TENSOR DECOMPOSITIONS FOR HIGH-DIMENSIONAL HAMILTON-JACOBI-BELLMAN EQUATIONS

SERGEY DOLGOV*, DANTE KALISE†, AND KARL KUNISCH‡

Abstract. A tensor decomposition approach for the solution of high-dimensional, fully nonlinear Hamilton-Jacobi-Bellman equations arising in optimal feedback control of nonlinear dynamics is presented. The method combines a tensor train approximation for the value function together with a Newton-like iterative method for the solution of the resulting nonlinear system. The tensor approximation leads to a polynomial scaling with respect to the dimension, partially circumventing the curse of dimensionality. A convergence analysis for the linear-quadratic case is presented. For nonlinear dynamics, the effectiveness of the high-dimensional control synthesis method is assessed in the optimal feedback stabilization of the Allen-Cahn and Fokker-Planck equations with a hundred of variables.

1. Introduction. Richard Bellman first coined the expression "curse of dimensionality" when referring to the overwhelming computational complexity associated to the solution of multi-stage decision processes through dynamic programming, what is nowadays known as Bellman's equation. More than 60 years down the road, the curse of dimensionality has become ubiquitous in different fields such as numerical analysis, compressed sensing and statistical machine learning. However, it is in the computation of optimal feedback policies for the control of dynamical systems where its meaning continues to be most evident. Here, the curse of dimensionality arises since the synthesis of optimal feedback laws by dynamic programming techniques demands the solution of a Hamilton-Jacobi-Bellman (HJB) fully nonlinear Partial Differential Equation (PDE) cast over the state space of the dynamics. This intrinsic relation between the dimensions of the state space of the control system and the domain of the HJB PDE generates computational challenges of formidable complexity even for relatively simple dynamical systems[1]. Much of the research in control revolves around circumventing this barrier through different trade-offs between dimensionality, performance, and optimality of the control design. Prominent examples of the research landscape shaped by the curse of dimensionality include model order reduction, model predictive control, suboptimal feedback design, reinforcement learning and distributed control. However, the effective computational solution of dynamic programming equations of arbitrarily high dimensions through deterministic methods remains an open quest with fundamental implications in optimal control design. In this paper, we present a computational approach based on tensor decomposition techniques for the solution of high-dimensional HJB PDEs arising in optimal feedback control of systems governed by partial differential equations. We show that for evolution equations arising from the semi-discretization of PDEs, our technique

*S. Dolgov is with the Department of Mathematical Sciences, University of Bath, North Rd, BA2 7AY Bath, United Kingdom (S.Dolgov@bath.ac.uk).

†D. Kalise is with the School of Mathematical Sciences, University of Nottingham, University Park Campus, NG7 2QL Nottingham, United Kingdom (dante.kalise@nottingham.ac.uk).

‡K. Kunisch is with Institute of Mathematics and Scientific Computing, University of Graz, Heinrichstr. 36, A-8010 Graz, Austria and Radon Institute for Computational and Applied Mathematics (RICAM), Altenbergerstraße 69, A-4040 Linz, Austria(karl.kunisch@uni-graz.at).

[1]As an illustration, consider the simplest double integrator dynamics $\ddot{x} = u$, whose optimal feedback synthesis already requires the solution of a two-dimensional PDE. For a quadrocopter model where the dynamics are described by a 12-dimensional nonlinear dynamical system, the associated HJB PDE has to be solved in $\mathbb{R}^{12}$. Bear in mind that much of the research in computational PDEs is devoted to the solution of problems in physical space, that is $\mathbb{R}^{3+1}$, at most.

scales at a rate which is at most polynomially with the dimension. This scaling allows the computation of accurate feedback laws for nonlinear dynamics with over 100 dimensions. We assess our design over a class of challenging problems, including the optimal feedback stabilization of nonlinear parabolic PDEs such as the Allen-Cahn and Fokker-Planck equations in one and two-spatial dimensions, and in the presence of control constraints.

1.1. The Numerical Approximation of HJB PDEs. Since the seminal work by Crandall and Lions [18], the approximation of HJB equations through computational PDE methods has been addressed by a range of discretization strategies, most notably finite differences, level-set methods and semi-Lagrangian schemes [25]. The aforementioned techniques have proven to overcome the difficulties associated to the fully nonlinear character of the HJB PDEs [48]. However, they are inherently grid-based schemes suffering from the curse of dimensionality. That is, for a multidimensional ansatz defined from a tensor product of 1d objects, the scaling of the total number of degrees of freedom of the discretization grows exponentially with respect to the dimension of the HJB PDE. In practice, this makes the problem computationally intractable for dimensions larger than 4. This is a fundamental limitation in the context of nonlinear optimal feedback control, where the dimension of the associated HJB PDE is determined by the dimension of the state space of the control system. A partial remedy to this difficulty is the coupling of grid-based discretizations for low-dimensional HJB PDEs with model reduction techniques to lower the dimension of the control system [43, 3]. This approach has been successfully applied to dynamics with strong dissipative properties, but its overall performance relies on a good state space sampling and deteriorates for dynamics including transport, delays, or highly nonlinear phenomena [36]. While the rigorous design of numerical methods for the solution of very high-dimensional HJB PDEs remains largely an open problem, encouraging results have been obtained over the last years. A non-exhaustive list includes the use of machine learning techniques [57, 23, 31, 35], approximate dynamic programming in the context of reinforcement learning [11, 54], causality-free methods and convex optimization [39, 17], max-plus algebra methods [46, 6], polynomial approximation [37, 38], tree structure algorithms [4], and sparse grids [16, 26]. A very recent stream of works [23, 57, 35, 53] has explored the use of machine learning techniques to approximate high-dimensional nonlinear PDEs. The work [57] proposes the so-called Deep Galerkin Method, combining a deep neural network ansatz for the solution together with a PDE residual minimization. In [23, 31, 35], the authors focus on the class of time-dependent HJB equations arising in stochastic control where a pointwise evaluation of the solution can be realized through a representation formula involving the solution of a backward stochastic differential equation. These latter approaches can be linked to causality-free methods, with deterministic counterparts explored in [39, 47, 17, 63].

1.2. A tensor calculus framework for nonlinear HJB. In this work we propose a numerical method for the solution of HJB equations based on tensor decomposition techniques [9, 42, 41], which have proven to be successful in tackling the curse of dimensionality in the context of numerical analysis of PDEs [40, 20]. The use of low-rank structures such as the tensor-train (TT) format [50] to represent high-dimensional objects allows the solution of linear high-dimensional problems by generalizing standard numerical linear algebra techniques to multi-index arrays of coefficients (tensors) and the multivariate functions they approximate. This approach has been recently explored in [34] for solving a class of finite-horizon stochastic control

problems where the associated time-dependent HJB PDE can be transformed into a linear equation [59]. A fixed-point iteration algorithm using tensor approximations with a Markov chain discretization was proposed in [28, 5]. A model-free learning of a TT representation of the value function from Monte Carlo realisations was proposed in [64]. Here, we extend the tensor calculus framework to approximate the solution of fully nonlinear, first-order, stationary HJB PDEs arising from deterministic infinite horizon control, using a fast Newton-type policy iteration and a spectral discretization. Furthermore, through the selection of suitable control penalties [44], our method allows the inclusion of control constraints in the design.

1.3. Contributions of this work. We develop a computational approach based on tensor decompositions for the solution of Hamilton-Jacobi-Bellman PDEs arising in the computation of optimal actions in feedback form. Our method scales at most polynomially with the state space dimension for a wide class of systems governed by high-dimensional nonlinear evolution equations, including dynamics originating from the Allen-Cahn and Fokker-Planck equations. The mitigation of the curse of dimensionality allows the accurate synthesis of optimal feedback maps for nonlinear dynamics with over 100 dimensions at a moderate computational cost. The method can effectively incorporate control constraints in the design.

The rest of the paper is structured as follows. In Section 2, we formulate the design problem of computing optimal feedback controllers for nonlinear dynamics via the HJB equation, together with an abstract iterative method for its approximation. In Section 3 we develop all the building blocks underpinning a tensor decomposition framework for the solution of high-dimensional HJB equations. Finally, in Section 4 we apply the proposed methodology to the computation of optimal feedback laws for the Allen-Cahn and Fokker-Planck equations.

2. The HJB PDE in optimal feedback control. We study the following infinite horizon optimal control problem:

$$\min_{u(\cdot)\in\mathcal{U}} \mathcal{J}(u(\cdot),\mathbf{x}) := \int_0^\infty \ell(\mathbf{y}(t)) + \gamma|u(t)|^2\,dt\,, \tag{2.1}$$

subject to the nonlinear dynamical constraint

$$\dot{\mathbf{y}}(t) = \mathbf{f}(\mathbf{y}(t)) + \mathbf{g}(\mathbf{y})u(t)\,, \quad \mathbf{y}(0) = \mathbf{x}, \tag{2.2}$$

where the state $\mathbf{y}(t) = (y_1(t),\dots,y_d(t))^\top \in \mathbb{R}^d$, the control $u \in \mathcal{U} \equiv L^\infty([0;+\infty[;U)$, with U a compact set of $\mathbb{R}$, the running cost $\ell(\mathbf{y}) : \mathbb{R}^d \to \mathbb{R}_0^+$, and the control penalization $\gamma > 0$. We assume the state cost $\ell(\mathbf{y})$ and the system dynamics $\mathbf{f}(\mathbf{y}) : \mathbb{R}^d \to \mathbb{R}^d$ and $\mathbf{g}(\mathbf{y}) : \mathbb{R}^d \to \mathbb{R}^d$ to be $\mathcal{C}^1(\mathbb{R}^d)$. Without loss of generality, the origin $\mathbf{y} = \mathbf{0}$ is an equilibrium of the uncontrolled dynamics and $\mathbf{g}(\mathbf{0}) = \ell(\mathbf{0}) = 0$. The control problem (2.1) corresponds to the design of a globally asymptotically stabilizing control signal $u(t)$, which can be solved by dynamic programming techniques. Defining the value function

$$V(\mathbf{x}) := \inf_{u(\cdot)\in\mathcal{U}} J(u(\cdot),\mathbf{x})\,, \tag{2.3}$$

we characterize the solution of the infinite horizon control problem as the unique viscosity solution of the HJB PDE

$$\min_{u\in U}\{(\mathbf{f}(\mathbf{x}) + \mathbf{g}(\mathbf{x})u)^\top DV(\mathbf{x}) + \ell(\mathbf{x}) + \gamma|u|^2\} = 0\,, \tag{2.4}$$

where $DV(\mathbf{x}) = (\partial_{x_1} V, \ldots, \partial_{x_d} V)^\top$. In the unconstrained case $U \equiv \mathbb{R}$, the minimizer u^* is expressed in a feedback form

$$u^*(\mathbf{x}) = -\frac{1}{2\gamma}\mathbf{g}(\mathbf{x})^\top DV(\mathbf{x})\,, \tag{2.5}$$

which after inserting in (2.4) leads to the HJB equation

$$DV(\mathbf{x})^\top \mathbf{f}(\mathbf{x}) - \frac{1}{4\gamma} DV(\mathbf{x})^\top \mathbf{g}(\mathbf{x})\mathbf{g}(\mathbf{x})^\top DV(\mathbf{x}) + \ell(\mathbf{x}) = 0\,. \tag{2.6}$$

This derivation can be extended to controls in $\mathbb{R}^m$ with $m > 1$. Moreover, it can be modified to enforce box constraints in the control action [44], by replacing the control penalty $\gamma|u|^2$ by

$$W(u) = 2\gamma \int_0^u \mathcal{P}^{-1}(\mu) d\mu\,, \tag{2.7}$$

where $\mathcal{P} : \mathbb{R} \to \mathbb{R}$ is an odd, bounded, integrable, bijective $\mathcal{C}^1$ function. The optimal feedback is given by

$$u^*(\mathbf{x}) = -\mathcal{P}\left(\frac{1}{2\gamma}\mathbf{g}(\mathbf{x})^\top DV(\mathbf{x})\right)\,, \tag{2.8}$$

where we can impose lower and upper bound constraints by choosing penalties of the type $\mathcal{P}(x) = u_{\max} \cdot \tanh(x/u_{\max})$.

