,0.4]}$ and discretise with $n = 12$ equidistant grid points. The time horizon is $T = 0.3$ and the time step size $\tau = t_{i+1} - t_i$ is 0.001, which is used for both the macro-intervals as well as the micro-intervals of the policy iteration (see Section 6). The same step size is also used to discretise the integral in (29) when computing the costs. As a threshold for the policy iteration, $\delta = 10^{-6}$ is set. We choose $\Omega = (-2, 2)^d$ and let ρ be the uniform distribution on Ω. For the TT approximations, we use the first n $H^2_{\mathsf{mix}}(\Omega)$-orthonormal polynomials $\phi_1, \ldots, \phi_n$ as basis functions (up to degree $n-1$). Here, $H^2_{\mathsf{mix}}(\Omega)$ denotes the tensorised space $\bigotimes_{\mu=1}^d H^2((-2,2))$, H^2 is the Sobolev space of twice weakly differentiable functions. We set $n = 9$, yielding a maximal polynomial degree of 8 in the basis. The rank of the TT manifold is chosen to be

$$\mathbf{r} = (3, 5, 5, 5, 5, 5, 5, 5, 5, 5, 3).$$

Note that by this the dimension of the approximation space is reduced from $n^d = 9^{12} > 282$ trillion to a manageable number of degrees of freedom $\leq nd \max_{\mu=1,\dots,d}(r_\mu)^2 = 2700$. The number of sample points used to approximate the L^2-norm in (24) is chosen as

$$M = 6 \cdot nd \max_{\mu=1,\dots,d}(r_\mu)^2 = 6 \cdot 8 \cdot 12 \cdot 5^2 = 16200,$$

which is a generous upper bound for the number of degrees of freedom of the fit.

As a benchmark for assessing the performance of our method, we use the TT-based approach from [OSS21a] with the same hyper-parameters. To make this precise, instead of solving (24) by means of our dynamical low-rank scheme, $\hat{V}_i$ is approximated in each policy iteration step by sampling the trajectories $x_k(t)$, $t \in [t_i, t_{i+1}]$ of all sample points. With this, the integrals

$$\hat{V}_i(t_i, x_k) = \int_{t_i}^{t_{i+1}} \ell(t, x_k(t), \alpha(t, x_k(t))\mathrm{d}t + \hat{V}_{i+1}(t_{i+1}, x_k(t_{i+1}))$$

are evaluated subsequently. An approximation of $V_i(t_i, \cdot)$ is then obtained via a nonlinear fit of a rank-$\mathbf{r}$ TT to the resulting data-target pairs $(x_k, y_k = \hat{V}_i(t_i, x_k))_{k=1}^M$, which is performed by the ALS. Note that the authors in [OSS21a] suggest replacing the upper integral bound t_{i+1} with t_{i+l}, $l > 1$, where the trajectory on $[t_{i+1}, t_{i+l}]$ is controlled by the already computed (nearly optimal) controls from previous steps, to remove the error associated with $\hat{V}_{i+1}$ from the computation of $\hat{V}_i$. Since this greatly increases the computational complexity, we stick with the above mentioned "one-step scheme" and refer to this benchmark method as the *Bellman* method, since it explicitly utilises Bellman's principle in the form of the terminal cost $\hat{V}_{i+1}$. Our method, utilising Dynamical Low Rank Approximation, will be called the DLRA method. Even for the DLRA method we have found it beneficial for stable convergence to compute some $\hat{V}_i$ with the Bellman method before starting the dynamical low rank solver. In this example the first 10 of the 300 approximations are computed in this way.

Remark 1. *In both the nonlinear fit required for the Bellman method and the linear fit of our DLRA method, we add a regularisation term $\delta\|\hat{V}_i(t_i, \cdot)\|^2_{H^2_{\mathsf{mix}}(\Omega)}$ to the minimisation functional. Due to the multilinear structure of the TT and our choice of the basis functions as $H^2_{\mathsf{mix}}(\Omega)$-orthonormal, this leads to local minimisation problems of the form*

$$\min_{\mathbf{c}} \|M\mathbf{c} - \mathbf{y}\|_2^2 + \delta\|\mathbf{c}\|_F^2$$

in ALS (compare to [OSS21b]). Here, $\mathbf{c} \in \mathbb{R}^{r_{\mu-1} \times n_\mu \times r_\mu}$ denotes the core that is currently optimised and $\|.\|_F$ denotes the Frobenius norm. In both methods, we use $\delta = 10^{-10}$ but in ALS we successively lower δ via

$$\delta \to \max(0.9, \|M\mathbf{c} - \mathbf{y}\|_2^2 / \|\mathbf{y}\|_2^2) \cdot \delta$$

after every sweep. This is a purely heuristical rule to make sure the regularisation is relaxed once the attractor of the global minimum is found.

Remark 2. *For the DLRA method we add an additional regularisation term*

$$\delta_0 |B\phi(0)|^2$$

to the minimisation in (24) *since we know that the right-hand side satisfies $F(t, A(t)\phi(0)) = 0$. Note that this can be realised by simply adding the point $x_{M+1} = 0$ to the set of samples $\{x_k\}_{k=1}^M$. Since this is a hard constraint on the true solution, we set $\delta_0 = 10^{10}$.*

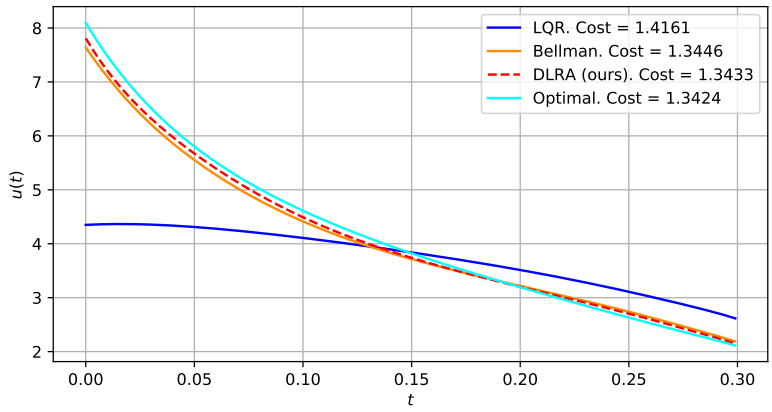

Figure 1: Control values $u(t)$ of the different controllers along the trajectory of a fixed polynomial initial condition x_0.

