2024 IEEE 63rd Conference on Decision and Control (CDC)
December 16-19, 2024. MiCo, Milan, Italy

Computing Solutions to the Polynomial-Polynomial Regulator Problem*

Nicholas A. Corbin[1] and Boris Kramer[2]

2024 IEEE 63rd Conference on Decision and Control (CDC) | 979-8-3503-1633-9/24/$31.00 ©2024 IEEE | DOI: 10.1109/CDC56724.2024.10885897

***Abstract*— We consider the optimal regulation problem for nonlinear control-affine dynamical systems. Whereas the linear-quadratic regulator (LQR) considers optimal control of a *linear* system with *quadratic* cost function, we study *polynomial* systems with *polynomial* cost functions; we call this problem the polynomial-polynomial regulator (PPR). The resulting polynomial feedback laws provide two potential improvements over linear feedback laws: 1) they more accurately approximate the optimal control law, resulting in lower control costs, and 2) for *some* problems they can provide a larger region of stabilization. We derive explicit formulas—and a scalable, general purpose software implementation—for computing the polynomial approximation to the value function that solves the optimal control problem. The method is illustrated first on a low-dimensional aircraft stall stabilization example, for which PPR control recovers the aircraft from more severe stall conditions than LQR control. Then we demonstrate the scalability of the approach on a semidiscretization of dimension $n = 129$ of a partial differential equation, for which the PPR control reduces the control cost by approximately 75% compared to LQR for the initial condition of interest.**

I. Introduction

Optimal control for linear dynamical systems has been well studied for decades. This is evidenced by the availability of simple software tools that allow practitioners to design optimal controllers with just a few lines of code, such as MATLAB's `lqr()`. These software tools have also been scaled to work for large-scale dynamical systems [1], such as those arising from semidiscretization of partial differential equations (PDEs). While nonlinear optimal control theory remains an active area of research, its adoption to large-scale nonlinear systems remains a major computational challenge. The development of scalable software tools thus remains another active research direction.

A computational challenge in nonlinear optimal control theory involves solving the Hamilton-Jacobi-Bellman (HJB) PDE for the *value function*. Many approaches exist for approximating the solutions to HJB PDEs, including state-dependent Riccati equations [2], algebraic Gramians [3]–[5], discretization techniques [6], iterative approaches [7], [8], and many others. In this paper, we expand on a recent body of work [9]–[13] bringing renewed interest to the method of Al'brekht [14], which is based on Taylor series expansions. This approach to solving HJB PDEs has been popular ever since it was introduced in the 1960s; however, since Al'brekht's results were presented in an abstract manner without closed-form solutions, the method historically was only applied to models with a few state dimensions and simplified dynamics in order to make the computations manageable to carry out [15], [16]. Scalable software tools automating these computations are required for general purpose use of Al'brekht's method, e.g. for larger state dimensions and for models with more than one or two nonlinearities.

Krener's Nonlinear Systems Toolbox (NST), originally introduced in 1997, was, to the authors' knowledge, the first openly available implementation of Al'brekht's method for general purpose use [17]. However, certain symbolic computations used in NST hindered its scalability, as detailed in [18]. Borggaard and Zietsman went on to provide an implementation based on the Kronecker product in [10], [11]; a similar approach was simultaneously presented by Almubarak et al. [12]. These approaches were shown to scale well to state-dimensions on the order of $n = 20$ to $n = 40$; however, these works only consider systems with polynomial drift, linear inputs, and quadratic cost functions. There are, however, applications requiring HJB PDE solutions for systems with polynomial input maps and even polynomial state-dependence in the cost function; see, for example, nonlinear balanced truncation model reduction [19]–[22].

The major contributions of this paper are to derive closed-form formulas, and existence of solution proofs, for value function approximations for systems with:

1) polynomial state-dependence in the drift *and* input map;
2) polynomial state-dependence in the state penalty in the cost function.

In addition to these two theoretical contributions, we provide an accompanying open-source, scalable implementation of the proposed algorithms for practical use. The function `ppr()` in the `cnick1/PPR` repository [23] acts as a nonlinear analog to MATLAB's `lqr()`.

II. Preliminaries

We review relevant optimal control theory in Section II-A, followed by a summary of Al'brekht's method in Section II-B. Kronecker product definitions and identities are then reviewed in Section II-C.

A. Optimal Control Theory

Consider the control-affine dynamical system

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t)) + \mathbf{g}(\mathbf{x}(t))\mathbf{u}(t), \tag{1}$$

*This work was supported by the National Science Foundation under Grant CMMI-2130727.

[1]N. Corbin is with the Department of Mechanical and Aerospace Engineering, University of California San Diego, La Jolla, CA 92093-0411 USA ncorbin@ucsd.edu

[2]B. Kramer is with the Faculty of the Department of Mechanical and Aerospace Engineering, University of California San Diego, La Jolla, CA 92093-0411 USA bmkramer@ucsd.edu

where: t is time, $\mathbf{x}(t) \in \mathbb{R}^n$ is the state, $\mathbf{u}(t) \in \mathbb{R}^m$ is the input, $\mathbf{f}\colon \mathbb{R}^n \to \mathbb{R}^n$ is the drift, and $\mathbf{g}\colon \mathbb{R}^n \to \mathbb{R}^{n\times m}$ is the input map. The optimal control problem for (1) seeks to find an input signal $\mathbf{u}(t)$ that minimizes the infinite-horizon scalar cost function

$$J(\mathbf{x}_0, \mathbf{u}) \coloneqq \frac{1}{2}\int_0^\infty \left(\mathbf{x}^\top \boldsymbol{Q}(\mathbf{x})\mathbf{x} + \mathbf{u}^\top \boldsymbol{R}(\mathbf{x})\mathbf{u}\right) \mathrm{d}t, \quad (2)$$

where $\boldsymbol{Q}(\mathbf{x}) \succeq \mathbf{0}$ and $\boldsymbol{R}(\mathbf{x}) \succ \mathbf{0}$ are nonnegative definite and positive definite symmetric matrix-valued functions, respectively, of appropriate dimensions.