2.1. An iterative approach for solving nonlinear HJB PDEs. The construction of a numerical scheme for (2.4) begins by dealing with the quadratic nonlinearity in the gradient. We apply the Continuous Policy Iteration developed in [7], a variant of the well-known policy iteration algorithm in dynamic programming [8, 52, 2]. Conceptually speaking, given an initial guess $u_0(\mathbf{x})$ for the optimal feedback control, we insert it into (2.4) which then becomes a linear PDE for $V(\mathbf{x})$, whose solution dictates the update of the feedback control via (2.5). Algorithm 1 summarizes the main steps of the procedure. The algorithm is equivalent to the application of a Newton-type method for $V(\mathbf{x})$ directly over (2.6). To guarantee the convergence of the policy iteration when solving (2.4) over a subdomain $\Omega \subset \mathbb{R}^d$, we require an initial feedback map $u_0(\mathbf{x})$ such that $\mathcal{J}(u_0(\mathbf{x}), \mathbf{x}) < \infty$ over Ω.

Algorithm 1 Continuous Policy Iteration Algorithm

Require: Admissible feedback $u_0(\mathbf{x})$, stopping tolerance $\delta > 0$

1: **while** $error > \delta$ **do**
2: **Solve the linearized HJB:** $(\mathbf{f}(\mathbf{x}) + \mathbf{g}(\mathbf{x})u_s)^T DV_s(\mathbf{x}) + \ell(\mathbf{x}) + \gamma|u_s|^2 = 0$.
3: **Feedback update:** $u_{s+1}(\mathbf{x}) = -\mathcal{P}\left(\frac{1}{2\gamma}\mathbf{g}^\top DV_s(\mathbf{x})\right)$.
4: **Set** $error = \|V_s - V_{s-1}\|$, $s := s + 1$.
5: **end while**
6: **return** $(V_s, u_s) \approx (V^*, u^*)$

3. A tensor decomposition framework for high-dimensional HJB equations. We employ the Galerkin spectral element approximation similarly to [37] except that we construct the basis functions from the Legendre polynomials of bounded

maximal *individual* degree $n-1$,

$$\Phi_{\mathbf{i}}(\mathbf{x}) := \phi_{i_1}(x_1)\cdots\phi_{i_d}(x_d), \quad i_k = 0,\dots,n-1, \tag{3.1}$$

where $\phi_{i_k}(x_k)$ are the univariate Legendre polynomials of degree $i_k \le n-1$, and the multi-index $\mathbf{i} = (i_1,\dots,i_d)$. In the sth step of Alg. 1, we seek the value function in the form

$$V_s(x_1,\dots,x_d) \approx \sum_{j_1,\dots,j_d=0}^{n-1} \mathbf{v}(j_1,\dots,j_d)\Phi_{j_1,\dots,j_d}(\mathbf{x}), \tag{3.2}$$

by making the Galerkin residual of the linearized HJB orthogonal to $\{\Phi_{\mathbf{i}}\}$. This requires solving a system of n^d Galerkin equations in n^d unknowns of $\mathbf{v}$,

$$\sum_{\mathbf{j}} \underbrace{\left\langle \Phi_{\mathbf{i}}, (\mathbf{f}+\mathbf{g}u_s)^\top D\Phi_{\mathbf{j}} \right\rangle}_{A(\mathbf{i},\mathbf{j})} \mathbf{v}(\mathbf{j}) = \underbrace{-\left\langle \Phi_{\mathbf{i}},\ \ell+\gamma u_s^2 \right\rangle}_{\mathbf{b}(\mathbf{i})}, \tag{3.3}$$

where $\langle\cdot,\cdot\rangle$ is an inner product in $L^2\left([-a,a]^d\right)$ with an appropriately chosen *domain size* $a>0$. Given the tensor product structure of (3.2), its accuracy can be analyzed with univariate polynomial approximation theory [60], with an exponential error decay rate $\mathcal{O}(n^{-p})$ for $V(\mathbf{x}) \in \mathcal{C}^p(\Omega)$.

3.1. Compressed Tensor Train representation. The coefficients in (3.2) are enumerated by d independent indices, so $\mathbf{v}$ can be treated as a d-dimensional *tensor*. Throughout the paper, we approximate such tensors by the so-called Tensor Train (TT) decomposition [50],

$$\tilde{\mathbf{v}}(\mathbf{i}) := \sum_{\alpha_0,\dots,\alpha_d=1}^{r_0,\dots,r_d} \mathbf{v}^{(1)}_{\alpha_0,\alpha_1}(i_1)\mathbf{v}^{(2)}_{\alpha_1,\alpha_2}(i_2)\cdots\mathbf{v}^{(d)}_{\alpha_{d-1},\alpha_d}(i_d). \tag{3.4}$$

The smaller (3-dimensional) tensors $\mathbf{v}^{(k)}$ on the right hand side are called *TT blocks*, and the new summation ranges $r_0,\dots,r_d$ are called *TT ranks*. For convenience we can introduce the maximal TT rank $r := \max_{k=0,\dots,d} r_k$. Counting the number of unknowns in the TT blocks in (3.4), one can conclude that the TT decomposition needs at most dnr^2 unknowns. For numerical efficiency, we assume that r can be taken much smaller than the original cardinality n^d for a desired approximation accuracy.

For theoretical analysis, it is convenient to combine (3.4) with (3.2) and to consider the *functional* TT format [49],

$$V(\mathbf{x}) \approx \widetilde{V}(\mathbf{x}) := \sum_{\alpha_0,\dots,\alpha_d=1}^{r_0,\dots,r_d} v^{(1)}_{\alpha_0,\alpha_1}(x_1)\cdots v^{(d)}_{\alpha_{d-1},\alpha_d}(x_d),$$

to work directly over the TT ranks of a function. Sharp rank bounds are usually hard to derive though: the SVD might reveal an optimal decomposition that is difficult to express analytically. It was proven in special cases [61, 56, 29] and extensively tested numerically that smooth (e.g. analytic) functions exhibit a logarithmic growth of TT ranks, $r \sim \log^p \varepsilon$, to achieve an error ε. As a rationale for using the TT format for the HJB equations, here we show that linear-quadratic value functions admit TT approximations with the similar convergence rate.

THEOREM 3.1. *Assume that $\ell(\mathbf{y}) = \frac{1}{2}\mathbf{y}^T Q\mathbf{y}$, where Q is a symmetric positive definite matrix with the Cholesky decomposition $Q = D^\top D$, and that the linear system*

$\dot{\mathbf{y}}(t) = A\mathbf{y}(t) + Bu$ *is stabilisable. There exists a solution* $\Pi \in \mathbb{R}^{d\times d}$ *of the Riccati equation*

$$A^\top \Pi + \Pi A - \frac{1}{\gamma}\Pi B B^\top \Pi + Q = 0,$$

such that the eigenvalues of $A_\pi = AD^{-1} - \frac{1}{\gamma}BB^\top \Pi D^{-1}$ *satisfy* $\lambda(A_\pi) \in [\lambda_{\min}, \lambda_{\max}] \oplus \mathrm{i}[-\mu, \mu]$, $\lambda_{\max} < 0$. *Assume that the ranks of the off-diagonal blocks of* $A_D = AD^{-1}$ *are bounded by a constant,* $\operatorname{rank} A_D(k+1:d,\ 1:k) \le M$ *for all* $k = 1, \dots, d-1$, *and that* $\operatorname{rank}(B) \le r_b$. *Then for any* $\varepsilon \in (0,1)$ *the value function* $V(\mathbf{x}) = \mathbf{x}^\top \Pi \mathbf{x}$ *admits a TT approximation* $\widetilde{V}(\mathbf{x})$ *with the TT ranks*

$$r_k \le \min\left((M + r_b)\left(\log\frac{1}{\varepsilon} + C\right)^{7/2},\ \ \min(k, d-k)\right) + 2,$$

and the error $\max_{\mathbf{x}\in[-a,a]^d} |V(\mathbf{x}) - \widetilde{V}(\mathbf{x})| \le \varepsilon$ *for some offset*

$$C = C_0 + C_1 \frac{\mu}{|\lambda_{\max}|} + C_2 \log\left[\frac{\lambda_{\min}}{\lambda_{\max}} \frac{\|A_\pi\| \|D^{-\top}\Pi B\|}{\gamma}\right] > 0\,,$$

where C_0, C_1, C_2 *are independent of* $d, \varepsilon, M, r_b, \gamma, \mu, \lambda_{\min}, \lambda_{\max}$. *If the second bound* $r_k = \min(k, d-k) + 2$ *is attained for all* k, *the TT decomposition* $\widetilde{V}$ *is exact.*

Proof. For the upper bound we employ [21, Thm 4.2]: for the second order polynomial $V(\mathbf{x}) = \mathbf{x}^\top \Pi \mathbf{x}$ with a symmetric matrix Π there exists an exact TT decomposition with the TT ranks governed by the ranks of the off-diagonal blocks of Π,

$$r_k \le \operatorname{rank}\left(\Pi(k+1:d,\ 1:k)\right) + 2. \tag{3.5}$$

This gives an obvious bound $\min(k, d-k) + 2$.

However, there might exist an approximate TT decomposition of lower ranks. First, we notice that the Riccati equation can be rewritten as a Lyapunov equation. In fact, using a stabilised matrix [14] $A - \frac{1}{\gamma}BB^\top \Pi$, and also the Cholesky factor of Q, we obtain

$$A_\pi^\top \Pi + \Pi A_\pi = -\frac{1}{\gamma} D^{-\top} \Pi B B^\top \Pi D^{-1} - I, \tag{3.6}$$

where I is a $d \times d$ identity matrix. The left hand side is constructed from the stable matrix A_π. Using the Kronecker product, we can write the Lyapunov equation as a large linear system,

$$\underbrace{(A_\pi \otimes I + I \otimes A_\pi)}_{\mathcal{A}} \operatorname{vec}(\Pi) = -\frac{1}{\gamma} \sum_{i=1}^{\operatorname{rank}(B)} (D^{-1}\Pi B_i) \otimes (D^{-1}\Pi B_i) - \operatorname{vec}(I),$$

where $\operatorname{vec}(\cdot)$ stacks all columns of a matrix into a vector. Now we can use [29, Thm. 9]: there exists an approximate inverse $\widetilde{\mathcal{A}}^{-1}$ of the Kronecker product form

$$\widetilde{\mathcal{A}}^{-1} = \sum_{j=-R}^{R} \frac{2w_j}{\lambda_{\max}} \exp\left(-\frac{2t_j}{\lambda_{\max}} A_\pi\right) \otimes \exp\left(-\frac{2t_j}{\lambda_{\max}} A_\pi\right)$$

with the approximation error

$$\|\mathcal{A}^{-1} - \widetilde{\mathcal{A}}^{-1}\| \le \tilde{C}\|\mathcal{A}\| \frac{\sqrt{\lambda_{\min}^2 + \mu^2}}{|\lambda_{\max}|} \exp\left(\frac{2}{\pi}\frac{\mu}{|\lambda_{max}|} - \pi\sqrt{2R}\right). \tag{3.7}$$

Multiplying the approximation $\widetilde{\mathcal{A}}^{-1}$ with the low-rank first term in the right hand side of (3.6), we obtain an incomplete solution of the form $\mathrm{vec}(\widehat{\Pi}) = \sum_{i=1}^{(2R+1)r_b} p_i \otimes q_i$, and hence the rank of $\widehat{\Pi}$ is bounded by $(2R+1)r_b$. The negative identity matrix in (3.6) yields the second term

$$\mathrm{vec}(\check{\Pi}) = \widetilde{\mathcal{A}}^{-1}\mathrm{vec}(-I), \quad \check{\Pi} = \sum_{j=-R}^{R} -\frac{2w_j}{\lambda_{\max}} \exp\left(-\frac{2t_j}{\lambda_{\max}}(A_\pi + A_\pi^\top)\right) \tag{3.8}$$

in the ultimate approximate solution $\widetilde{\Pi} = \widehat{\Pi} + \check{\Pi}$.