As a second, classical benchmark, we consider the linear quadratic regulator (LQR), resulting from linearising the problem around $x = 0$. Since this controller does not see the unstable reaction term, we expect poor performance compared to both the Bellman and the DLRA method.

To compare the practical performance of the methods, two different sets of initial conditions $\tilde{x}_0$ are generated. For the first set, we sample a polynomial degree between 2 and 20 and then again randomly sample the coefficients of a univariate polynomial of that degree. Denoting this polynomial p, we then set $\tilde{x}_0(s) = (s-1)^2(s+1)^2 p(s)$ to make sure $\tilde{x}_0$ satisfies the Neumann boundary conditions. Finally, in order to have interesting trajectories (27) for which the x^3-term requires strong control beyond LQR, we normalise such that $\max_{s\in[-1,1]} |\tilde{x}_0(s)| = 1.9$. The second set of initial conditions is generated by simply setting $\tilde{x}_0(s) \equiv c$ for constants $c \in [1, 2)$.

Figures 1 and 3 show the control values $u(t)$ along one trajectory of each type of initial conditions. Figures 2 and 4 depict the mean costs over 500 randomly sampled initial conditions in each of the two cases, where we have omitted those initial conditions for which the open-loop solver used to compute the optimal control did not converge. Examining the graphs, we note that the Bellman method and the DLRA method achieve similar, almost optimal performance over the chosen test sets. Interestingly, the DLRA method actually slightly outperforms the full Bellman method and is often closer to the optimal control trajectories, which for instance can be seen in Figure 3. We attribute this to the generalisation error of the Bellman method: even if the value function approximation should be more accurate – due to a projection directly onto the manifold – the associated optimisation is nonlinear and may get stuck in local optima. In the DLRA method, we avoid this problem by coping only with linear minimisation problems.

7.1 Computational cost and a hybrid approach

The distinct advantage of the DLRA method is its greatly reduced computational cost. Table 1 contains the computation times for the two methods (Bellman and DLRA), as well as their mean costs on the set of polynomial initial conditions, with the same hyper-parameters and maximal polynomial degrees of 4, 6 and 8, respectively. We observe that the two methods achieve comparable performance for degrees 6 and 8. However, the DLRA method achieves this performance in roughly one tenth of the time that the Bellman method requires. We stress again that the version we used is the *fastest* version of the

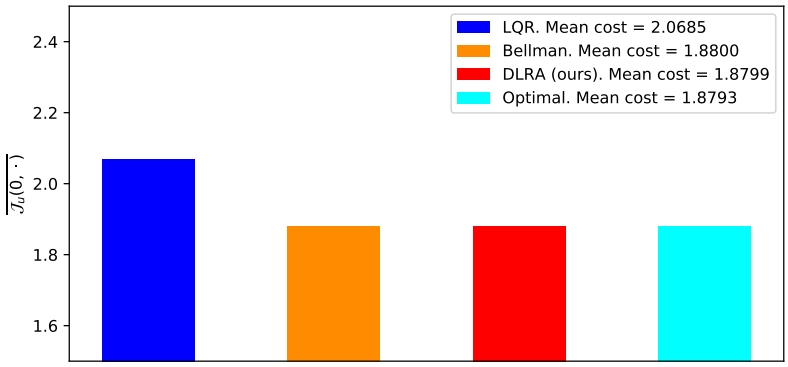

Figure 2: Sample-mean costs $\overline{\mathcal{J}_u(0,\cdot)} \approx \frac{1}{N}\sum_k^N J_u(0, x_0^{(k)})$ of polynomial initial conditions $x_0^{(k)}$ with the different controllers.

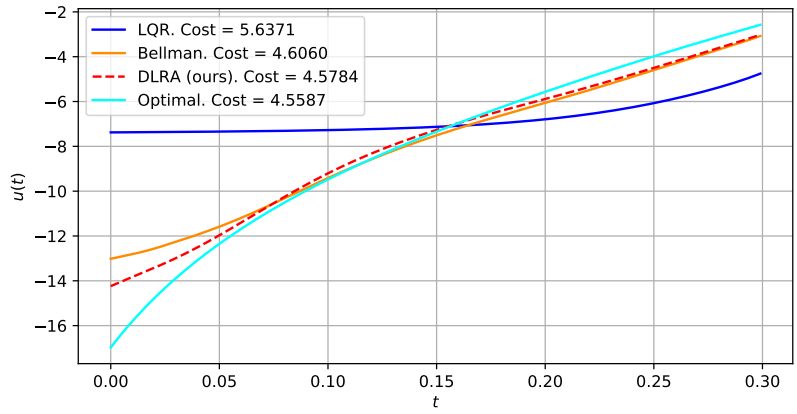

Figure 3: Control values $u(t)$ of the different controllers along the trajectory of a fixed uniform initial condition $x_0 = c \cdot (1, \ldots, 1)^\intercal$. In this example $c = 1.28$ is used.