Definition 1: The value the cost function takes under the action of the optimal control is given by *the value function*:

$$V(\mathbf{x}_0) \coloneqq \min_{\mathbf{u}} J(\mathbf{x}_0, \mathbf{u}). \quad (3)$$

The optimal control is denoted $\mathbf{u}_*$, so $V(\mathbf{x}_0) \equiv J(\mathbf{x}_0, \mathbf{u}_*)$. The next theorem summarizes the well-known result that the solution to the optimal control problem can be obtained by solving the HJB PDE.

Theorem 1 (e.g. [24], [25]): Assume that the cost (2) is continuously differentiable in all of its arguments and is strictly convex in $\mathbf{u}$. Then the value function is the solution to the HJB PDE

$$0 = \frac{\partial V^\top(\mathbf{x})}{\partial \mathbf{x}}\mathbf{f}(\mathbf{x}) - \frac{1}{2}\frac{\partial V^\top(\mathbf{x})}{\partial \mathbf{x}}\mathbf{g}(\mathbf{x})\boldsymbol{R}^{-1}(\mathbf{x})\mathbf{g}^\top(\mathbf{x})\frac{\partial V(\mathbf{x})}{\partial \mathbf{x}} + \frac{1}{2}\mathbf{x}^\top \boldsymbol{Q}(\mathbf{x})\mathbf{x}. \quad (4)$$

Furthermore, the optimal control $\mathbf{u}_*$ is given in feedback form by the gradient of the value function as

$$\mathbf{u}_*(\mathbf{x}) = -\boldsymbol{R}^{-1}(\mathbf{x})\mathbf{g}^\top(\mathbf{x})\frac{\partial V(\mathbf{x})}{\partial \mathbf{x}}. \quad (5)$$

Assuming a solution exists that satisfies the HJB PDE (4), then the optimal control is given by (5); hence, the optimal control problem reduces to solving the HJB PDE (4).

B. Al'brekht's Method

Computing solutions to the HJB PDE (4) is nontrivial and, in general, not possible analytically. In the interest of developing scalable algorithms, we adopt the approach of Al'brekht [14] to compute polynomial approximations to the value function. Al'brekht's method has three main features, which we summarize in the next theorem.

Theorem 2 (Al'brekht's method [14], [26]): Assume that the functions $\mathbf{f}(\mathbf{x})$ and $\mathbf{g}(\mathbf{x})$ in the dynamics, along with the functions $\boldsymbol{Q}(\mathbf{x})$ and $\boldsymbol{R}(\mathbf{x})$ in the cost function, are analytic. Also assume that a stabilizing solution exists to the LQR problem associated with the linearized dynamics[1]. Then:

1) the value function is analytic and can be approximated by a degree d polynomial;
2) the lowest degree polynomial term in the value function is degree 2, whose polynomial coefficient is given by the solution to the algebraic Riccati equation associated with the LQR problem on the linearized dynamics;
3) the remaining higher degree polynomial coefficients of the value function are solutions to linear algebraic equations that are entirely determined by the already-computed polynomial coefficients.

The remainder of this paper will focus on computing approximate solutions to the HJB PDE (4) using Al'brekht's method; the next section introduces Kronecker product notation to aid in that task.

C. Kronecker Product Definitions and Notation

The Kronecker product of two matrices $\mathbf{A} \in \mathbb{R}^{p\times q}$ and $\mathbf{B} \in \mathbb{R}^{s\times t}$ is the $ps \times qt$ block matrix

$$\mathbf{A} \otimes \mathbf{B} \coloneqq \begin{bmatrix} a_{11}\mathbf{B} & \cdots & a_{1q}\mathbf{B} \\ \vdots & \ddots & \vdots \\ a_{p1}\mathbf{B} & \cdots & a_{pq}\mathbf{B} \end{bmatrix},$$

where a_{ij} denotes the (i,j)th entry of $\mathbf{A}$. We write repeated Kronecker products as

$$\mathbf{x}^{\otimes k} \coloneqq \underbrace{\mathbf{x} \otimes \cdots \otimes \mathbf{x}}_{k \text{ times}} \in \mathbb{R}^{n^k}.$$

For $\mathbf{A} \in \mathbb{R}^{p\times q}$, we define the *$k$-way Lyapunov matrix* as

$$\mathcal{L}_k(\mathbf{A}) \coloneqq \sum_{i=1}^{k} \underbrace{\mathbf{I}_p \otimes \mathbf{A} \otimes \mathbf{I}_p \otimes \cdots \otimes \mathbf{I}_p}_{k \text{ factors, } \mathbf{A} \text{ in the } i\text{th position}} \in \mathbb{R}^{p^k \times p^{k-1} q}.$$

We also use the vec$[\cdot]$ operator, which stacks the columns of a matrix into one tall column vector, and the *perfect shuffle matrix* $\mathbf{S}_{q\times p}$ [27], [28], defined as the permutation matrix which shuffles vec$[\mathbf{A}]$ to match vec$\left[\mathbf{A}^\top\right]$:

$$\mathrm{vec}\left[\mathbf{A}^\top\right] = \mathbf{S}_{q\times p}\mathrm{vec}\left[\mathbf{A}\right]. \quad (6)$$

A concept which arises when dealing with Kronecker product polynomials is symmetry of the coefficients (a generalization of symmetry of a matrix), as defined next.

Definition 2 (Symmetric coefficients): Given a homogeneous polynomial of the form $\mathbf{v}_d^\top \mathbf{x}^{\otimes d}$, the coefficient $\mathbf{v}_k \in \mathbb{R}^{n^k \times 1}$ is *symmetric* if for all $\mathbf{a}_i \in \mathbb{R}^n$ it satisfies

$$\mathbf{v}_k^\top \left(\mathbf{a}_1 \otimes \mathbf{a}_2 \otimes \cdots \otimes \mathbf{a}_k\right) = \mathbf{v}_k^\top \left(\mathbf{a}_{i_1} \otimes \mathbf{a}_{i_2} \otimes \cdots \otimes \mathbf{a}_{i_k}\right),$$

where the indices $\{i_j\}_{j=1}^k$ are any permutation of $\{1, \ldots, k\}$.