Since the first term $\widehat{\Pi}$ is a low-rank matrix, all its off-diagonal blocks (3.5) have low ranks of at most $(2R+1)r_b$ too. However, $\check{\Pi}$ is a full-rank matrix and we need to investigate its off-diagonal, also called *quasi-separable* [24], ranks directly. First, we recall [29, Lemma 16] that each matrix exponential in (3.8) can be approximated by a sum of $2k_e + 1$ resolvents with an error

$$\begin{aligned} &\left\| \exp\left(-\frac{2t_j}{\lambda_{\max}}(A_\pi + A_\pi^\top)\right) - \sum_{\ell=-k_e}^{k_e} \kappa_\ell \left(z_\ell I + \frac{2t_j}{\lambda_{\max}}(A_\pi + A_\pi^\top)\right)^{-1} \right\| \\ &\le \bar{C} \exp\left(4\left(\frac{4t_j\mu}{|\lambda_{max}|} + 1\right)^2 - \left(\frac{4t_j\mu}{|\lambda_{max}|} + 1\right)^{2/3} k_e^{2/3}\right). \end{aligned} \tag{3.9}$$

Since the quasi-separable rank of $A_{z_\ell} := z_\ell I + \frac{2t_j}{\lambda_{\max}}(A_\pi + A_\pi^\top)$ coincides with that of A_π, which is $M + r_b$, and on the other hand it coincides with the quasi-separable rank of the inverse matrix $A_{z_\ell}^{-1}$ [24, 30], we can conclude that the approximate quasi-separable rank of $\exp\left(-\frac{2t_j}{\lambda_{\max}}(A_\pi + A_\pi^\top)\right)$ is bounded by $(2k_e+1)(M+r_b)$, and the quasi-separable rank of $\check{\Pi}$ (3.8) is bounded by $(2R+1)(2k_e+1)(M+r_b)$.

The approximate value function is constructed as $\widetilde{V}(\mathbf{x}) = \mathbf{x}^\top \widetilde{\Pi} \mathbf{x}$, and by (3.5) we can estimate its TT rank as

$$r_k(\widetilde{V}) \le (2R+1)(r_b + (2k_e+1)(M+r_b)) + 2. \tag{3.10}$$

For the error estimate, we have

$$\varepsilon = \max_{\mathbf{x}\in[-a,a]^d} |V(\mathbf{x}) - \widetilde{V}(\mathbf{x})| \le a^2 \|\Pi - \widetilde{\Pi}\| \le a^2 \|\mathcal{A}^{-1} - \widetilde{\mathcal{A}}^{-1}\| \left(\frac{\|D^{-\top}\Pi B\|^2}{\gamma} + 1\right).$$

From (3.7) we obtain

$$\begin{aligned} R &\le \frac{1}{2\pi^2}\left(\frac{2}{\pi}\frac{\mu}{|\lambda_{\max}|} + \log\|A\| + \log\frac{\sqrt{\lambda_{\min}^2+\mu^2}}{|\lambda_{\max}|} + \hat{C} + \left|\log\|\mathcal{A}^{-1} - \widetilde{\mathcal{A}}^{-1}\|\right|\right)^2, \\ &\le \left(\log\frac{1}{\varepsilon} + \frac{\mu}{|\lambda_{\max}|} + \log\left(\frac{a^2\|D^{-\top}\Pi B\|^2\|A_\pi\|\sqrt{\lambda_{\min}^2+\mu^2}}{\gamma|\lambda_{\max}|}\right) + \hat{C}\right)^2 \end{aligned}$$

for some constant $\hat{C} > 0$, while (3.9) together with [29, Lemma 5] gives

$$k_e \leq \left(\left| \log \|\mathcal{A}^{-1} - \widetilde{\mathcal{A}}^{-1}\| \right| + \log R + \check{C} \right)^{3/2}$$

for $\check{C} > 0$ being some other constant. Plugging these bounds into (3.10), we obtain the first estimate of the TT rank.

□

REMARK 1. *In many cases one can take Q, and hence D, to be diagonal matrices, for example, if the 2-norm of the state vector (corresponding to the L^2-norm of the state function) is used in the cost functional. In this case the ranks of the off-diagonal blocks of A_D coincide with those of A.*

REMARK 2. *Although Thm. 3.1 is formulated only for linear systems, the remarkable proportionality between the TT ranks of the value function and the off-diagonal ranks of the linearised system matrix seems to hold more generally in practice. In particular, we observe from Fig. 4.1 that the TT ranks of the value function for discretised one-dimensional PDEs grow very mildly with the number of variables, which can be also attributed to the growth of the ratios $\lambda_{min}/\lambda_{max}$, $\mu/\lambda_{\max}$. However, when the system is produced from a two-dimensional PDE, the TT ranks grow proportionally to the number of degrees of freedom introduced in each spatial direction (see Fig. 4.6). Thus, the ranks of the actuator matrix and of the off-diagonal blocks of the Jacobian matrix can give a useful hint whether the TT approach might be efficient for the HJB equation of a particular dynamical system of interest.*

REMARK 3. *Alternatively, fast convergence of the TT approximation can be related to the smoothness of the original function [61, 56]. For example, it was verified [12] that the cost functional for the bilinear optimal control problem for the Fokker-Planck equation we present in our numerical results belongs to $C^\infty(\mathbb{R}^d)$.*

3.2. TT decomposition of the system functions and HJB equation. To take advantage of the TT decomposition of the value function, it is necessary to find also compatible representations of the stiffness matrix and right hand side of the Galerkin HJB equations (3.3). For example, matrices of size $n^d \times n^d$, such as that in the left hand side of (3.3), can be represented in a slightly different *matrix* TT format, where we separate pairs of row and column indices,

$$A(\mathbf{i}, \mathbf{j}) = \sum_{\beta_0, \ldots, \beta_d = 1}^{R_0, \ldots, R_d} A^{(1)}_{\beta_0, \beta_1}(i_1, j_1) \cdots A^{(d)}_{\beta_{d-1}, \beta_d}(i_d, j_d). \tag{3.11}$$

For complexity estimates we define also the upper bound $R \geq R_k$, $k = 0, \ldots, d$. In particular, we need to construct linear parts of the stiffness matrix,

$$A_{f_p}(\mathbf{i}, \mathbf{j}) := \langle \Phi_{\mathbf{i}}, f_p \partial_{x_p} \Phi_{\mathbf{j}} \rangle = \int_{[-a,a]^d} \Phi_{\mathbf{i}}(\mathbf{x}) f_p(\mathbf{x}) \partial_{x_p} \Phi_{\mathbf{j}}(\mathbf{x}) d\mathbf{x}, \tag{3.12}$$

where $f_p(\mathbf{x})$ is the p-th component of the drift $\mathbf{f}(\mathbf{x}) = (f_1(\mathbf{x}), \ldots, f_d(\mathbf{x}))$, $p = 1, \ldots, d$. The integral in (3.12) can be approximated by a tensorised Gauss-Legendre quadrature, e.g.

$$A_{f_p}(\mathbf{i}, \mathbf{j}) \approx \sum_{k_1, \ldots, k_d = 1}^{m} w_{k_1} \cdots w_{k_d} f_p(x_{k_1}, \ldots, x_{k_d}) \Phi_{\mathbf{i}}(x_{k_1}, \ldots, x_{k_d}) \partial_{x_p} \Phi_{\mathbf{j}}(x_{k_1}, \ldots, x_{k_d}),$$

where x_k, w_k, $k = 1, \dots, m$, are quadrature nodes on $(-a, a)$ and weights, respectively. Suppose that a TT decomposition of $f_p(x_{k_1}, \dots, x_{k_d})$ is given,

$$\mathbf{f}_p(k_1, \dots, k_d) := f_p(x_{k_1}, \dots, x_{k_d}) \approx \sum_{\beta_0, \dots, \beta_d = 1}^{R_0, \dots, R_d} \mathbf{f}^{(1)}_{p, \beta_0, \beta_1}(k_1) \cdots \mathbf{f}^{(d)}_{p, \beta_{d-1}, \beta_d}(k_d). \quad (3.13)$$

Plugging (3.13) into (3.12) and distributing the summations, we can compute directly the TT blocks of a TT decomposition of the stiffness matrix part,

$$A_{f_p}(\mathbf{i}, \mathbf{j}) = \sum_{\beta_0, \dots, \beta_d = 1}^{R_0, \dots, R_d} A^{(1)}_{p, \beta_0, \beta_1}(i_1, j_1) \cdots A^{(d)}_{p, \beta_{d-1}, \beta_d}(i_d, j_d),$$

where

$$A^{(p)}_{p, \beta_{p-1}, \beta_p}(i_p, j_p) = \left(\sum_{k_p = 1}^{m} w_{k_p} \phi_{i_p}(x_{k_p}) \mathbf{f}^{(p)}_{p, \beta_{p-1}, \beta_p}(k_p) \frac{d\phi_{j_p}(x_{k_p})}{dx} \right), \quad (3.14)$$

and

$$A^{(q)}_{p, \beta_{q-1}, \beta_q}(i_q, j_q) = \left(\sum_{k_q = 1}^{m} w_{k_q} \phi_{i_q}(x_{k_q}) \mathbf{f}^{(q)}_{p, \beta_{q-1}, \beta_q}(k_q) \phi_{j_q}(x_{k_q}) \right) \quad (3.15)$$

for $q \neq p$. Summing all different components A_{f_p}, one obtains the complete linear part $\langle \Phi_{\mathbf{i}}, \mathbf{f}^\top D \Phi_{\mathbf{j}} \rangle_{L^2(\Omega)}$. This summation can be performed in the TT format directly, followed by a rank truncation [50], or using the iterative Alternating Linear Scheme approximation (see Sec. 3.3 and [33]). In our numerical calculations we use the latter approach which requires $\mathcal{O}(d^2 n^2 R^2 r^2)$ floating point operations. Similarly we can compute the right hand side entries $\mathbf{b}(\mathbf{i}) = \langle -\Phi_{\mathbf{i}}(\mathbf{x}), \ell(\mathbf{x}) + \gamma u_s(\mathbf{x})^2 \rangle_{L^2(\Omega)}$, as well as nonlinear parts of the stiffness matrix, provided that tensors of nodal values $\mathbf{g}(x_{k_1}, \dots, x_{k_d})$ and $u_s(x_{k_1}, \dots, x_{k_d})$ are approximated by TT decompositions, similarly to (3.13).

The system functions are approximated in the TT format (3.13) using another iterative procedure, the so-called *TT-Cross* algorithm [51]. Any exact TT decomposition, e.g. (3.13), can be recovered from samples of the original tensor by an *interpolation formula* [55]

$$\mathbf{f}_p(\mathbf{i}) = \sum_{\substack{\beta_k, \beta'_k = 1 \\ k = 1, \dots, d-1}}^{R_k} \mathbf{f}_p(i_1, \mathcal{I}^{>1}_{\beta'_1}) \left(\mathbf{f}_p(\mathcal{I}^{\leq 1}, \mathcal{I}^{>1}) \right)^{-1}_{\beta'_1, \beta_1} \mathbf{f}_p(\mathcal{I}^{\leq 1}_{\beta_1}, i_2, \mathcal{I}^{>2}_{\beta'_2}) \cdots \mathbf{f}_p(\mathcal{I}^{\leq d-1}_{\beta_{d-1}}, i_d), \quad (3.16)$$

where $\mathcal{I}^{\leq k} = \{ i_1^{\beta_k}, \dots, i_k^{\beta_k} \}_{\beta_k = 1}^{R_k}$ and $\mathcal{I}^{>k} = \{ i_{k+1}^{\beta_k}, \dots, i_d^{\beta_k} \}_{\beta_k = 1}^{R_k}$ are *left*, respectively, *right* index sets chosen such that the intersection matrices $\mathbf{f}_p(\mathcal{I}^{\leq k}, \mathcal{I}^{>k})$ are invertible. For uniformity of notation, we let $\mathcal{I}^{\leq 0} = \mathcal{I}^{>d} = \emptyset$. Note that the inverse intersection matrices can be multiplied with the adjacent three-dimensional factors to obtain the TT blocks of (3.13), e.g.

$$\mathbf{f}^{(k)}_{p, \beta_{k-1}, \beta_k}(i_k) = \sum_{\beta'_k} \mathbf{f}_p(\mathcal{I}^{\leq k-1}_{\beta_{k-1}}, i_k, \mathcal{I}^{>k}_{\beta'_k}) \left(\mathbf{f}_p(\mathcal{I}^{\leq k}, \mathcal{I}^{>k}) \right)^{-1}_{\beta'_k, \beta_k}.$$

For numerical stability, the inversion is computed via the QR decomposition [51] or the incremental LU decomposition [55].