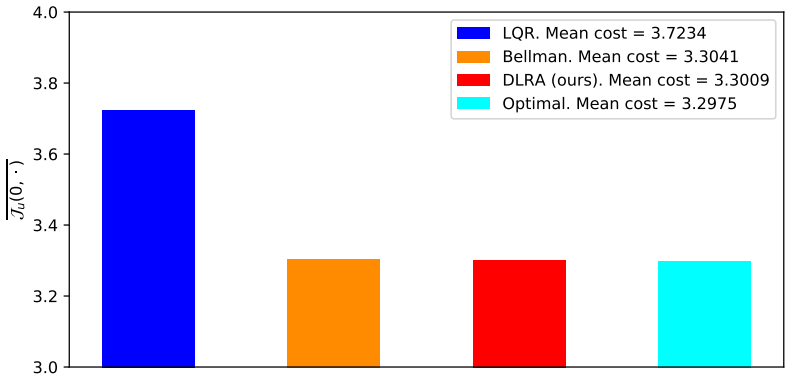

Figure 4: Sample-mean costs $\overline{\mathcal{J}_u(0,\cdot)} \approx \frac{1}{N}\sum_k^N J_u(0, x_0^{(k)})$ of uniform initial conditions $x_0^{(k)}$ with the different controllers.

Bellman method available, since we employ the one-step scheme. As discussed in [OSS21a; OSS21b], this method also suffers from error propagation due to a large number of time steps. Moreover our proposed method projects onto the tangent space, whereas Bellmann always tries to project onto the tensor manifold.

The DLRA method performs significantly worse for a lower polynomial degree of 4. We attribute this to an effect that can be seen already for degree 8 in Figure 3. The DLRA controller drifts away from the true optimal control the further it moves away from the terminal time $t = T$. This error seems to originate from two main factors: for one, the true value function $V^*(t, \cdot)$ successively moves further away from the manifold $\mathcal{M}_{\mathbf{r}}$ even if the terminal condition satisfies $c_T \in \mathcal{M}_{\mathbf{r}}$. To visualise that the true solution does not stay on the manifold, the relative norm error of the last tangent fit in each policy iteration is plotted over time in Figure 5. Note that these errors should be close to 0 if the solution to the GHJB equation is an element of the manifold. Instead, the errors increase monotonically over time. The second major source of error is the retraction after every Euler step. Both sources of errors get worse for lower degrees because of the restricted manifold. Hence, a degree of 4, which is perfectly feasible for the Bellman method, produces bad results with the DLRA method. Note that the observed behaviour is expected.

	Bellman		DLRA		Hybrid	
	comp. time	mean cost	comp. time	mean cost	comp. time	mean cost
pol. deg. 4	3078.44	1.8822	333.29	2.6147	909.65	1.8804
pol. deg. 6	4270.33	1.8801	421.52	1.8802	1851.93	1.8798
pol. deg. 8	5967.91	1.8800	499.96	1.8799	–	–

Table 1: Computation time of the methods in seconds as well as mean costs of polynomial initial conditions for different maximal polynomial degrees of the basis functions. The mean optimal cost is 1.8793.

This observation leads to a natural formulation of a hybrid method, possibly alleviating the main weaknesses of both methods. These are the high computational complexity for the Bellman method and error accumulation for the DLRA method. The hybrid method uses DLRA updates but after each m steps, instead of computing $\hat{V}_i$ with the regular DLRA update, it performs a full Bellman update [OSS21a] with an m-step scheme

$$\hat{V}_i(t_i, x_k) = \int_{t_i}^{t_{i+m}} \ell(t, x_k(t), \alpha(t, x_k(t))dt + \hat{V}_{i+m}(t_{i+m}, x_k(t_{i+m})). \tag{30}$$

For $m = 1$ this method is equivalent to the Bellman method, for m greater than the number of total time steps it is equivalent to the DLRA method. For any intermediate m it periodically performs one costly but accurate Bellman update in between fast DLRA updates. Since the maximal number of consecutive DLRA steps is now m, the DLRA solver is prevented from drifting too far away from the real solution, before being corrected again by the Bellman update, yielding a new (more accurate) initial condition. Note in particular that the evaluation of (30) does not include any $\hat{V}_j$ computed with the DLRA method. Hence, after every m steps, the accumulated error of the DLRA steps is reset to 0. Globally, only the error of the m-step Bellman updates (30) accumulates.

The results for the hybrid method with $m = 10$ are depicted in Table 1 for degrees 4 and 6. We remark that for polynomials of degree 4, the hybrid scheme provides an essential improvement with respect to accuracy when compared to both Bellmann and DLRA. There is an improvement for degree 6 but

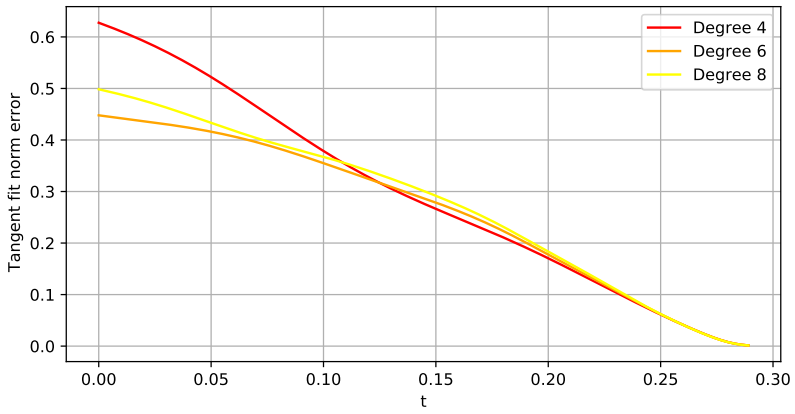

Figure 5: Relative norm error of the last tangent fit (see Appendix A) in each policy iteration step of the DLRA solver over time and for different polynomial degrees.

compared to DLRA this effect is not pronounced. Surprisingly, for a sufficiently accurate model, DLRA alone was sufficiently accurate. The case of degree 8 is omitted since the DLRA controller is already nearly optimal in that case. The periodic 10-step Bellman updates with intermediate DLRA steps are sufficient to outperform the full 1-step Bellman method, but at much lower computational costs. From the perspective of the DLRA method, the periodic Bellman updates enable the use of more restricted manifolds.