From (6), one can see that if the matrix $\mathbf{A}$ is symmetric, the vector vec$[\mathbf{A}]$ is invariant under certain permutations. The next proposition formalizes this concept in terms of the perfect shuffle matrix and the definition of symmetry in Definition 2.

Proposition 1 (Permutation of symmetric coefficients): If a coefficient $\mathbf{v}_k \in \mathbb{R}^{n^k \times 1}$ is symmetric as per Definition 2, then

$$\mathbf{v}_k = \mathbf{S}_{n^j \times n^i}\mathbf{v}_k \qquad \forall i, j \geq 0 \quad \text{s.t.} \quad i + j = k.$$

Table III in Appendix A provides a collection of additional Kronecker product identities compiled from [27]–[30].

[1] In practice this is the main limitation/assumption for the method: the algebraic Riccati equation for the LQR problem must have a solution.

III. Kronecker Polynomial-Based Value Function Computations

For computational purposes, we consider a nonlinear control-affine dynamical system with polynomial structure

$$\dot{\mathbf{x}} = \underbrace{\mathbf{A}\mathbf{x} + \sum_{p=2}^{\ell} \mathbf{F}_p \mathbf{x}^{\textcircled{p}}}_{\mathbf{f}(\mathbf{x})} + \underbrace{\left(\sum_{p=1}^{\ell} \mathbf{G}_p \left(\mathbf{x}^{\textcircled{p}} \otimes \mathbf{I}_m \right) + \mathbf{B} \right)}_{\mathbf{g}(\mathbf{x})} \mathbf{u}, \tag{7}$$

where $\mathbf{A} \in \mathbb{R}^{n \times n}$, $\mathbf{F}_p \in \mathbb{R}^{n \times n^p}$, $\mathbf{B} \in \mathbb{R}^{n \times m}$, and $\mathbf{G}_p \in \mathbb{R}^{n \times mn^p}$. Often, the model of interest (1) is already polynomial, in which case system (7) is an exact representation. For example, many common PDEs, including Navier-Stokes, Kuramoto-Sivashinsky, Burgers, Allen-Cahn, Korteweg-de Vries, and Fokker-Planck all feature polynomial nonlinearities; upon spatial discretization, these all yield systems of the form (7). If system (1) is not exactly polynomial but is analytic, then system (7) is its Taylor approximation. We also assume the cost (2) to be analytic so that the functions $\boldsymbol{Q}(\mathbf{x})$ and $\boldsymbol{R}(\mathbf{x})$ can be expanded as real convergent power series. In this article, we will only consider state dependence in $\boldsymbol{Q}(\mathbf{x})$ and drop the state-dependence of $\boldsymbol{R}(\mathbf{x})$; in theory it is possible, but not trivial, to include it. Then we write the cost in terms of the Kronecker product as

$$J(\mathbf{x}_0, \mathbf{u}) \coloneqq \frac{1}{2} \int_0^{\infty} \left(\mathbf{x}^\top \mathbf{Q} \mathbf{x} + \mathbf{u}^\top \mathbf{R} \mathbf{u} + \sum_{p=3}^{\lambda} \mathbf{q}_p^\top \mathbf{x}^{\textcircled{p}} \right) \mathrm{d}t, \tag{8}$$

where $\mathbf{q}_p \in \mathbb{R}^{n^p}$ and we assume $\mathbf{Q} \succeq \mathbf{0}$ and $\mathbf{R} \succ \mathbf{0}$. In many cases, the cost function chosen by the control engineer is already polynomial and this representation is exact; otherwise, it can be viewed as a Taylor approximation.

Following the result of Theorem 2, the value function can be approximated as a degree d polynomial; in this work, we write this explicitly using the Kronecker product as

$$V(\mathbf{x}) \approx \frac{1}{2} \mathbf{x}^\top \mathbf{V}_2 \mathbf{x} + \frac{1}{2} \sum_{i=3}^{d} \mathbf{v}_i^\top \mathbf{x}^{\textcircled{i}}, \tag{9}$$

with the coefficients $\mathbf{v}_i \in \mathbb{R}^{n^i}$ for $i = 2, 3, \ldots, d$. The next theorem provides our main result, which is the explicit equations for the coefficients $\mathbf{v}_i$ in Kronecker product form.

Theorem 3 (Value function coefficients): Let the value function $V(\mathbf{x})$, which solves the HJB PDE (4) for the polynomial system (7), be of the form (9) with the coefficients $\mathbf{v}_i \in \mathbb{R}^{n^i}$ for $i = 2, 3, \ldots, d$. Then $\mathbf{v}_2 = \mathrm{vec}\,[\mathbf{V}_2]$, where $\mathbf{V}_2$ is the symmetric positive semidefinite solution to the algebraic Riccati equation

$$\mathbf{0} = \mathbf{A}^\top \mathbf{V}_2 + \mathbf{V}_2 \mathbf{A} - \mathbf{V}_2 \mathbf{B} \mathbf{R}^{-1} \mathbf{B}^\top \mathbf{V}_2 + \mathbf{Q}. \tag{10}$$