In practice, one seeks an approximate decomposition of the form (3.16). In this case it becomes important to find indices that not only give invertible intersection matrices, but deliver a small approximation error. The TT-Cross algorithm optimises the index positions iteratively. In the first step, assume that the right sets $\mathcal{I}^{>k}$ are given (for example at random). One can compute the first factor $F^{\{1\}}(i_1, \beta_1) = \mathbf{f}_p(i_1, \mathcal{I}^{>1}_{\beta_1})$, which can be seen as an $m \times R_1$ matrix. The smallest approximation error among sampling rank-R_1 approximations is given by such $i_1 \in \mathcal{I}^{\leq 1} \subset \{1, \dots, m\}$ that select the submatrix of maximum volume, i.e. $|\det F^{\{1\}}(\mathcal{I}^{\leq 1}, :)| = \max_{\#\mathcal{I}=R_1} |\det F^{\{1\}}(\mathcal{I}, :)|$. This set can be found by the *maxvol* algorithm [27] in $\mathcal{O}(mR_1^2)$ operations, similarly to the LU decomposition with pivoting.

Assume now that we have the left index set $\mathcal{I}^{\leq k-1}$ and the right set $\mathcal{I}^{>k}$. We can compute $R_{k-1}mR_k$ elements of the tensor and arrange them as a $R_{k-1}m \times R_k$ matrix with elements $F^{\{k\}}(\beta_{k-1}i_k,\ \beta_k) = \mathbf{f}_p(\mathcal{I}^{\leq k-1}_{\beta_{k-1}}, i_k, \mathcal{I}^{>k}_{\beta_k})$. Now we can apply the maxvol algorithm to $F^{\{k\}}$ to derive the next index set $\mathcal{I}^{\leq k}$ as a subset of the union of $\mathcal{I}^{\leq k-1}$ and i_k. This recursive procedure continues until the last TT block $\mathbf{f}_p(\mathcal{I}^{\leq d-1}, i_d)$ is computed. Moreover, we can reverse it in a similar fashion and carry out several TT-Cross iterations, as shown in Algorithm 2. This allows to optimise all index sets and, consequently, the approximations (3.16), (3.13) even if the initial guess was inaccurate.

Algorithm 2 TT-Cross iteration for the TT approximation (3.13)

Require: Initial sets $\mathcal{I}^{>k} \in \mathbb{N}^{r_k \times d-k}$, $k = 1, \dots, d-1$.

1: **for** $k = 1, 2, \dots, d$ **do**

2: Evaluate $r_{k-1}mr_k$ elements $F^{\{k\}}(\beta_{k-1}i_k;\ \beta_k) := \mathbf{f}_p\left(\mathcal{I}^{\leq k-1}_{\beta_{k-1}}, i_k, \mathcal{I}^{>k}_{\beta_k}\right)$.

3: Apply *maxvol* algorithm to $F^{\{k\}}$ to obtain $\mathcal{I}^{\leq k} \subset \mathcal{I}^{\leq k-1} \cup \{i_k\}$.

4: **end for**

5: **for** $k = d, d-1, \dots, 2$ **do**

6: Evaluate $r_{k-1}mr_k$ elements $F^{\{k\}}(\beta_{k-1};\ i_k\beta_k) := \mathbf{f}_p\left(\mathcal{I}^{\leq k-1}_{\beta_{k-1}}, i_k, \mathcal{I}^{>k}_{\beta_k}\right)$.

7: Apply *maxvol* algorithm to $(F^{\{k\}})^\top$ to obtain $\mathcal{I}^{>k-1} \subset \{i_k\} \cup \mathcal{I}^{>k}$.

8: **end for**

9: Assemble TT blocks $\mathbf{f}^{(k)}_{p,\beta_{k-1},\beta_k}(i_k) = \sum_{\beta'_k} \mathbf{f}_p(\mathcal{I}^{\leq k-1}_{\beta_{k-1}}, i_k, \mathcal{I}^{>k}_{\beta'_k}) \left(\mathbf{f}_p(\mathcal{I}^{\leq k}, \mathcal{I}^{>k})\right)^{-1}_{\beta'_k,\beta_k}$.

REMARK 4. *The result of TT-Cross might still depend on the heuristically chosen initial indices. Therefore, we distinguish the stopping threshold (called δ from now on) and the actual approximation error ε in the rest of the paper.*

Having computed TT approximations to all components f_p, we construct the matrix TT blocks (3.14)–(3.15). Similarly, we apply the TT-Cross algorithm to construct all components of $\mathbf{g}u_s$, assemble the corresponding parts of the stiffness matrix (3.3) in the TT format, and sum them together.

We can also precompute a TT matrix of the form (3.11) which maps the value function coefficients into the tensor of $u_{s+1}(\mathbf{x})$ values on the quadrature grid. In the unconstrained control case, we have that $u_{s+1} = -\frac{1}{2\gamma}\mathbf{g}^\top DV_s$, which maps the coefficients of the previous iterate of the value function $\mathbf{v}$ into a tensor of the corresponding

control values $\mathbf{u}(\mathbf{j}) := u_{s+1}(x_{j_1}, \dots, x_{j_d})$, and hence we assemble

$$\hat{B}(\mathbf{j}, \mathbf{i}) = \sum_{p=1}^{d} \sum_{\beta_0, \dots, \beta_d} -\frac{1}{2\gamma} \left(\mathbf{g}^{(1)}_{p,\beta_0,\beta_1}(j_1) \phi^{\delta(p,1)}_{i_1}(x_{j_1}) \right) \cdots \left(\mathbf{g}^{(d)}_{p,\beta_{d-1},\beta_d}(j_d) \phi^{\delta(p,d)}_{i_d}(x_{j_d}) \right), \tag{3.17}$$

using the TT approximations of $g_p(\mathbf{x})$, where

$$\phi_i^{\delta(p,q)} = \begin{cases} d\phi_i/dx, & p = q, \\ \phi_i, & \text{otherwise.} \end{cases}$$

Now the control signal $\mathbf{u} \approx \hat{B}\mathbf{v}$ can be constructed simply as a sum of products of a TT matrix (3.11) and a TT tensor (3.4), again with $\mathcal{O}(d^2 n^2 R^2 r^2)$ complexity [50]. In the constrained control case, the first step is the same, followed by approximating the pointwise constraint function

$$\mathbf{u}(j_1, \dots, j_d) = \tilde{u}_{\max} \tanh \left((\hat{B}\mathbf{v})(j_1, \dots, j_d) / \tilde{u}_{\max} \right) \tag{3.18}$$

in the TT format using again the TT-Cross method. Since the TT approximation may over- or undershoot the exact limits by the relative approximation error $\varepsilon \leq \delta$, we seek a slightly tighter bound $\tilde{u}_{\max} = (1 - \delta) u_{\max}$.

3.3. Iterative tensor algorithm for solving (3.3)**.** The policy update solves (3.3) at every iteration by taking the previous iterate of the value tensor $\check{\mathbf{v}}$, constructing the control signal, the stiffness matrix and the right hand side, and finally by solving the linear system on the new value tensor approximation. The latter step implies using only iterative methods that can preserve the TT structure of all data. One of the most robust techniques used nowadays is the Alternating Linear Scheme (ALS) [33] and its enhanced version, the Alternating Minimal Energy (AMEn) algorithm [22].

The ALS is a linear projection method similarly to the Krylov techniques, but in contrast to the latter it projects the equations onto bases constructed from the TT decomposition of the solution itself. Notice that the TT decomposition (3.4) is linear with respect the the elements of each particular TT block, e.g. $\mathbf{v}^{(k)}$. Given (3.4), let us define a partial TT decomposition where $\mathbf{v}^{(k)}$ is replaced by the identity matrix,

$$\begin{aligned} V_{\neq k}(i_1, \dots, i_d;\ \alpha_{k-1}, j_k, \alpha_k) = \sum_{\substack{\alpha_0, \dots, \alpha_{k-2}, \\ \alpha_{k+1}, \dots, \alpha_d}} & \mathbf{v}^{(1)}_{\alpha_0,\alpha_1}(i_1) \cdots \mathbf{v}^{(k-1)}_{\alpha_{k-2},\alpha_{k-1}}(i_{k-1}) \\ & \cdot \delta(i_k, j_k) \\ & \cdot \mathbf{v}^{(k+1)}_{\alpha_k,\alpha_{k+1}}(i_{k+1}) \cdots \mathbf{v}^{(d)}_{\alpha_{d-1},\alpha_d}(i_d). \end{aligned} \tag{3.19}$$

Clearly, the original TT decomposition (3.4) can be produced from $V_{\neq k}$ by just multiplying it with the k-th TT block. Specifically, we introduce the vector form

$$\bar{v}^{(k)}(\alpha_{k-1}, i_k, \alpha_k) = \mathbf{v}^{(k)}_{\alpha_{k-1},\alpha_k}(i_k), \quad \bar{v}^{(k)} \in \mathbb{R}^{r_{k-1} n_k r_k}, \tag{3.20}$$

and treat $V_{\neq k}$ as a $n^d \times (r_{k-1} n_k r_k)$ matrix. One can check that

$$\tilde{\mathbf{v}} = V_{\neq k} \bar{v}^{(k)}$$

holds for any $k = 1, \dots, d$.

Now, assuming that the entire stiffness matrix and the right hand side in (3.3) are assembled in TT formats (3.11) and (3.4), the ALS method iterates over $k = 1, \dots, d$ (hence the name alternating), seeking for one TT block at a time by making the residual orthogonal to $V_{\neq k}$,

$$\left(V_{\neq k}^\top A V_{\neq k}\right) \bar{v}^{(k)} = \left(V_{\neq k}^\top \mathbf{b}\right). \tag{3.21}$$

Notice that the reduced system (3.21) is of size $r_{k-1} n r_k$, i.e. much smaller than the original system (3.3). Once we solve (3.21), we populate the TT block $\mathbf{v}^{(k)}$ with the elements of $\bar{v}^{(k)}$ through (3.20), and use the updated $\mathbf{v}^{(k)}$ to construct (3.19) in the next step $k \to k+1$ or $k \to k-1$. Efficient practical implementation employs the fact that (3.19) mimics the original TT decomposition (3.4) in the sense that same original indices $i_1, \dots, i_d$ are separated. Therefore, given (3.11), the reduced matrix $V_{\neq k}^\top A V_{\neq k}$ can be computed block by block in a total of $\mathcal{O}(dn^2 r^4)$ operations [33]. Here we assume also that $R = \mathcal{O}(r)$, which is the case for the quadratic HJB equation. Since the reduced matrix in (3.21) inherits the low-rank structure of the matrix TT decomposition (3.11), we can solve (3.21) iteratively using a fast matrix-vector product with the same cost of $\mathcal{O}(dn^2 r^4)$.

In the AMEn method [22], the TT blocks are also enriched with auxiliary vectors, such as approximate residuals, which gives a mechanism for increasing TT ranks and adapting them to the desired accuracy. Consider an auxiliary TT decomposition approximating the residual, projected onto the first $k-1$ TT blocks of the solution,

$$\begin{aligned}(\mathbf{b} - A\mathbf{v})(i_1, \dots, i_d) \approx \sum_{\alpha_0, \dots, \beta_d = 1}^{r_0, \dots, \rho_d} & \mathbf{v}^{(1)}_{\beta_0, \beta_1}(i_1) \cdots \mathbf{v}^{(k-1)}_{\alpha_{k-2}, \alpha_{k-1}}(i_{k-1}) \\ & \cdot \mathbf{z}^{(k)}_{\alpha_{k-1}, \beta_k}(i_k) \cdot \mathbf{z}^{(k+1)}_{\beta_k, \beta_{k+1}}(i_{k+1}) \cdots \mathbf{z}^{(d)}_{\beta_{d-1}, \beta_d}(i_d).\end{aligned}$$

After solving (3.21) and updating (3.20), we expand $\mathbf{v}^{(k)}$ and $\mathbf{v}^{(k+1)}$ increasing their ranks by ρ_k,

$$\mathbf{v}^{(k)}(i_k) := \begin{bmatrix}\mathbf{v}^{(k)}(i_k) & \mathbf{z}^{(k)}(i_k)\end{bmatrix}, \qquad \mathbf{v}^{(k+1)}(i_{k+1}) := \begin{bmatrix}\mathbf{v}^{(k+1)}(i_{k+1}) \\ 0\end{bmatrix}. \tag{3.22}$$

Note that this step does not perturb the whole solution tensor $\tilde{\mathbf{v}}$ due to the zero block in $\mathbf{v}^{(k+1)}$, but the subspace of columns of $V_{\neq k+1}$ in the next step is enriched due to the residual block $\mathbf{z}^{(k)}$. To reduce the TT rank, we can truncate $\mathbf{v}^{(k)}$ using the SVD.