8 Concluding remarks

In this paper we present a novel method to approximate optimal feedback laws for optimal control problems. The proposed method utilizes a tensor train compression to break the curse of dimensionality of a multivariate polynomial ansatz space. Moreover, it employs an empirical version of the Dirac-Frenkel variational principle to solve the HJB equation. The method was tested numerically on a canonical benchmark example which is difficult to control with standard methods, and demonstrated to achieve near optimal performance with greatly reduced computation time compared to state-of-the-art methods.

In the experiments it comes as no surprise that the proposed method works quite well for short time intervals. However, it is striking that we can also observe that with a sufficiently good model – meaning an adequate polynomial degree in our case – the method even performs well on a large time horizon. Although we have not considered infinite horizon problems yet, as long as we know stabilizing controls, the present approach probably is applicable as well. Moreover, for large time horizons we have presented a robust hybrid method.

We would like to point out that the present successful approach strongly exploits the explicit knowledge about the geometry of the considered model class, i.e. (multi-)polynomial tensor trains in our setting. This advantage is something which cannot be easily transferred to a neural network setting.

We expect the method to also perform favourably with higher dimensional problems, which might be a future research topic. We predict that this will require some form of rank adaptivity to retain the computational advantage over state-of-the-art methods while achieving similar levels of accuracy. Rank adaptivity can be incorporated very naturally in the proposed DLRA method: instead of the full retraction onto the manifold $\mathcal{M}_{\mathbf{r}}$ after every step of the solver, one could round the TT based on an adaptive

threshold. Analysing the effect of a changing manifold on the Dirac-Frenkel variational principle might be an interesting topic for future work.

As a second direction, the method could be applied to stochastic optimal control problems. There, the GHJB equation (17) gets an additional Laplacian term $\propto \Delta_x V(t,x)$, turning it into a Kolmogorov-Backward type equation. Equations of this type for instance govern the time development of observables of Itô diffusion processes. The application of our method to such problems is currently being investigated.

A Details of the empirical risk minimisation

We detail how to reduce the minimisation in (24) to a standard system of linear equations. To achieve this, we use the characterization of the tangent space given by Theorem 4 and represent an element $\delta U \in \tau(X)$ of the tangent space as a vector $\mathbf{x} \in \mathbb{R}^{n_X}$. The first step towards this representation is the parametrisation of the spaces U_μ^ℓ.

A.1 A parametrisation of the tangent space

By Theorem 4, U_μ^ℓ is precisely the set of all $r_{\mu-1}n_\mu \times r_\mu$-matrices whose columns are orthogonal to the columns of $L(U_\mu)$. Let $QR = L(U_\mu)$ be the QR decomposition and denote the orthonormal columns of Q by $q_1, \dots, q_{r_\mu}$. By the Gram-Schmidt procedure, we can expand the columns to an orthonormal basis of $\mathbb{R}^{r_{\mu-1}n_\mu}$ and denote the additional vectors by $\hat{q}_1 = q_{r_\mu+1}, \dots, \hat{q}_{r_{\mu-1}n_\mu - r_\mu} = q_{r_{\mu-1}n_\mu}$. Now, let $W_\mu \in U_\mu^\ell$ and denote its j-th column by w_j. Then there are coefficients $c_{j,k}$ such that

$$w_j = \sum_{k=1}^{r_{\mu-1}n_\mu - r_\mu} c_{j,k}\hat{q}_k.$$

In total we get $(r_{\mu-1}n_\mu - r_\mu)r_\mu$ coefficients $c_{j,k}$, which are stored in a vector

$$\begin{aligned}\mathbf{x}_\mu &= (c_{1,1}, \dots, c_{1,r_{\mu-1}n_\mu - r_\mu}, c_{2,1}, \dots, c_{2,r_{\mu-1}n_\mu - r_\mu}, \dots, \dots, c_{r_\mu,1}, \dots, c_{r_\mu, r_{\mu-1}n_\mu - r_\mu})^\intercal \\ &\in \mathbb{R}^{(r_{\mu-1}n_\mu - r_\mu)r_\mu} = \mathbb{R}^{r_{\mu-1}n_\mu r_\mu - r_\mu^2}.\end{aligned}$$

From now on we always identify an element of U_μ^ℓ with its coefficient vector $\mathbf{x}_\mu$. Elements of $C_d = \mathbb{R}^{r_{d-1}\times n_d \times r_d}$ are represented in the same manner with the only difference that the sum in each column representation goes from $k = 1$ to $r_{d-1}n_d$ and the $\hat{q}_i$ can be chosen as the canonical basis in $\mathbb{R}^{r_{d-1}n_d}$.