For $3 \leq k \leq d$, let $\tilde{\mathbf{v}}_k \in \mathbb{R}^{n^k}$ solve the linear system

$$\begin{aligned}
\mathcal{L}_k \left(\mathbf{A} - \mathbf{B}\mathbf{R}^{-1}\mathbf{B}^\top \mathbf{V}_2 \right)^\top \tilde{\mathbf{v}}_k = & - \sum_{\substack{i,p \geq 2 \\ i+p=k+1}} \mathcal{L}_i (\mathbf{F}_p)^\top \mathbf{v}_i \\
& - \mathbf{q}_k + \frac{1}{4} \sum_{\substack{i,j>2 \\ i+j=k+2}} ij \ \mathrm{vec}(\mathbf{V}_i^\top \mathbf{B} \mathbf{R}^{-1} \mathbf{B}^\top \mathbf{V}_j) \\
& + \frac{1}{4} \sum_{o=1}^{2\ell} \left(\sum_{\substack{p,q \geq 0 \\ p+q=o}} \left(\sum_{\substack{i,j \geq 2 \\ i+j=k-o+2}} ij \ \mathrm{vec} \left[\left(\mathbf{I}_{n^p} \otimes \mathrm{vec}\,[\mathbf{I}_m]^\top \right) \right. \right. \right. \\
& \times \left(\mathrm{vec} \left[\mathbf{G}_q^\top \mathbf{V}_j \right]^\top \otimes \left(\mathbf{G}_p^\top \mathbf{V}_i \otimes \mathbf{R}^{-1} \right) \right) \times \\
& \left. \left. \left. \left(\mathbf{I}_{n^{j-1}} \otimes \mathbf{S}_{n^{i-1} \times n^q m} \otimes \mathbf{I}_m \right) \left(\mathbf{I}_{n^{k-p}} \otimes \mathrm{vec}\,[\mathbf{I}_m] \right) \right] \right) \right)
\end{aligned} \tag{11}$$

Then the coefficient vector $\mathbf{v}_k = \mathrm{vec}\,(\mathbf{V}_k) \in \mathbb{R}^{n^k}$ for $3 \leq k \leq d$ is obtained by symmetrization of $\tilde{\mathbf{v}}_k$.

The proof of Theorem 3 can found in Appendix B; it consists of inserting the polynomial forms for the $\mathbf{f}(\mathbf{x})$, $\mathbf{g}(\mathbf{x})$, $\boldsymbol{Q}(\mathbf{x})$, and $V(\mathbf{x})$ into the HJB PDE (4). Collecting terms of the same polynomial degree leads to algebraic equations for each of the unknown value function coefficients $\mathbf{v}_i \in \mathbb{R}^{n^i}$ for $i = 2, 3, \ldots, d$.

Theorem 4 (Existence and uniqueness of solutions): Under the assumptions of Theorem 2, the linear system (11) has a unique solution for each coefficient $\mathbf{v}_k$ for $3 \leq k \leq d$.

Proof: The assumptions of Theorem 2 imply that $(\mathbf{A} - \mathbf{B}\mathbf{R}^{-1}\mathbf{B}^\top \mathbf{V}_2)$ is Hurwitz, which implies that $\mathcal{L}_k(\mathbf{A} - \mathbf{B}\mathbf{R}^{-1}\mathbf{B}^\top \mathbf{V}_2)$ is nonsingular. Hence the linear systems have unique solutions. ■

Remark 1: The cost of computing degree d approximations to the value function with Theorem 3 is $O(dn^{d+1})$ using our implementation, whereas a naive approach costs $O(n^{3d})$. The flop count is tedious, so we omit it here and refer the interested reader to our other publication [31] where similar details can be found. To summarize, the main algorithmic accelerations come from reshaping Kronecker products to leverage Basic Linear Algebra Subprograms (BLAS) operations and the use of a specialized linear solver [11], [32] that takes advantage of the *k-way Lyapunov structure* of the linear systems (11).

Remark 2: Given a degree d *approximation* to the value function computed using Theorem 3, a degree $d-1$ approximation to the optimal control is given by (5). This suboptimal polynomial feedback law has the form

$$\mathbf{u}(\mathbf{x}) = \mathbf{K}\mathbf{x} + \mathcal{K}^{[2]} \mathbf{x}^{\textcircled{2}} + \cdots + \mathcal{K}^{[d-1]} \mathbf{x}^{\textcircled{d-1}}. \tag{12}$$

The linear coefficient $\mathbf{K}$ is precisely the LQR gain matrix $\mathbf{K} = -\mathbf{R}^{-1}\mathbf{B}\mathbf{V}_2$. The higher-order gain matrices are obtained by collecting the terms of each degree from (5), where the polynomial form of $\mathbf{g}(\mathbf{x})$ is given in (7) and the gradient of the value function is given by (19).

IV. Numerical Examples

We demonstrate the approach for computing value functions and the associated controllers outlined in Theorem 3 on two examples. First, we consider an aircraft flight model with state dimension $n = 3$ in Section IV-A. Then, in Section IV-B, we demonstrate that the approach can be scaled to a semidiscretized model of the Allen-Cahn PDE with state dimension $n = 129$.

A. 3D Example: Aircraft Stall Stabilization

We consider an aircraft model used in [33] to formulate a nonlinear stall-stabilization controller for an F-8 Crusader cruising at $30,000$ ft at Mach $= 0.85$. In that work, the authors use an Al'brekht-based approach to compute nonlinear regulators by hand because a scalable, automated approach had not been developed. This tedious approach limited the authors to only consider cubic drift nonlinearities and no nonlinear input terms. Furthermore, they only consider quadratic-in-state costs. This problem was also considered in [12], but they also discard the nonlinear input terms. In contrast, our work provides the general purpose software to automate the computation so practitioners can easily implement nonlinear controllers including these terms.

Model: The derivation of the state-space model from Newton's second law can be found in [33]. The result is a state-space model in terms of the angle of attack x_1, the angle of the plane relative to the trim pitch x_2, and the rotation rate of the plane x_3, with the control input u corresponding to the angle of the tail elevator:

$$\begin{aligned}\dot{x}_1 &= x_3 - x_1^2 x_3 - 0.088 x_1 x_3 - 0.877 x_1 + 0.47 x_1^2 \\ &\quad - 0.019 x_2^2 + 3.846 x_1^3 - 0.215 u + 0.28 u x_1^2 \\ \dot{x}_2 &= x_3 \\ \dot{x}_3 &= -0.396 x_3 - 4.208 x_1 - 0.47 x_1^2 - 3.564 x_1^3 \\ &\quad - 20.967 u + 6.265 u x_1^2.\end{aligned} \tag{13}$$