Finally, we can *orthogonalise* TT blocks using QR decompositions [50] such that

$$\sum_{\alpha_{k-1} i_k} \mathbf{v}^{(k)}_{\alpha_{k-1}, \alpha_k}(i_k) \mathbf{v}^{(k)}_{\alpha_{k-1}, \beta_k}(i_k) = \delta(\alpha_k, \beta_k) \tag{3.23}$$

or

$$\sum_{i_k \alpha_k} \mathbf{v}^{(k)}_{\alpha_{k-1}, \alpha_k}(i_k) \mathbf{v}^{(k)}_{\beta_{k-1}, \alpha_k}(i_k) = \delta(\alpha_{k-1}, \beta_{k-1}). \tag{3.24}$$

This makes the projection matrix $V_{\neq k}$ orthogonal as well, which improves numerical stability of the algorithm.

If the matrix A was symmetric positive definite then the projected system (3.21) could be rigorously related to the energy optimization problem and the nonlinear

block Gauss–Seidel method. In our problem (3.3) this is not the case: A is non-symmetric and degenerate due to the gradient operator D, which annihilates any constant component in the solution. In this case, the degenerate reduced matrix in (3.21) can prevent convergence.

However, A is compatible with the right hand side under the condition $\ell(0) = \check{u}(0) = 0$. Moreover, the eigenvalues of A are located in the right half of the complex plane for a suitable choice of the domain size a and polynomial order n. In this case we can resolve both issues by solving shifted systems, mimicking the implicit Euler time propagation. We introduce a shift $\mu > 0$, and solve

$$(A + \mu I)\,\mathbf{v} = \mathbf{b} + \mu\check{\mathbf{v}}, \tag{3.25}$$

where $\check{\mathbf{v}}$ is the previous iterate of $\mathbf{v}$. In practice, we can even combine this shifted AMEn solver and the policy updates into a single iteration as shown in Alg. 3.

Algorithm 3 Policy update with shifted AMEn linear solver

Choose initial value tensor $\mathbf{v}$, shift $\mu > 0$, stopping threshold $\delta > 0$, previous iterate $\check{\mathbf{v}} = 0$, shift reduction factor $0 < q < 1$.
while $\|\mathbf{v} - \check{\mathbf{v}}\|_2 > \delta\|\mathbf{v}\|_2$ **do** ▷ Policy iteration
  Set $\check{\mathbf{v}} = \mathbf{v}$ and (optionally) $\mu := \mu q$.
  Compute the control $\mathbf{u}$ using (3.17) and (optionally) (3.18).
  Construct $A[\mathbf{u}]$ and $\mathbf{b}[\mathbf{u}]$ for (3.3) using Alg. 2 and (3.11)–(3.15).
  Orthogonalise TT blocks of $\mathbf{v}$ s.t. (3.24) holds.
  for $k = 1, \ldots, d$ **do** ▷ AMEn algorithm
    Assemble and solve $\left(V_{\neq k}^\top A V_{\neq k} + \mu I\right)\bar{v}^{(k)} = V_{\neq k}^\top \mathbf{b} + \mu V_{\neq k}^\top \check{\mathbf{v}}$ using (3.19).
    Update $\mathbf{v}^{(k)}$ via (3.20) and truncate r_k using SVD up to accuracy δ.
    Enrich $\mathbf{v}^{(k)}, \mathbf{v}^{(k+1)}$ using (3.22) and orthogonalise s.t. (3.23) holds.
  end for
end while

If Re $\lambda(A) \geq 0$, then the spectral radius of the transition matrix $\mu(A + \mu I)^{-1}$ is less than 1 for any $\mu > 0$. On the other hand, if $\mu > -\mathbf{v}^\top A\mathbf{v}$ for any $\mathbf{v} : \|\mathbf{v}\|_2 = 1$, then the reduced matrix $V_{\neq k}^\top A V_{\neq k} + \mu I$ (remember that $V_{\neq k}^\top V_{\neq k} = I$) is invertible. This gives a freedom to choose μ such that the method remains stable and converges fast enough. In practice we need to ensure $\mu > -\mathbf{v}^\top A\mathbf{v}$ only for those $\mathbf{v}$ that belong to span $V_{\neq k}$. It turns out that as the solution converges, we can decrease μ geometrically (in particular, we multiply it by a factor $q = 0.98$ in each iteration), which ensures faster convergence near the end of the process.

4. Optimal feedback control of nonlinear PDEs. High-dimensional nonlinear control systems naturally arise when the dynamics of the system are governed by partial differential equations. From a dynamical perspective, PDEs correspond to abstract, infinite-dimensional systems and therefore the HJB synthesis is understood over an infinite-dimensional state space. Computationally, the treatment is based on the so-called method of lines [62]. Given an evolutionary PDE, we discretize the space dependence either by finite differences/elements or spectral methods, leading to a large-scale dynamical system with as many state variables as the space discretization dictates. We perform the HJB synthesis over this finite but high-dimensional system. The accuracy of such a representation and its implications over the control design vary depending on the class of PDEs under consideration. Strongly dissipative PDEs

can be accurately represented with few degrees of freedom in space, while convection-dominated PDEs might require a much more complex state space representation. Therefore, the performance study of our framework with respect to the dimension parameter is central to our analysis. It benefits from the fact that, unlike the nonlinear ODE world, the taxonomy of physically meaningful nonlinearities in time-dependent PDEs is well-delimited. We focus on nonlinear reaction PDEs where we take the Allen-Cahn equation as a reference model due to its rich equilibrium structure, and to nonlinear convection in the Fokker-Planck equation, where the control action enters as a bilinear term. The semi-discretization in space of these nonlinearities leads to well-structured finite-dimensional realizations [37] which allow a systematic analysis of the scaling of our methodology with respect to the dimension. For implementation of the TT algorithms we used the TT-Toolbox[2] of Feb 9, 2019. All tests were carried out in Matlab 2017b on one core of an Intel Xeon E5-2640 v4 CPU at 2.4 GHz.

4.1. The Allen-Cahn equation. We consider the following nonlinear diffusion-reaction equation [37]:

$$\partial_t x(\xi,t) = \sigma \partial_{\xi\xi} x + x(1-x^2) + \chi_\omega(\xi) u(t)\,, \tag{4.1}$$

in $[-1,1] \times \mathbb{R}^+$ with Neumann boundary conditions. We set $\sigma = 0.2$, and the scalar control signal $u(t)$ acts through the indicator χ_ω of the subdomain $\omega = [-0.5, 0.2]$. This equation has $x = 0$ as unstable equilibrium and $x = \pm 1$ as stable equilibria. The cost is given by

$$\mathcal{J}(u,x) = \int_0^\infty \|x(\xi,t)\|^2_{L_2(-1,1)} + \gamma u(t)^2 dt\,. \tag{4.2}$$

where for concreteness we take $\gamma = 0.1$. Thus, the control objective consists in stabilizing the unstable equilibrium. (4.1) is discretized by Chebyshev pseudospectral collocation method [60] using d points $\xi_k = -\cos(\pi k/(d+1))$, $k = 1, \ldots, d$. The discrete state is collected into a vector of nodal values $X(t) = (X_1(t), \ldots, X_d(t))$ where $X_k(t) \approx x(\xi_k, t)$, leading to a d-dimensional nonlinear ODE

$$\frac{dX}{dt} = AX + X \odot (1 - X \odot X) + Bu(t), \tag{4.3}$$

where "$\odot$" is the coordinatewise Hadamard product, A is the pseudospectral differentiation matrix corresponding to the Laplace operator, and B is a vector corresponding to the pseudospectral discretisation of the indicator function $\chi_\omega(\xi)$. The HJB equation solver is applied directly to (4.3), restricting the domain of the value function to $(-3,3)^d$, sufficient to accommodate typical initial states. We compare our design against the linear-quadratic regulator LQR feedback law [58, Chapter 8] computed for the dynamics in (4.3) linearised around the origin, $A_L = A + I$.

Algorithmic performance for the one-dimensional Allen-Cahn equation. We first investigate the performance of Algorithm 3 with respect to the dimension of the dynamical system. We fix $n = 5$, $\delta = 10^{-3}$, and the initial shift in Alg. 3 $\mu = 50$. In Fig. 4.1 we can observe that the maximal TT rank grows linearly with the dimension. The $\mathcal{O}(dn^2r^4)$ complexity of the ALS method leads to a total cost bound in order of d^5. However, the effective cost is closer to $\mathcal{O}(d^4)$ (Fig. 4.1, right), which

[2]Available at `https://github.com/oseledets/TT-Toolbox`

can be attributed to a non-uniform distribution of TT ranks along the decomposition. This is a significant reduction compared to the exponential cost of the full Cartesian ansatz n^d. However, the method can become slow in very high dimensions, mainly due to the increase in the number of policy iterations as shown in Fig. 4.1, resulting from a larger condition number of the linearised system.

FIG. 4.1. *Allen-Cahn problem* (4.1). *Left: numbers of policy iterations and maximal TT ranks. Right: differences in total running cost and CPU times for different spatial dimensions d. Numbers above points in the right plot denote d. Other settings* $n = 5$ *and* $\delta = 10^{-3}$.

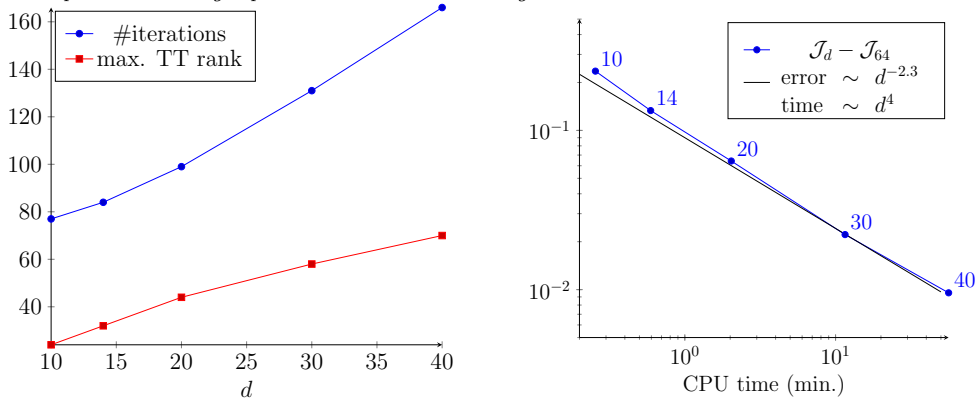
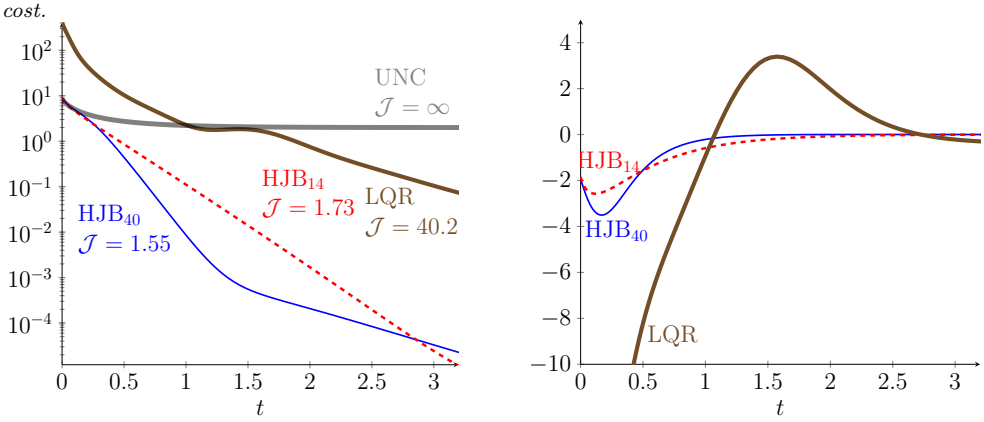