We eventually can represent an element of the tangent space $\delta U \in \tau(X)$ by the concatenation of its coefficients vectors,

$$\delta U \cong \mathbf{x} = (\mathbf{x}_1^\intercal, \dots, \mathbf{x}_d^\intercal)^\intercal \in \mathbb{R}^{n_X}.$$

Since this becomes important when solving the regression problem (24) on the tangent space later on, we define a “lift”

$$\mathcal{L}_\mu : \mathbb{R}^{r_{\mu-1}n_\mu r_\mu - r_\mu^2} \longrightarrow \mathbb{R}^{r_{\mu-1}n_\mu r_\mu},$$

which maps the coefficient vector of the gauged representation to the vectorised entries of the corresponding tensor in U_μ^ℓ. This is achieved by means of a $r_{\mu-1}n_\mu r_\mu \times (r_{\mu-1}n_\mu r_\mu - r_\mu^2)$-lifting matrix

$$Z_\mu = \begin{pmatrix} Q_\mu^\perp & \mathbf{0} & \dots & \dots & \mathbf{0} \\ \mathbf{0} & Q_\mu^\perp & \mathbf{0} & & \\ & & \ddots & & \\ & & & & \mathbf{0} \\ \mathbf{0} & & & \mathbf{0} & Q_\mu^\perp \end{pmatrix}, \qquad Q_\mu^\perp = [\hat{q}_1, \dots, \hat{q}_{r_{\mu-1}n_\mu - r_\mu}] \in \mathbb{R}^{r_{\mu-1}n_\mu \times (r_{\mu-1}n_\mu - r_\mu)}.$$

By construction, $Z_\mu \mathbf{x}_\mu$ is the concatenation of the columns of $L(W_\mu)$. Hence, $\mathcal{L}_\mu(\mathbf{x}_\mu)$ can be obtained by $\mathcal{L}_\mu(\mathbf{x}_\mu) = Z_\mu \mathbf{x}_\mu$.

A.2 Solving the system of linear equations

We examine problem (24) in a more general setting. Let $U \in \mathcal{M}_\mathbf{r}$ be d-orthogonal and consider the minimisation problem

$$\min_{T \in \mathcal{T}_U(\mathcal{M}_\mathbf{r})} \sum_{k=1}^M \left| T[x^{(k)}] - y^{(k)} \right|^2, \tag{31}$$

where $(x^{(k)}, y^{(k)})_{k=1}^M \subset \mathbb{R}^d \times \mathbb{R}$ is a set of data-target pairs. This leads to

$$\begin{aligned} &\min_{T \in \mathcal{T}_U(\mathcal{M}_\mathbf{r})} \sum_{k=1}^M |T[x^{(k)}] - y^{(k)}|^2 \\ &= \min_{W \in X} \sum_{k=1}^M |\tau(W)[x^{(k)}] - y^{(k)}|^2 \\ &= \min_{W \in X} \sum_{k=1}^M \left| \sum_{\mu=1}^d \left[\sum_{j_1,\dots,j_d}^{n_1,\dots,n_d} U_1(j_1) \cdot \dots \cdot W_\mu(j_\mu) \cdot \dots \cdot U_d(j_d) \phi_{j_1}(x_1^{(k)}) \dots \phi_{j_d}(x_d^{(k)}) \right] - y^{(k)} \right|^2 \\ &= \min_{W \in X} \|\mathfrak{O}(W) - \mathbf{y}\|_2^2, \end{aligned}$$

where $\mathbf{y} = (y^{(1)}, \dots, y^{(M)})^\intercal$ and the operator $\mathfrak{O} : X \longrightarrow \mathbb{R}^M$ is defined by

$$\mathfrak{O}(W) = \sum_{\mu=1}^d \mathfrak{C}_\mu(W_\mu), \quad \text{for} \quad W = (W_1, \dots, W_d), \tag{32}$$

with

$$\begin{aligned} &\mathfrak{C}_\mu : \mathbb{R}^{r_{\mu-1} \times n_\mu \times r_\mu} \longrightarrow \mathbb{R}^M, \\ &(\mathfrak{C}_\mu(W_\mu))_k = \left[\sum_{j_1,\dots,j_d}^{n_1,\dots,n_d} U_1(j_1) \cdot \dots \cdot W_\mu(j_\mu) \cdot \dots \cdot U_d(j_d) \phi_{j_1}(x_1^{(k)}) \dots \phi_{j_d}(x_d^{(k)}) \right]. \end{aligned}$$

Note that $\mathfrak{C}_\mu$ is a linear tensor operator in $\mathbb{R}^{M \times r_{\mu-1} \times n_\mu \times r_\mu}$, which we can transfer into a matrix $C_\mu \in \mathbb{R}^{M \times r_{\mu-1} n_\mu r_\mu}$ by successive unfolding

$$\mathbb{R}^{M \times r_{\mu-1} \times n_\mu \times r_\mu} \longrightarrow \mathbb{R}^{M \times r_{\mu-1} n_\mu \times r_\mu} \longrightarrow \mathbb{R}^{M \times r_{\mu-1} n_\mu r_\mu}.$$

At the first stage, the operator $\mathfrak{C}_\mu$ acts on a tensor $W_\mu \in U_\mu^\ell$. At the second stage, it acts on the left unfolding $L(W_\mu)$. And at the third stage, the matrix C_μ acts on the concatenation of the columns of $L(W_\mu)$. By the previous section, this concatenation is given by $Z_\mu \mathbf{x}_\mu$. We hence have $\mathfrak{C}_\mu(W_\mu) = C_\mu Z_\mu \mathbf{x}_\mu$, leading to

$$\mathfrak{O}(W) = \sum_{\mu=1}^{d} C_\mu Z_\mu x_\mu = A\mathbf{x},$$

where $A = [C_1 Z_1, \dots, C_{d-1} Z_{d-1}, C_d]$ (note that $Z_d \equiv I_d$). We have thus transformed (31) to a standard system of linear equations

$$\hat{\mathbf{x}} = \operatorname*{arg\,min}_{\mathbf{x}\in\mathbb{R}^{n_X}} \|A\mathbf{x} - \mathbf{y}\|^2 \iff A^\intercal A\hat{\mathbf{x}} = A^\intercal \mathbf{y}.$$

Once a solution is found by standard methods, we recover W from $\hat{\mathbf{x}}$ by reshaping of the component vectors $\mathbf{x}_\mu$.