Control Problem: We apply our algorithm to the same control problem presented in [33]: the model is subjected to disturbances in the angle of attack, physically corresponding to a gust of wind that puts the aircraft into a stall condition, which is reported in [33] to occur at an angle of $\alpha = 23.5°$. Hence given an initial condition $\mathbf{x}_0 = \begin{bmatrix}\alpha_0 & 0 & 0\end{bmatrix}^\top$, the objective is to design a controller that causes the plane to recover from stall. This is formulated as an optimal control problem where we seek a control $\mathbf{u}$ that minimizes the cost function

$$J(\mathbf{x}_0, \mathbf{u}) = \frac{1}{2}\int_0^\infty \left(\mathbf{x}^\top \mathbf{Q}\mathbf{x} + \mathbf{u}^\top \mathbf{R}\mathbf{u}\right)\, \mathrm{d}t$$

with $\mathbf{Q} = \frac{1}{4}\mathbf{I}$ and $\mathbf{R} = \mathbf{I}$ subject to the dynamics (13).

Results: We compute degree 2, 4, 6, and 8 approximations to the value function, giving degree 1, 3, 5 and 7 controllers of the form (12), denoted LQR, Cubic PPR, Quintic PPR, and Septic PPR, respectively. Fig. 1 shows the angle of attack response under the action of the different controllers for initial conditions of $\alpha_0 = 25°, 27°, 30°, 35°$. For the smallest angle of attack, $\alpha_0 = 25°$, all controllers—even the LQR—successfully stabilize the aircraft. However, as shown in Table I, the PPR controllers exhibit lower control costs, demonstrating the improved efficiency of polynomial feedback laws over linear feedback laws.

TABLE I. Aircraft control costs ($\alpha_0 = 25°$) computed up to $T = 12$. PPR controllers have a lower cost.

Controller	$\frac{1}{2}\int_0^T \left(\mathbf{x}^\top \mathbf{Q}\mathbf{x} + \mathbf{u}^\top \mathbf{R}\mathbf{u}\right)\, \mathrm{d}t$
LQR	0.053166
Cubic PPR	0.044503
Quintic PPR	0.040593
Septic PPR	0.039393

Regarding the ability to recover the aircraft from stall, we see a graceful degradation of the controllers. As the angle of attack increases, we gradually see each controller fail while the higher-order controllers are still able to stabilize the aircraft. The higher-order controllers achieve their improved performance by providing more rapid control inputs, so practitioners need to consider this when designing the controllers, i.e. picking the weights $\mathbf{Q}(\mathbf{x})$ and $\mathbf{R}$.

This simple example demonstrates two reasons to consider polynomial feedback laws: 1) the control costs can be lower, and 2) PPR controllers may work in cases where LQR fails. There are cases where LQR works while PPR fails though, so practitioners should always exercise caution.

B. Allen-Cahn Equation

We consider a semidiscretization of the Allen-Cahn PDE

$$\frac{\partial w(z,t)}{\partial t} = \epsilon \frac{\partial^2 w(z,t)}{\partial z^2} + w(z,t) - w(z,t)^3$$

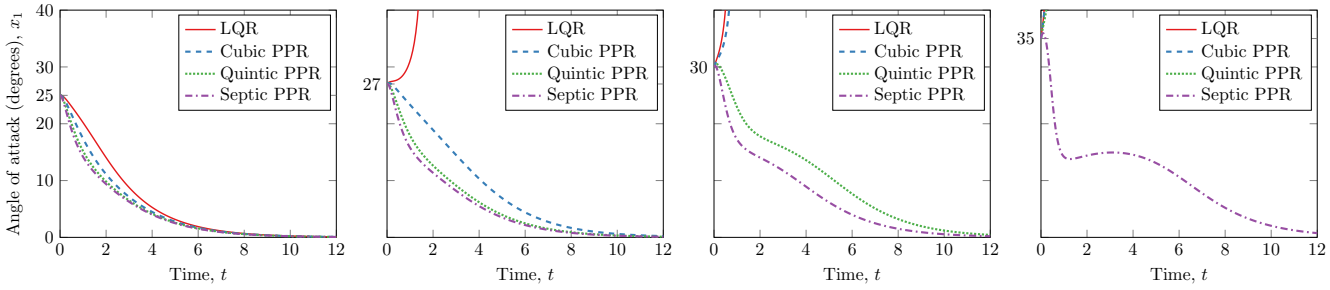


Fig. 1. Angle of attack response for initial conditions (from left to right) $\alpha_0 = 25°, 27°, 30°, 35°$. The PPR controllers stabilize the aircraft faster than LQR, and they are able to recover the aircraft from stall for larger initial angles of attack.

for $t > 0$ and $z \in \Omega = [-1, 1]$. The system is subject to the boundary conditions $w(-1, t) = -1$ and $w(1, t) = 1$, and the initial condition $w(z, 0) = 0.53z + 0.47\sin(-1.5\pi z)$ is chosen based on Example 34 in [34]. This reaction-diffusion PDE models phase separation in a two-phase mixture. The solution describes the evolution of interfaces between the two phases, with the diffusion coefficient ϵ determining the interface thickness. The system has the three constant equilibrium solutions $w_{\text{ss}}(z) = -1, 0, 1$. The solutions $w_{\text{ss}}(z) = \pm 1$ are stable and represent a pure single phase, whereas $w_{\text{ss}}(z) = 0$ is unstable and represents a homogeneous mixture of the two phases. The PDE also has a family of nontrivial steady-state solutions given by

$$w_{\text{ss}}(z; \epsilon, z_0) = \tanh\left(\frac{z - z_0}{\sqrt{2\epsilon}}\right), \tag{14}$$

where z_0 is the position of the interface. This is the only solution that satisfies the boundary conditions, so our problem is expected to converge to this type of solution.

Finite-Dimensional Model: To put the model in the form (7), the PDE is spatially discretized using a Chebychev pseudospectral collocation method with $n = 129$ nodes [34, Ex. 34], as depicted in Fig. 2. The model is also augmented with three control inputs, as shown in Fig. 2. The Allen-Cahn PDE has polynomial structure, so the only approximation comes from the finite element discretization.