FIG. 4.2. *Allen-Cahn problem* (4.1) *with* $d = 40$, $n = 5$ *and* $\delta = 10^{-3}$. *Left: time evolution of running costs. Right: control signals. The uncontrolled state converges to* $X = 1$ *with an infinite cost.*



A possible remedy is to construct the value function for a lower-dimensional discretisation of the PDE, and interpolate the state of a system with finer discretisation onto this lower-dimensional spatial grid. The resulting error, proportional to the discretisation error of the lower-dimensional grid, might be still much smaller than the error resulting from e.g. the linearisation of the system. Figure 4.2 (left) shows the running costs $\mathcal{J} = \|x(\xi, t)\|^2 + \gamma u(t)^2$ for the HJB control with system dimension 40 with $n = 5$ and $\delta = 10^{-3}$ (HJB_{40}), an interpolated HJB control from dimension 14 and the same $n = 5, \delta = 10^{-3}$ (HJB_{14}), an LQR feedback, and finally the cost of the UNControlled system (both with $d = 40$), with the initial condition

$x(\xi, 0) = 2 + \cos(2\pi\xi)\cos(\pi\xi)$. Fig. 4.2 (right) shows the corresponding control signals. Since the linearized system is unstable, the LQR acts very aggressively during the transient phase. The HJB synthesis is able to detect the stabilizing effect of the nonlinearity and produce a control at much lower cost. We observe differences in the control signals and total costs for the HJB laws depending on the dimension of the dynamical system, justifying the need for accurate high-dimensional HJB solvers.

FIG. 4.3. *Allen-Cahn problem* (4.1) *with* $\delta = 10^{-3}$ *and* $d = 14$. *Left: numbers of policy iterations and maximal TT ranks. Right: differences in total running cost and CPU times for different univariate polynomial degrees* n. *Numbers above points in the right plot denote* n.

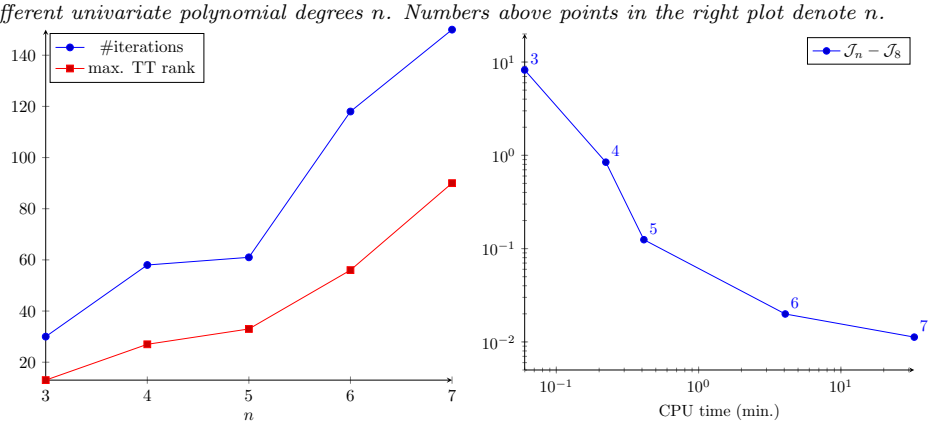
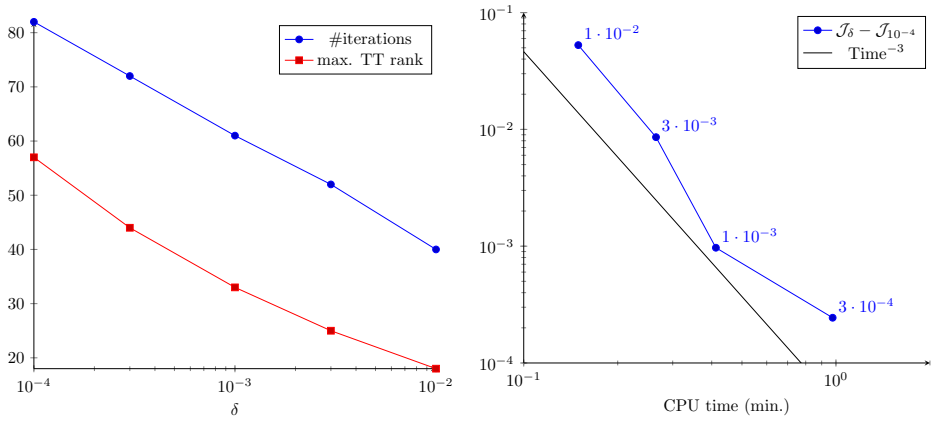


FIG. 4.4. *Allen-Cahn problem* (4.1) *with* $n = 5$ *and* $d = 14$. *Left: numbers of policy iterations and maximal TT ranks. Right: differences in total running cost and CPU times for different TT approximation thresholds* δ. *Numbers above points in the right plot denote* δ.

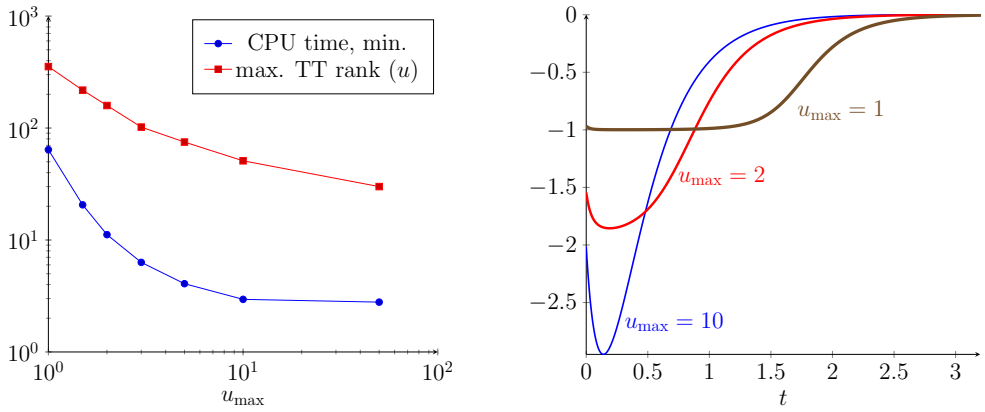


Now we analyse the performance of the TT-HJB scheme depending on the number of Legendre polynomials n in each variable (Fig. 4.3) and the stopping threshold δ in Alg. 3 (Fig. 4.4). Due to the nonlinearity in (4.3), the value function is significantly far from a quadratic polynomial, which is reflected in Fig. 4.3 by the linear growth of TT ranks with n, and a relatively slow algebraic decay of the error. Nevertheless, even an order-4 approximation can give a substantially better control signal than the LQR approximation, see Fig. 4.2. From Fig. 4.4 we see that the number of iterations and the TT ranks depend logarithmically on the TT approximation error, which is a more

optimistic result than that predicted by Thm. 3.1, although the problem is nonlinear. The errors in the total cost start higher than the threshold δ, but eventually the two error indicators are of the same order.

Allen-Cahn problem with control constraints. As discussed in Section 2, the proposed framework allows to enforce control constraints through a suitable choice of the control penalties in (2.8). Figure 4.5 (left) shows the total CPU times and TT ranks of the constrained feedback law. Figure 4.5 (right) presents the control signals for three bound parameters. As $\mathcal{P}(x)$ becomes steeper for more severe control constraints, the TT ranks increase leading to longer computing times. Nevertheless, Alg. 3 remains effective for a wide range of constraints, adjusting the value function accordingly.

FIG. 4.5. *Allen-Cahn problem with control constraints. Left: CPU times and TT ranks. Right: control signals for different control constraints* $-u_{\max} \leq u \leq u_{\max}$. *We set* $d = 20$, $\delta = 10^{-3}$ *and* $n = 5$.



Allen-Cahn problem with 2-dimensional space. We study the extension of the problem (4.1) to two spatial dimensions with the state depending on two space coordinates, $x(\boldsymbol{\xi}, t) = x(\xi_1, \xi_2, t)$. We replace the second derivative with the Laplace operator, and the control is applied on the domain $\omega = [-0.5, 0.2]^2$. We use the Cartesian product of the same Chebyshev grids in each direction, and similar homogeneous Neumann conditions on the boundary of $\Omega = [-1, 1]^2$. The CPU times and TT ranks are shown in Fig. 4.6 (left). Although Theorem 3.1 is not immediately applicable to a nonlinear system, we still observe a linear growth of the TT ranks with the number of Chebyshev points in each direction. The values of the ranks are larger than those in the one-dimensional case, leading to increased computing times. However, the performance of the high-dimensional HJB controller is satisfactory. Figures 4.6 (right) and 4.7 show the response of the system with an initial state $x(\boldsymbol{\xi}, 0) = 2 + \cos(2\pi\xi_1)\cos(\pi\xi_2)$. We can see again that the HJB-controlled state is stabilized while the LQR synthesis fails.

4.2. The Fokker-Planck equation. We compute optimal feedback regulators for the stabilised bilinearly controlled Fokker-Planck equation

$$\begin{aligned} \partial_t x(\xi, t) &= \nu \partial_{\xi\xi} x + \partial_\xi (x \partial_\xi G) + u \partial_\xi (x \partial_\xi H), \quad \xi \in \Omega, \\ 0 &= \left[\partial_\xi x + x \partial_\xi (G + uH)\right] |_{\xi \in \partial\Omega}, \end{aligned} \tag{4.4}$$

FIG. 4.6. *Two-dimensional Allen-Cahn control problem. Left: CPU times and TT ranks. Right: HJB control signals.*

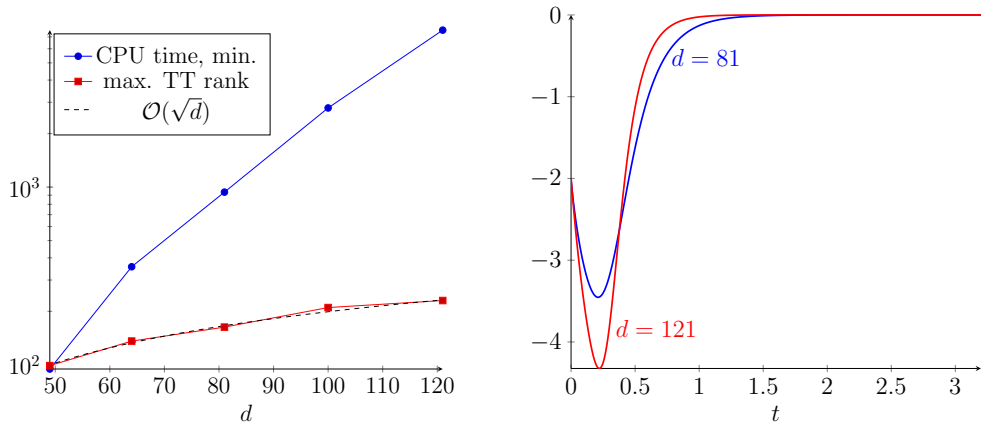
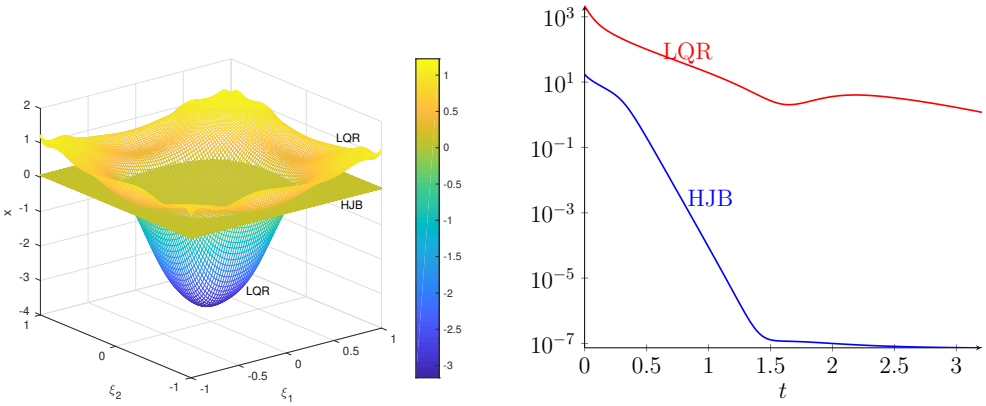