Remark 3. *We would like to make two remarks about the implementation. First, note that the matrices Z_μ do not have to be stored in order to compute the product $C_\mu Z_\mu$ since we can compute*

$$\begin{aligned} C_\mu Z_\mu &= [C_\mu[0 : r_{\mu-1}n_\mu]| \dots |C_\mu[(r_\mu - 1)(r_{\mu-1}n_\mu) : r_\mu r_{\mu-1} n_\mu]] \cdot \mathsf{diag}(Q_\mu^\perp, \dots, Q_\mu^\perp) \\ &= [C_\mu[0 : r_{\mu-1}n_\mu]Q_\mu^\perp| \dots |C_\mu[(r_\mu - 1)(r_{\mu-1}n_\mu) : r_\mu r_{\mu-1} n_\mu]Q_\mu^\perp]. \end{aligned}$$

Second, note that U_μ^ℓ is 0-dimensional if $r_{\mu-1}n_\mu - r_\mu = 0$. In this case, the space consists only of the tensor of constant zeros $\mathbf{0} \in \mathbb{R}^{r_{\mu-1}\times n_\mu \times r_\mu}$ and hence $W_\mu = \mathbf{0}$. No basis coefficients $\mathbf{x}_\mu$ need to be computed. Consequently, the index μ can be skipped entirely during optimisation. By this, A and $\mathbf{x}$ become

$$\begin{aligned} A &= [C_1 Z_1, \dots, C_{\mu-1} Z_{\mu-1}, C_{\mu+1} Z_{\mu+1}, \dots, C_{d-1} Z_{d-1}, C_d], \\ \mathbf{x} &= (\mathbf{x}_1^\intercal, \dots, \mathbf{x}_{\mu-1}^\intercal, \mathbf{x}_{\mu+1}^\intercal, \dots, \mathbf{x}_d^\intercal)^\intercal. \end{aligned}$$

References

[AF18] Marianne Akian and Eric Fodjo. “Probabilistic Max-Plus Schemes for Solving Hamilton-Jacobi-Bellman Equations”. In: Jan. 2018, pages 183–209. ISBN: 978-3-030-01958-7. DOI: 10.1007/978-3-030-01959-4_9 (cited on page 3).

[AGL08] Marianne Akian, Stéphane Gaubert, and Asma Lakhoua. “The max-plus finite element method for solving deterministic optimal control problems: basic properties and convergence analysis”. In: *SIAM Journal on Control and Optimization* 47.2 (2008), pages 817–848 (cited on page 3).

[AS19] Alessandro Alla and Luca Saluzzi. *A HJB-POD approach for the control of nonlinear PDEs on a tree structure*. May 2019 (cited on page 3).

[AKK21] Behzad Azmi, Karl Kunisch, and Dante Kalise. *Optimal Feedback Law Recovery by Gradient-Augmented Sparse Polynomial Regression*. Jan. 2021 (cited on page 3).

[Bac+21] Markus Bachmayr, Henrik Eisenmann, Emil Kieri, and André Uschmajew. “Existence of dynamical low-rank approximations to parabolic problems”. In: *Mathematics of Computation* (2021) (cited on page 3).

[BSU16] Markus Bachmayr, Reinhold Schneider, and André Uschmajew. “Tensor Networks and Hierarchical Tensors for the Solution of High-Dimensional Partial Differential Equations”. In: *Found. Comput. Math.* 16.6 (Dec. 2016), pages 1423–1472. ISSN: 1615-3375. DOI: 10.1007/s10208-016-9317-9. URL: https://doi.org/10.1007/s10208-016-9317-9 (cited on page 3).

[BC97] Martino Bardi and Italo Capuzzo-Dolcetta. “Optimal Control and Viscosity Solutions of Hamilton-Jacobi-Bellman Equations”. In: 1997 (cited on pages 1, 2, 4, 5).

[BD+97] Martino Bardi, Italo Capuzzo Dolcetta, et al. *Optimal control and viscosity solutions of Hamilton-Jacobi-Bellman equations*. Volume 12. Springer, 1997 (cited on pages 1, 4).

[Bay+21] Christian Bayer, Martin Eigel, Leon Sallandt, and Philipp Trunschke. *Pricing high-dimensional Bermudan options with hierarchical tensor formats*. Mar. 2021 (cited on page 3).

[Bel57] Richard Bellman. *Dynamic Programming*. Dover Publications, 1957. ISBN: 9780486428093 (cited on page 2).

[Ber05] Dimitri P. Bertsekas. *Dynamic Programming and Optimal Control*. 3rd. Volume I. Belmont, MA, USA: Athena Scientific, 2005 (cited on page 2).

[BGP61] VG Boltyanskiy, Revaz Valer’yanovich Gamkrelidze, and Lev Semenovich Pontryagin. *Theory of optimal processes*. Technical report. JOINT PUBLICATIONS RESEARCH SERVICE ARLINGTON VA, 1961 (cited on page 3).

[CA13] Eduardo F Camacho and Carlos Bordons Alba. *Model predictive control*. Springer science & business media, 2013 (cited on page 1).

[CKL21] Gianluca Ceruti, Jonas Kusch, and Christian Lubich. *A rank-adaptive robust integrator for dynamical low-rank approximation*. 2021. arXiv: 2104.05247 [math.NA] (cited on pages 3, 8).

[CL20] Gianluca Ceruti and Christian Lubich. *An unconventional robust integrator for dynamical low-rank approximation*. 2020. arXiv: 2010.02022 [math.NA] (cited on pages 3, 8).

[Con20] Dajana Conte. “Dynamical low-rank approximation to the solution of parabolic differential equations”. In: *Applied Numerical Mathematics* 156 (2020), pages 377–384 (cited on page 3).