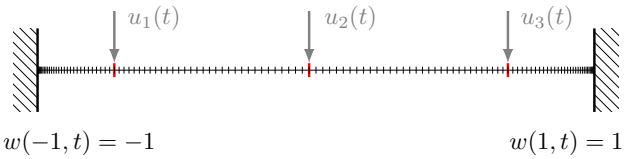


Fig. 2. Spatial domain and control locations for the Allen-Cahn PDE discretized with 129 Chebychev nodes, which are denser near the boundaries.

The semidiscretized model takes the form

$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{F}_3\mathbf{x}^{③} + \mathbf{B}\mathbf{u}. \tag{15}$$

The state $\mathbf{x}(t) = \begin{bmatrix} w(z_1, t), \ldots, w(z_n, t) \end{bmatrix}^\top \in \mathbb{R}^n$ represents the solution $w(z, t)$ evaluated at the Chebychev nodes $\mathbf{z} = \begin{bmatrix} z_1, \ldots, z_n \end{bmatrix}^\top$. The linear component of the drift is given by $\mathbf{A} = \epsilon \mathtt{D}_z^2 + \mathbf{I}$, where $\mathtt{D}_z^2$ is the differentiation matrix approximating the operator $\partial^2/\partial z^2$. The cubic drift coefficient $\mathbf{F}_3$ is a sparse binary matrix with ones in the positions to satisfy $\mathbf{F}_3\mathbf{x}^{③} = \mathbf{x} \odot \mathbf{x} \odot \mathbf{x}$, where $\odot$ is the Hadamard product (element-wise multiplication). Three independent control inputs are placed at nodes 33, 65, and 97, so the input matrix is $\mathbf{B} = \begin{bmatrix} \mathbf{e}_{33} & \mathbf{e}_{65} & \mathbf{e}_{97} \end{bmatrix}$, where $\mathbf{e}_i \in \mathbb{R}^n$ is the ith standard basis vector.

Fig. 3 shows the open-loop behavior of the system for $\epsilon = 0.01$. As described in [34], the system first exhibits a "metastable" configuration with three interfaces before suddenly transitioning to a steady-state solution of the form (14) with the interface at around $z_0 = 0$ at $t \approx 40$.

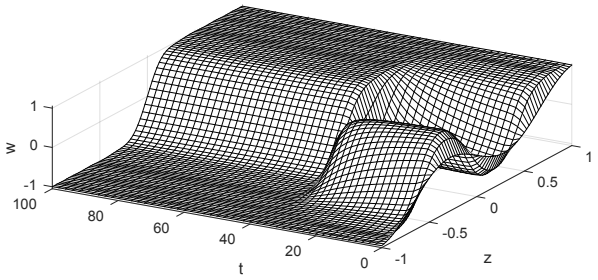


Fig. 3. Open-loop behavior of the Allen-Cahn example for $\epsilon = 0.01$, as shown in [34, Ex. 34].

Control Problem: Our objective is to dictate the location z_0 of the interface between the two phases in the steady-state solution. This is formulated as an optimal control problem seeking to asymptotically stabilize an equilibrium of the form (14) for a desired interface location z_0. The equilibrium at the origin $\mathbf{x} = \mathbf{0}$ of (15) corresponds to the unstable trivial solution $w_{\text{ss}}(z) = 0$, which does not satisfy the boundary conditions, so we shift the desired reference solution to the origin using a change of coordinates $\bar{\mathbf{x}} = \mathbf{x} - \mathbf{x}_{\text{ref}}$, where $\mathbf{x}_{\text{ref}} = \begin{bmatrix} w_{\text{ss}}(z_1; \epsilon, z_0), \ldots, w_{\text{ss}}(z_n; \epsilon, z_0) \end{bmatrix}^\top$ corresponds to a reference equilibrium solution defined in (14) for a particular interface location z_0. The cost function we choose is

$$J(\bar{\mathbf{x}}_0, \mathbf{u}) = \frac{1}{2}\int_0^\infty \left(\bar{\mathbf{x}}^\top\mathbf{Q}\bar{\mathbf{x}} + \mathbf{u}^\top\mathbf{R}\mathbf{u} + \mathbf{q}_4^\top\bar{\mathbf{x}}^{④}\right) \mathrm{d}t$$

with $\mathbf{Q} = 0.1\mathbf{I}$, $\mathbf{R} = \mathbf{I}$, and $\mathbf{q}_4$ the sparse vector satisfying $\mathbf{q}_4^\top\bar{\mathbf{x}}^{④} = \sum_{i=1}^n \bar{x}_i^4$ (analgous to the identity matrix for $\mathbf{Q}$).

Results: Selecting the desired interface location as $z_0 = 0.5$, we compute a standard LQR controller, a Quadratic PPR controller, and a Cubic PPR controller to stabilize the system to the desired solution. In addition to the diffusion parameter value $\epsilon = 0.01$ from [34], we also include results for the cases $\epsilon = 0.0075$ and $\epsilon = 0.005$. The smaller the diffusion coefficient, the more the cubic nonlinearity dominates the dynamics; this has the effect of making the metastable configuration more persistent and making it more difficult to stabilize the desired equilibrium.

Closed-loop simulations of the nonlinear system (15) are performed using MATLAB's `ode23s()` up to $T = 1000$. We then evaluate the cost function to quantitatively assess each of the controllers; the results are shown in Table II.