FIG. 4.7. *Two-dimensional Allen-Cahn control problem with* $d = 121$. *Left: state snapshots at* $t = 0.6$ *for HJB and LQR control laws. Right: total running costs.*



where the computational domain will be set $\Omega = (-6, 6)$. This equation models the density of particles, controlled with laser-induced electric force with potential $G(x) + u(t)H(x)$ [32], with G the ground and H the control potential. This system has 0 as an eigenvalue with associated eigenstate $x_\infty = \exp(-(\log \nu + \frac{G}{\nu}))$, see eg. [13]. Henceforth the eigenstates are considered as normalized in $L^2(\Omega)$. It is known that x_∞ is stable, but the convergence to this steady state, which is given by the second eigenvalue and depends on ν and G, can be extremely slow, see for instance [45, pg 251]. Thus, to speed up convergence in the transient phase, control is of importance. To obtain a suitable stabilization problem we introduce the shifted state $y = x - x_\infty$. It satisfies

$$\begin{aligned} \partial_t y(\xi, t) &= \nu \partial_{\xi\xi} y + \partial_\xi(y \partial_\xi G) + u \partial_\xi(y \partial_\xi H) + u \partial_\xi(x_\infty \partial_\xi H), \quad \xi \in \Omega, \\ 0 &= [\partial_\xi y + y \partial_\xi G + u(y + x_\infty)\partial_\xi H] \,|_{\xi \in \partial\Omega}. \end{aligned} \tag{4.5}$$

The control objective consists now in driving y to zero. To compute the controller we further introduce a positive, i.e. destabilising, shift by adding σy to the right hand side of (4.5). If this controller is applied to the unshifted equation it accelerates

convergence of y to 0 and hence the convergence of x to x_∞.

Considering the variational form of (4.5) one observes that the control will not have an effect on a subspace of co-dimension one. For this reason we introduce $Y_\mathcal{P} = \{v \in L^2(\Omega) : \int_\Omega v\, d\xi = 0\}$, and denote by $\mathcal{P} \in \mathcal{L}(L^2(\Omega), Y_\mathcal{P})$ the projection onto $Y_\mathcal{P}$ along x_∞, which is given by $\mathcal{P}y = y - (\int_\Omega y\, d\xi)x_\infty$. Subsequently we apply $\mathcal{P}$ to (4.5) with initial datum given by $\mathcal{P}x(0)$. For the details we refer to [12].

The Fokker-Planck equation (4.5) is discretized using a finite difference scheme with D intervals. To allow for possible further reduction of the dimension a balanced truncation based model reduction, adapted to bilinear systems [10], is used, to reduce the system to dimension d.

For the numerical results we fix $\gamma = 10^{-2}, \nu = 1$, $\sigma = 0.2$, and the potentials $G(\xi)$ and $H(\xi)$ are chosen to reproduce the setting in [13], as shown in Fig. 4.8. That is, the ground potential is set to be

$$G(\xi) = \frac{\left((0.5\xi^2 - 15)\xi^2 + 119\right)\xi^2 + 28\xi + 50}{200}, \tag{4.6}$$

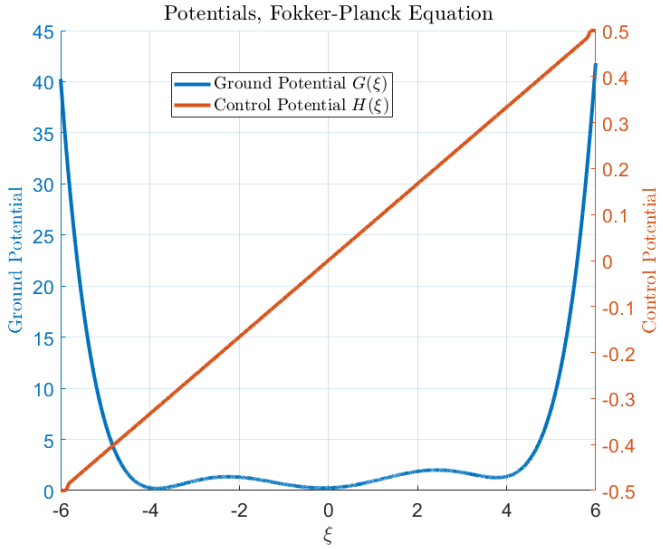
whereas $H(\xi)$ is given by

$$H(\xi) = \begin{cases} -1/2 & \text{if} \quad -6.0 \le \xi \le -5.9 \\ \xi/12 & \text{if} \quad -5.8 \le \xi \le 5.8 \\ 1/2 & \text{if} \quad 5.9 \le \xi \le 6.0 \end{cases} \tag{4.7}$$

with the disjoint intervals united with an Hermite interpolant.

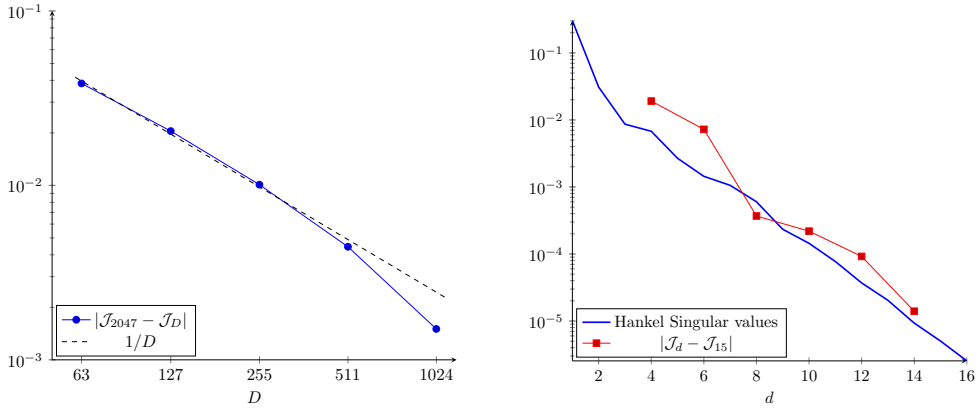
Since both the original system size D, and the reduced dimension d, are approximation parameters, we need to set them to appropriate values that deliver a desired accuracy in the model outcomes, such as the total cost. Note also that in contrast to the linear case, the generalized balanced truncation method for bilinear systems does not exhibit an a priori error bound [10].

FIG. 4.8. *Ground and control potentials in the Fokker-Planck control system.*



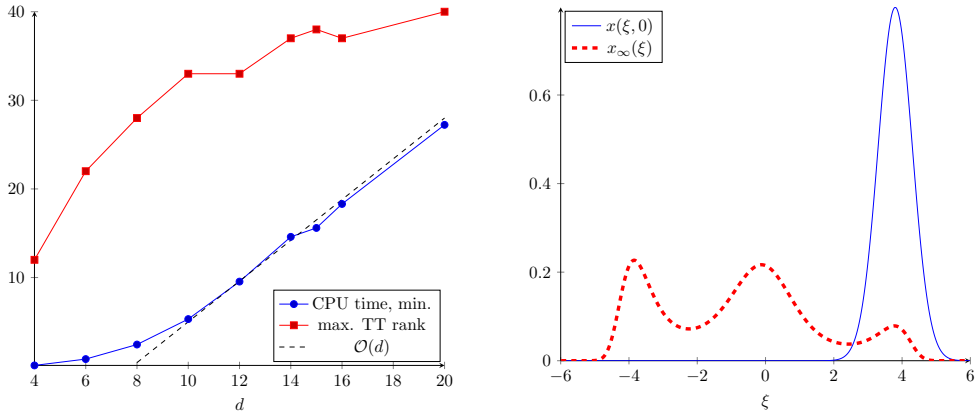
In Fig. 4.9 (left) we study the total cost in the LQR stabilised system. Here we initialise the Fokker-Planck system with the density function of the uniform distribution on $[-6, 6]$. We can deduce that an absolute error of about 10^{-3} (a relative error of 1%) is achieved for 1023 points in the initial finite difference discretization. Setting $D = 1023$ and varying the basis size in the balanced truncation, we compare the Hankel singular values and the differences in the total cost in the HJB stabilised system in Fig. 4.9 (right). We observe that $d = 10$ dimensions in the reduced model are sufficient to drop the absolute error below the same level of 10^{-3}.

FIG. 4.9. *Errors in the total cost in the Fokker-Planck model with the uniform initial state* $x(\xi, 0) = \frac{1}{12}$ *for different numbers of discretisation points (left) and different dimensions of the reduced model (right).*



In Fig. 4.10 (left) we vary the dimension d of the reduced state and investigate CPU times and TT ranks of the value function. The TT approximation threshold $\delta = 10^{-4}$, the initial shift in Alg. 3 $\mu = 5$, and the polynomial degree $n - 1 = 4$. We see that the TT ranks stabilize as the dimension increases, and hence the CPU time grows linearly.

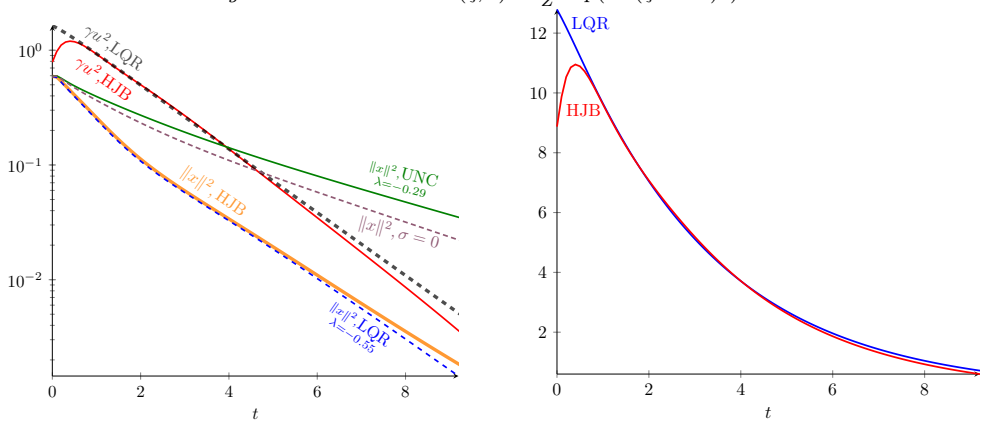
FIG. 4.10. *Left: CPU times and TT ranks for different dimensions d for the Fokker-Planck problem with the right-sided initial state* $x(\xi, 0) = \frac{1}{Z}\exp(-2(\xi - 3.8)^2)$. *Right: initial and equilibrium states.*



Moreover, we change the initial distribution to the right-sided state $x(\xi, 0) =$

$\frac{1}{Z}\exp(-2(\xi-3.8)^2)$, where $Z=\int_{-6}^{6}\exp(-2(\xi-3.8)^2)d\xi$ is the normalisation constant (Fig 4.10, right). It was observed [13] that the free system exhibits a very slow convergence to equilibrium when started from a right-sided distribution, since the particles must flow through a region of low probability. In Fig. 4.11 we show the components of the running cost for the *original* unshifted system, both UNControlled and controlled with HJB and LQR signals, obtained for the shifted system. We see that the free system converges at a slow rate $\|x\|^2 \sim \exp(-0.29t)$, while the controller computed for the de-stabilised system can accelerate this rate by almost a factor of 2. Note that when the HJB controller is computed for the original system ($\sigma = 0$), it accelerates the convergence only a little, so the shift is important to achieve the speedup. However, larger shifts make the HJB equation more difficult to solve. In particular, for larger shifts σ and larger state domain sizes a the stiffness matrix in (3.3) might become indefinite, and the policy iteration fails to converge. The domain size should be large enough to fit the trajectory, e.g. for the right-sided initial state the domain size of $a = 20$ is necessary to avoid excessive extrapolation of Legendre polynomials. This poses certain limitations on the range of possible applications of the TT-HJB approach. Nevertheless, when the policy iteration converges the HJB regulator can deliver a lower cost than LQR.