[DLM19] Jérôme Darbon, Gabriel Provencher Langlois, and Tingwei Meng. “Overcoming the curse of dimensionality for some Hamilton–Jacobi partial differential equations via neural network architectures”. In: *Research in the Mathematical Sciences* 7 (2019), pages 1–50 (cited on page 3).

[DKK21] Sergey Dolgov, Dante Kalise, and Karl Kunisch. *Tensor Decomposition Methods for High-dimensional Hamilton-Jacobi-Bellman Equations*. 2021. arXiv: 1908.01533 [math.OC] (cited on page 3).

[EST20] Martin Eigel, Reinhold Schneider, and Philipp Trunschke. “Convergence bounds for empirical nonlinear least-squares”. In: *arXiv preprint arXiv:2001.00639* (2020) (cited on page 3).

[Eig+19] Martin Eigel, Reinhold Schneider, Philipp Trunschke, and Sebastian Wolf. “Variational Monte Carlo—bridging concepts of machine learning and high-dimensional partial differential equations”. In: *Advances in Computational Mathematics* 45.5 (2019), pages 2503–2532 (cited on page 10).

[Fac+20] Konstantin Fackeldey, Mathias Oster, Leon Sallandt, and Reinhold Schneider. *Approximative Policy Iteration for Exit Time Feedback Control Problems driven by Stochastic Differential Equations using Tensor Train format*. 2020. arXiv: 2010.04465 [math.OC] (cited on page 2).

[Fal87] Maurizio Falcone. “A numerical approach to the infinite horizon problem of deterministic control theory”. In: *Applied Mathematics and Optimization* 15 (Feb. 1987), pages 1–13. DOI: 10.1007/BF01442644 (cited on page 3).

[FK14] Maurizio Falcone and Dante Kalise. “A high-order semi-Lagrangian/finite volume scheme for Hamilton-Jacobi-Bellman-Isaacs equations”. In: volume 443. Nov. 2014. ISBN: 978-3-662-45503-6. DOI: 10.1007/978-3-662-45504-3_10 (cited on page 3).

[FLS94] Maurizio Falcone, Piero Lanucara, and Alessandra Seghini. “A splitting algorithm for Hamilton-Jacobi-Bellman equations”. In: *Applied Numerical Mathematics* 15.2 (1994), pages 207–218 (cited on page 3).

[Hac12] Wolfgang Hackbusch. *Tensor Spaces and Numerical Tensor Calculus*. Volume 42. Jan. 2012. ISBN: 978-3-642-28026-9. DOI: 10.1007/978-3-642-28027-6 (cited on page 3).

[Hac14] Wolfgang Hackbusch. “Numerical tensor calculus”. In: *Acta numerica* 23 (2014), pages 651–742. ISSN: 1474-0508. DOI: 10.1017/S0962492914000087 (cited on page 3).

[HS14] Wolfgang Hackbusch and Reinhold Schneider. “Tensor spaces and hierarchical tensor representations”. In: *Extraction of quantifiable information from complex systems*. Springer, 2014, pages 237–261 (cited on page 3).

[HRS12a] Sebastian Holtz, Thorsten Rohwedder, and Reinhold Schneider. “On Manifolds of Tensors of Fixed TT-Rank”. In: *Numer. Math.* 120.4 (Apr. 2012), pages 701–731. ISSN: 0029-599X. DOI: 10.1007/s00211-011-0419-7. URL: https://doi.org/10.1007/s00211-011-0419-7 (cited on page 2).

[HRS12b] Sebastian Holtz, Thorsten Rohwedder, and Reinhold Schneider. “On manifolds of tensors of fixed TT-rank”. In: *Numerische Mathematik* 120 (2012), pages 701–731 (cited on pages 6, 7, 10).

[How60] R. A. Howard. *Dynamic Programming and Markov Processes*. Cambridge, MA: MIT Press, 1960 (cited on page 3).

[IRZ21] Kazufumi Ito, Christoph Reisinger, and Yufei Zhang. “A neural network based policy iteration algorithm with global H2-superlinear convergence for stochastic games on domains”. In: *Found. Comput. Math.* 21 (2021), pages 331–374 (cited on page 3).

[KDK13] B Kafash, A Delavarkhalafi, and SM Karbassi. “Application of variational iteration method for Hamilton–Jacobi–Bellman equations”. In: *Applied Mathematical Modelling* 37.6 (2013), pages 3917–3928 (cited on page 3).

[KK18] Dante Kalise and Karl Kunisch. “Polynomial Approximation of High-Dimensional Hamilton–Jacobi–Bellman Equations and Applications to Feedback Control of Semilinear Parabolic PDEs”. In: *SIAM Journal on Scientific Computing* 40.2 (Jan. 2018), A629–A652. ISSN: 1095-7197. DOI: 10.1137/17m1116635. URL: http://dx.doi.org/10.1137/17M1116635 (cited on pages 1, 3, 11).

[KKD19] Dante Kalise, Karl Kunisch, and Sergey Dolgov. *Tensor Decomposition Methods for High-dimensional Hamilton-Jacobi-Bellman Equations*. Aug. 2019 (cited on pages 2, 3).

[KW17] Wei Kang and Lucas C. Wilcox. “Mitigating the curse of dimensionality: sparse grid characteristics method for optimal feedback control and HJB equations”. In: *Computational Optimization and Applications* 68 (2017), pages 289–315 (cited on page 3).

[KLW16] Emil Kieri, Christian Lubich, and Hanna Walach. “Discretized Dynamical Low-Rank Approximation in the Presence of Small Singular Values”. In: *SIAM J. Numer. Anal.* 54 (2016), pages 1020–1038 (cited on pages 3, 8).