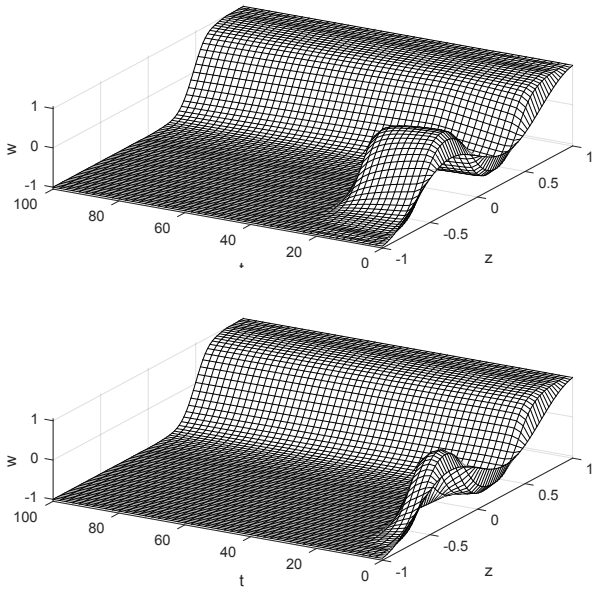
TABLE II. Cost $\frac{1}{2}\int_0^T \left(\bar{\mathbf{x}}^\top\mathbf{Q}\bar{\mathbf{x}} + \mathbf{u}^\top\mathbf{R}\mathbf{u} + \mathbf{q}_4^\top\bar{\mathbf{x}}^{④}\right) \mathrm{d}t$ integrated to $T = 1000$ for the Allen-Cahn example.

Controller	$\epsilon = 0.01$	$\epsilon = 0.0075$	$\epsilon = 0.005$
LQR	5475.640	19376.855	87268.670
Quadratic PPR	4339.483	14042.908	57876.913
Cubic PPR	1372.454	4153.668	20711.449

For the parameter values considered, the Cubic PPR controller has a cost about 75% lower than the LQR controller. As a rule of thumb, controllers of even degree are discouraged, since the associated value function (which locally acts as a Lyapunov function for the closed-loop dynamics) would be of odd degree, which is ill-advised. Nonetheless,

we have included the results for a Quadratic PPR controller for comparison purposes; it also performs better than LQR for this initial condition and these parameter values, but the improvement is not as significant as the Cubic PPR controller. Note that neither the Quadratic PPR nor the Cubic PPR controller solves the problem exactly; an infinite Taylor series would be required, and the solution is only guaranteed to converge locally. Still, including higher-order polynomial terms from the dynamics and the cost function noticably improves the performance of the controllers. In particular, the extra term in the cost function penalizes deviations from the reference configuration more heavily, so the PPR controllers are expected to stabilize the system more quickly and avoid the metastable configuration present in the open-loop solution.

Fig. 4a and Fig. 4b show the closed-loop system under the action of the LQR controller and the Cubic PPR controller, respectively, for the $\epsilon = 0.01$ case whose open-loop behavior is shown in Fig. 3. While both controllers stabilize the system to the desired solution, the Cubic PPR controller is able to perform the same task much more rapidly.



(b) Cubic PPR

Fig. 4. Allen-Cahn example closed-loop simulations for $\epsilon = 0.01$

.

V. Conclusions And Future Work

In this paper, we provided a general scalable approach and open-access software for computing solutions to the polynomial-polynomial regulator problem. The main contributions of this work include the development of algorithms capable of handling general polynomial state-dependence in drift, input map, and in the state penalty in the cost function. These contributions enable more accurate local approximations of optimal control laws, and the capability to include polynomial terms in the cost function gives practitioners more flexibility to tune controller behavior.

The results for the two examples considered herein demonstrate some of the potential benefits of polynomial controllers over linear controllers:

1) PPR controllers may have lower control costs than LQR for initial conditions sufficiently close to the origin;
2) PPR controllers may be able to stabilize initial conditions that LQR fails to stabilize.

However, these controllers also experience the following limitations, since they are based on Taylor expansions:

1) the solutions are only guaranteed to be valid within a neighborhood of the origin;
2) the region of attraction in the closed-loop system does not necessarily increase as more terms are added to the controller.

Practically speaking, there exist systems that are globally stabilized by LQR but only locally stabilized by PPR, so practitioners seeking the advantages of polynomial controllers should be wary of their limitations as well.

As future work, we plan to investigate further the practical role of higher-order terms in the cost function. While the $\mathbf{Q}$ and $\mathbf{R}$ matrices in LQR are well understood for trading controller performance for efficiency, the significance of higher-order terms such as $\mathbf{q}_4$ in the cost function is not yet well understood, in particular regarding the global behavior of the controllers. Additionally, the tensor structure of the higher-order value function and feedback gain coefficients is not fully understood. They are known to often have low numerical rank which can be exploited by some numerical methods, but a major limitation that remains is the RAM requirements for storing the higher-order coefficients. We plan to investigate this topic and other potential approximations for accelerating Al'brekht's method.

References

[1] P. Benner, J. Li, and T. Penzl, "Numerical solution of large-scale Lyapunov equations, Riccati equations, and linear-quadratic optimal control problems," *Numerical Linear Algebra with Applications*, vol. 15, no. 9, pp. 755–777, Nov. 2008.

[2] T. Çimen, "Survey of state-dependent Riccati equation in nonlinear optimal feedback control synthesis," *Journal of Guidance, Control, and Dynamics*, vol. 35, no. 4, pp. 1025–1047, Jul. 2012.

[3] M. Condon and R. Ivanov, "Nonlinear systems – algebraic Gramians and model reduction," *COMPEL - The international journal for computation and mathematics in electrical and electronic engineering*, vol. 24, no. 1, pp. 202–219, Mar. 2005.

[4] W. S. Gray and E. I. Verriest, "Algebraically defined gramians for nonlinear systems," in *2006 45th IEEE Conference on Decision and Control*. IEEE, Dec. 2006.

[5] P. Benner and P. Goyal, "Balanced truncation for quadratic-bilinear control systems," *Advances in Computational Mathematics*, vol. 50, no. 88, Aug. 2024.

[6] M. Falcone and R. Ferretti, "Numerical methods for Hamilton-Jacobi type equations," in *Handbook of Numerical Analysis*. Elsevier, 2016, pp. 603–626.

[7] D. Kalise and K. Kunisch, "Polynomial approximation of high-dimensional Hamilton-Jacobi-Bellman equations and applications to feedback control of semilinear parabolic PDEs," *SIAM Journal on Scientific Computing*, vol. 40, no. 2, pp. A629–A652, 2018.