FIG. 4.11. *Running costs (left) and control signals (right) for the reduced Fokker-Planck problem with $d = 10$ with the right-sided initial state $x(\xi, 0) = \frac{1}{Z}\exp(-2(\xi - 3.8)^2)$.*



Conclusion. We have presented a numerical method for the solution of high-dimensional HJB PDEs arising in optimal feedback control for nonlinear dynamical systems. Our algorithm combines a continuous policy iteration together with a a tensor-train ansatz for the value function. An important matter of investigation is the identification of a class of optimal control problems where the value function can be accurately represented with a low-rank tensor train structure. For the class of optimal control problems we have explored in this work consisting of systems governed by nonlinear parabolic PDEs, we have consistently shown that the maximum TT rank in the value function approximation scales linearly with the dimension. This allows us to circumvent the curse of dimensionality up to a great extent, solving HJB PDEs with more than 100 dimensions. Control constraints are effectively enforced through penalties, despite the deterioration of the low-rank rank structure of the value function. The applications of the proposed methodology are extensive. In this work we have explored the synthesis of feedback control laws for high-dimensional dynamics

arising from the semi-discretisation of nonlinear PDEs. However, high-dimensional dynamics also play a crucial role in aerospace engineering [15], networks and agent-based models [1]. Finally, our methodology based on spectral approximation and tensor calculus, opens possibilities for a rigorous error analysis.

ACKNOWLEDGMENTS. S. Dolgov acknowledges the support of the Engineering and Physical Sciences Research Council through Fellowship EP/M019004/1. K. Kunisch was partially supported by the ERC advanced grant 668998 (OCLOC) under the EU's H2020 research program. This work was partially supported by the PGMO-PRMO program of the FMJH.

REFERENCES

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

[2] A. Alla, M. Falcone and D. Kalise. *An efficient policy iteration algorithm for dynamic programming equations*, SIAM J. Sci. Comput., 37(1)(2015):A181–A200, 2015.

[3] A. Alla, M. Falcone and S. Volkwein. *Error Analysis for POD Approximations of Infinite Horizon Problems via the Dynamic Programming Approach*, SIAM J. Control Optim. 55(5)(2017):3091-3115.

[4] 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)(2019):A2384–A2406.

[5] J. I. Alora, A. Gorodetsky, S. Karaman, Y. Marzouk, N. Lowry , *Automated synthesis of low-rank control systems from sc-LTL specifications using tensor-train decompositions* , IEEE 55th Conference on Decision and Control (CDC) (2016): 1131–1138.

[6] 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)(2008):817-848.

[7] R. W. Beard, G. N. , and J. T. Wen. *Galerkin approximation of the Generalized Hamilton-Jacobi-Bellman equation*, Automatica 33(12)(1997) 2159–2177.

[8] R. Bellman. *Functional equations in the theory of dynamic programming. V. Positivity and quasi-linearity*, Proc. Natl. Acad. Sci., 41(10)(1955):742–746.

[9] G. Belykin and M. Mohlenkamp. *Numerical operator calculus in higher dimensions*, Proc. Natl. Acad. Sci., 99(2002):10246-10251.

[10] P. Benner and T. Damm, *Lyapunov equations, energy functionals, and model order reduction of bilinear and stochastic systems*, SIAM Journal on Control and Optimization, 49 (2011):686–711.

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

[12] T. Breiten, K. Kunisch, and L. Pfeiffer, *Infinite-horizon bilinear optimal control problems: Sensitivity analysis and polynomial feedback laws*, SIAM Journal on Control and Optimization, 56 (2018):3184–3214.

[13] T. Breiten, K. Kunisch, and L. Pfeiffer, *Numerical study of polynomial feedback laws for a bilinear control problem*, Mathematical Control and Related Fields, 8 (2018):557.

[14] T. Breiten, K. Kunisch and L. Pfeiffer, *Feedback stabilization of the two-dimensional Navier-Stokes equations by value function approximation*, arXiv:1902.00394, 2019.

[15] J.-M. Biannic and J. Boada-Bauxel. *A Civilian Aircraft Landing Challenge*, `https://w3.onera.fr/smac/sites/w3.onera.fr.smac/files/CALC_v2.pdf` (2016).

[16] 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)(2013):575-605.

[17] 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(2019):376-409.

[18] M.G. Crandall and P.L. Lions. *Two approximations of solutions of Hamilton-Jacobi equations*, Math. Comp. 43(167)(1984):1-19.

[19] M.G. Crandall and P.L. Lions. *Hamilton-Jacobi equations in infinite dimensions I. Uniqueness of viscosity solutions*, J. Funct. Anal. 62(3)(1985):379–396.

[20] A. Dektor and D. Venturi. *Dynamically orthogonal tensor methods for high-dimensional nonlinear PDEs*, arXiv:1907.05924, 2019.

[21] S. Dolgov and B. Khoromskij, *Two-level QTT-Tucker format for optimized tensor calculus*, SIAM J. on Matrix An. Appl., 34 (2013):593–623.

[22] S. V. Dolgov and D. V. Savostyanov, *Alternating minimal energy methods for linear systems in higher dimensions*, SIAM J. Sci. Comput., 36 (2014):A2248–A2271.

[23] W. E, J. Han and A. Jentzen. *Deep learning-based numerical methods for high-dimensional parabolic partial differential equations and backward stochastic differential equations*, Comm. Math. Stat. 5(4)(2017):349-380.

[24] Y. Eidelman and I. Gohberg, *On a new class of structured matrices*, Integral Equations and Operator Theory, 34 (1999):293–324.

[25] M. Falcone, and R. Ferretti. *Numerical methods for Hamilton-Jacobi type equations*, Handb. Numer. Anal. 17(2016):603-626.

[26] J. Garcke and A. Kröner. *Suboptimal feedback control of PDEs by solving HJB equations on adaptive sparse grids*, J. Sci. Comput., doi:10.1007/s10915-016-0240-7 (2016).

[27] S. A. Goreinov, I. V. Oseledets, D. V. Savostyanov, E. E. Tyrtyshnikov, and N. L. Zamarashkin, *How to find a good submatrix*, in Matrix Methods: Theory, Algorithms, Applications, V. Olshevsky and E. Tyrtyshnikov, eds., World Scientific, Hackensack, NY (2010):247–256.

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

[29] L. Grasedyck, *Existence and computation of low Kronecker-rank approximations for large systems in tensor product structure*, Computing, 72 (2004):247–265.

[30] W. Hackbusch, *A sparse matrix arithmetic based on $\mathcal{H}$-matrices. Part I: Introduction to $\mathcal{H}$-matrices*, Computing, 62 (1999):89–108.

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

[32] C. Hartmann, B. Schäfer-Bund, and A. Thöns-Zueva, *Balanced averaging of bilinear systems with applications to stochastic control*, SIAM Journal on Control and Optimization, 51 (2013):2356–2378.

[33] S. Holtz, T. Rohwedder, and R. Schneider, *The alternating linear scheme for tensor optimization in the tensor train format*, SIAM J. Sci. Comput., 34 (2012):A683–A713.

[34] M. B. Horowitz, A. Damle and J. W. Burdick. *Linear Hamilton Jacobi Bellman equations in high dimensions*, Proc. IEEE/CDC 2014:5880-5887.

[35] C. Huré. H. Pham and X. Warin. *Some machine learning schemes for high-dimensional nonlinear PDEs*, arXiv:1902.01599 (2019).

[36] D. Kalise and A. Kröner. *Reduced-order minimum time control of advection-reaction-diffusion systems via dynamic programming,* Proc. 21st International Symposium on Mathematical Theory of networks and Systems, 1196-1202 (2014).

[37] 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)(2018):629-652.

[38] D. Kalise, S. Kundu, and and K. Kunisch. *Robust feedback control of nonlinear PDEs by numerical approximation of high-dimensional Hamilton-Jacobi-Isaacs equations*, arXiv:1905.06276, 2019.

[39] W. Kang and L. Wilcox. *Mitigating the Curse of Dimensionality: Sparse Grid Characteristics Method for Optimal Feedback Control and HJB Equations*, Comput. Optim. Appl. 68(2)(2017):289-315.

[40] B. Khoromskij. *Tensor numerical methods for multidimensional PDES: theoretical analysis and applications*, ESAIM: Proc. 48(2015):1-28.

[41] W. Hackbusch. *Tensor Spaces And Numerical Tensor Calculus*, Springer–Verlag, Berlin, 2012.

[42] T. Kolda and B. Bader. *Tensor Decompositions and Applications*, SIAM Rev. 51(3)(2009):455-500.

[43] K. Kunisch, S. Volkwein, L. Xie. *HJB-POD Based Feedback Design for the Optimal Control of Evolution Problems,* SIAM J. Appl. Dyn. Sys. 4(2004):701-722.

[44] S.E. Lyshevski. *Optimal control of nonlinear continuous-time systems: design of bounded controllers via generalized nonquadratic functionals*, Proc. American Control Conf. 1998, pp. 205–209.

[45] B. J. Matkowsky and Z. Schuss, *Eigenvalues of the Fokker-Planck operator and the approach to equilibrium for diffusions in potential fields*, SIAM Journal on Applied Mathematics, 40 (1981):242–254.

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

[47] T. Nakamura-Zimmerer, Q. Gong and W. Kang. *Adaptive Deep Learning for High Dimensional Hamilton-Jacobi-Bellman Equations*, arXiv:1907.05317, 2019.
[48] M. Neilan, A.J. Salgado and W. Zhang. *Numerical analysis of strongly nonlinear PDEs*, Acta Numerica 26(2017):137-303.
[49] I. V. Oseledets, *Constructive representation of functions in low-rank tensor formats*, Constr. Approx., 37 (2013):1–18.
[50] I. V. Oseledets. *Tensor-Train Decomposition*, SIAM J. Sci. Comput., 33(5)(2011):2295–2317.
[51] I. V. Oseledets and E. E. Tyrtyshnikov, *TT-cross approximation for multidimensional arrays*, Linear Algebra Appl., 432 (2010):70–88.
[52] M.L. Puterman and S.L. Brumelle. *On the Convergence of Policy Iteration in Stationary Dynamic Programming*, 4(1)(1979):60–69.
[53] M. Raissi and G.E. Karniadakis. *Hidden physics models: Machine learning of nonlinear partial differential equations*, J. Comput. Phys. 357(2018):125-141.
[54] C. Reisinger and Y. Zhang. *Rectified deep neural networks overcome the curse of dimensionality for nonsmooth value functions in zero-sum games of nonlinear stiff systems*, arXiv:1903.06652, 2019.
[55] D. V. Savostyanov, *Quasioptimality of maximum–volume cross interpolation of tensors*, Linear Algebra Appl., 458 (2014):217–244.
[56] R. Schneider and A. Uschmajew, *Approximation rates for the hierarchical tensor format in periodic Sobolev spaces*, Journal of Complexity, (2013).
[57] J. Sirignano and K. Spiliopoulos. *DGM: A deep learning algorithm for solving partial differential equations*, J. Comput. Phys. 375(2018):1339-1364.
[58] E. Sontag *Mathematical Control Theory: Deterministic Finite Dimensional Systems*, Second Edition, Springer-Verlag New York, 1998.
[59] E. Todorov. *Efficient computation of optimal actions*, Proc. Nat. Acad. Sci. 106(28)(2009):11478–11483.
[60] L. N. Trefethen, *Spectral methods in MATLAB*, SIAM, Philadelphia, 2000.
[61] E. E. Tyrtyshnikov, *Tensor approximations of matrices generated by asymptotically smooth functions*, Sbornik: Mathematics, 194 (2003):941–954.
[62] J. G. Verwer and J. M. Sanz-Serna. *Convergence of method of lines approximations to partial differential equations*, Computing 33(3-4)(1984):297–313.
[63] I. Yegorov and P.M. Dower. *Perspectives on Characteristics Based Curse-of-Dimensionality-Free Numerical Approaches for Solving Hamilton–Jacobi Equations*, Appl Math Optim (2018).
[64] M. Oster, L. Sallandt, R. Schneider *Approximating the Stationary Hamilton-Jacobi-Bellman Equation by Hierarchical Tensor Products*, arXiv:1911.00279, 2019.