[KL07] Othmar Koch and Christian Lubich. “Dynamical Low-Rank Approximation”. In: *SIAM J. Matrix Anal. Appl.* 29 (2007), pages 434–454 (cited on pages 3, 8).

[KL10] Othmar Koch and Christian Lubich. “Dynamical tensor approximation”. In: *SIAM Journal on Matrix Analysis and Applications* 31.5 (2010), pages 2360–2375 (cited on page 3).

[LO13] Christian Lubich and Ivan Oseledets. “A projector-splitting integrator for dynamical low-rank approximation”. In: *BIT* 54 (Jan. 2013). DOI: 10.1007/s10543-013-0454-0 (cited on page 8).

[LOV15] Christian Lubich, Ivan Oseledets, and Bart Vandereycken. “Time Integration of Tensor Trains”. In: *SIAM Journal on Numerical Analysis* 53 (Jan. 2015), pages 917–941. DOI: 10.1137/140976546 (cited on pages 3, 8).

[Lub+13] Christian Lubich, Thorsten Rohwedder, Reinhold Schneider, and Bart Vandereycken. “Dynamical Approximation By Hierarchical Tucker And Tensor-Train Tensors”. In: *SIAM Journal on Matrix Analysis and Applications* 34 (Apr. 2013), pages 470–494. DOI: 10.1137/120885723 (cited on pages 2, 3, 8).

[Luo+14] Biao Luo, Huai-Ning Wu, Tingwen Huang, and Derong Liu. “Data-based approximate policy iteration for affine nonlinear continuous-time optimal control design”. In: *Automatica* 50.12 (2014), pages 3281–3290 (cited on page 3).

[McL64] A.D. McLachlan. “A variational solution of the time-dependent Schrodinger equation”. In: *Molecular Physics* 8.1 (1964), pages 39–44. DOI: 10.1080/00268976400100041. eprint: https://doi.org/10.1080/00268976400100041. URL: https://doi.org/10.1080/00268976400100041 (cited on page 3).

[Mur35] FD Murnaghan. “J. frenkel, wave mechanics; advanced general theory”. In: *Bulletin of the American Mathematical Society* 41.11 (1935), pages 776–776 (cited on pages 3, 7).

[NGK19] Tenavi Nakamura-Zimmerer, Qi Gong, and Wei Kang. *Adaptive Deep Learning for High-Dimensional Hamilton-Jacobi-Bellman Equations*. July 2019 (cited on page 3).

[NR21] Nikolas Nüsken and Lorenz Richter. “Solving high-dimensional Hamilton–Jacobi–Bellman PDEs using neural networks: perspectives from the theory of controlled diffusions and measures on path space”. In: *Partial Differential Equations and Applications* 2 (Aug. 2021). DOI: 10.1007/s42985-021-00102-x (cited on page 3).

[Ose11] Ivan Oseledets. “Tensor-Train Decomposition”. In: *SIAM J. Scientific Computing* 33 (Jan. 2011), pages 2295–2317. DOI: 10.1137/090752286 (cited on page 3).

[OT09] Ivan Oseledets and E. Tyrtyshnikov. “Breaking the Curse of Dimensionality, Or How to Use SVD in Many Dimensions”. In: *SIAM J. Sci. Comput.* 31 (Jan. 2009), pages 3744–3759. DOI: 10.1137/090748330 (cited on pages 3, 10).

[OSS21a] Mathias Oster, Leon Sallandt, and Reinhold Schneider. *Approximating optimal feedback controllers of finite horizon control problems using hierarchical tensor formats*. 2021. arXiv: 2104.06108 [math.OC] (cited on pages 2, 3, 5, 11, 13, 16).

[OSS21b] Mathias Oster, Leon Sallandt, and Reinhold Schneider. *Approximating the Stationary Bellman Equation by Hierarchical Tensor Products*. 2021. arXiv: 1911.00279 [math.OC] (cited on pages 13, 16).

[Pon87] Lev Semenovich Pontryagin. *Mathematical theory of optimal processes*. CRC press, 1987 (cited on page 3).

[RSN21] Lorenz Richter, Leon Sallandt, and Nikolas Nüsken. *Solving high-dimensional parabolic PDEs using the tensor train format*. Feb. 2021 (cited on page 3).

[Sal21] Leon Jasper Sallandt. "Computing High-Dimensional Value Functions of Optimal Feedback Control Problems using the Tensor-Train Format". Doctoral Thesis. Berlin: Technische Universität Berlin, 2021 (cited on page 2).

[SL79] George N. Saridis and C. S. George Lee. "An Approximation Theory of Optimal Control for Trainable Manipulators". In: *IEEE Transactions on Systems, Man, and Cybernetics* 9 (1979), pages 152–159 (cited on page 9).

[Ste16] Michael Maximilian Steinlechner. "Riemannian Optimization for Solving High-Dimensional Problems with Low-Rank Tensor Structure". In: (2016), page 165. DOI: 10.5075/epfl-thesis-6958. URL: http://infoscience.epfl.ch/record/217938 (cited on pages 2, 6, 7).

[TAK17] Daniela Tonon, Maria Aronna, and Dante Kalise. *Optimal Control: Novel Directions and Applications*. Jan. 2017. ISBN: 978-3-319-60770-2. DOI: 10.1007/978-3-319-60771-9 (cited on page 3).

[TSG21] Philipp Trunschke, Reinhold Schneider, and Michael Götte. "A block-sparse Tensor Train Format for sample-efficient high-dimensional Polynomial Regression". In: *Frontiers in Applied Mathematics and Statistics* (2021), page 57 (cited on page 3).

[ZH21] Mo Zhou and Jiequn Han. *Actor-Critic Method for High Dimensional Static Hamilton–Jacobi–Bellman Partial Differential Equations based on Neural Networks*. Feb. 2021 (cited on page 3).