[8] S. Dolgov, D. Kalise, and K. Kunisch, "Tensor decomposition methods for high-dimensional Hamilton-Jacobi-Bellman equations," *SIAM Journal on Scientific Computing*, vol. 43, no. 3, pp. A1625–A1650, Jan. 2021.

[9] T. Breiten, K. Kunisch, and L. Pfeiffer, "Taylor expansions of the value function associated with a bilinear optimal control problem," *Annales de l'Institut Henri Poincaré C, Analyse non linéaire*, vol. 36, no. 5, pp. 1361–1399, Aug. 2019.

[10] J. Borggaard and L. Zietsman, "The quadratic-quadratic regulator problem: approximating feedback controls for quadratic-in-state nonlinear systems," in *2020 American Control Conference (ACC)*, Jul. 2020, pp. 818–823.

[11] ——, "On approximating polynomial-quadratic regulator problems," *IFAC-PapersOnLine*, vol. 54, no. 9, pp. 329–334, 2021.

[12] H. Almubarak, N. Sadegh, and D. G. Taylor, "Infinite horizon nonlinear quadratic cost regulator," in *2019 American Control Conference (ACC)*, Jul. 2019, pp. 5570–5575.

[13] B. Kramer, S. Gugercin, J. Borggaard, and L. Balicki, "Scalable computation of energy functions for nonlinear balanced truncation," *Computer Methods in Applied Mechanics and Engineering*, vol. 427, p. 117011, Jul. 2024.

[14] E. G. Al'brekht, "On the optimal stabilization of nonlinear systems," *Journal of Applied Mathematics and Mechanics*, vol. 25, no. 5, pp. 1254–1266, Jan. 1961.

[15] W. L. Garrard, N. H. McClamroch, and L. G. Clark, "An approach to sub-optimal feedback control of non-linear systems," *International Journal of Control*, vol. 5, no. 5, pp. 425–435, May 1967.

[16] W. L. Garrard, "Suboptimal feedback control for nonlinear systems," *Automatica*, vol. 8, no. 2, pp. 219–221, Mar. 1972.

[17] A. J. Krener, "Nonlinear Systems Toolbox," Available on request to ajkrener@nps.edu, 2019.

[18] J. Borggaard and L. Zietsman, "Computation of nonlinear feedback for flow control problems," in *2018 American Control Conference (ACC)*. IEEE, Jun. 2018.

[19] J. M. A. Scherpen, "Balancing for nonlinear systems," *Systems & Control Letters*, vol. 21, no. 2, pp. 143–153, Aug. 1993.

[20] A. J. Krener, "Reduced order modeling of nonlinear control systems," in *Analysis and Design of Nonlinear Control Systems*. Springer Berlin Heidelberg, 2008, pp. 41–62.

[21] K. Fujimoto and D. Tsubakino, "Computation of nonlinear balanced realization and model reduction based on Taylor series expansion," *Systems & Control Letters*, vol. 57, no. 4, pp. 283–289, Apr. 2008.

[22] B. Kramer, S. Gugercin, and J. Borggaard, "Nonlinear balanced truncation: Part 2—model reduction on manifolds," Feb. 2023, *arXiv:2302.02036*.

[23] N. A. Corbin, "PPR repository," Available online: https://github.com/cnick1/PPR, Aug. 2024.

[24] R. E. Kalman, "The theory of optimal control and the calculus of variations," in *Mathematical Optimization Techniques*, R. Bellman, Ed. University of California Press, Dec. 1963, ch. 16, pp. 309–331.

[25] F. L. Lewis, *Optimal control*. Wiley, 2012.

[26] D. L. Lukes, "Optimal regulation of nonlinear dynamical systems," *SIAM Journal on Control*, vol. 7, no. 1, pp. 75–100, Feb. 1969.

[27] H. V. Henderson and S. R. Searle, "The vec-permutation matrix, the vec operator and Kronecker products: a review," *Linear and Multilinear Algebra*, vol. 9, no. 4, pp. 271–288, Jan. 1981.

[28] C. F. Van Loan, "The ubiquitous Kronecker product," *Journal of Computational and Applied Mathematics*, vol. 123, no. 1-2, pp. 85–100, Nov. 2000.

[29] J. Brewer, "Kronecker products and matrix calculus in system theory," *IEEE Transactions on Circuits and Systems*, vol. 25, no. 9, pp. 772–781, Sep. 1978.

[30] J. R. Magnus and H. Neudecker, *Matrix differential calculus with applications in statistics and econometrics*, 3rd ed. Wiley, Feb. 2019.

[31] N. A. Corbin and B. Kramer, "Scalable computation of $\mathcal{H}_\infty$ energy functions for polynomial control-affine systems," Aug. 2024, *arXiv:2408.08970*.

[32] M. Chen and D. Kressner, "Recursive blocked algorithms for linear systems with Kronecker product structure," *Numerical Algorithms*, vol. 84, no. 3, pp. 1199–1216, Sep. 2019.

[33] W. L. Garrard and J. M. Jordan, "Design of nonlinear automatic flight control systems," *Automatica*, vol. 13, no. 5, pp. 497–505, Sep. 1977.

[34] L. N. Trefethen, *Spectral methods in MATLAB*. Society for Industrial and Applied Mathematics, 2000.

APPENDIX

A. Kronecker Product Identities

TABLE III. Relevant Kronecker product identities.

ID 1	$(\mathbf{A}\otimes\mathbf{B})(\mathbf{D}\otimes\mathbf{G}) = \mathbf{AD}\otimes\mathbf{BG}$
ID 2	$\mathbf{A}\otimes\mathbf{B} = \mathbf{S}_{s\times p}(\mathbf{B}\otimes\mathbf{A})\mathbf{S}_{q\times t}$
ID 3	$(\mathbf{I}_p\otimes\mathbf{x})\mathbf{A} = \mathbf{A}\otimes\mathbf{x}$
ID 4	$\text{vec}\,[\mathbf{ADB}] = (\mathbf{B}^\top\otimes\mathbf{A})\text{vec}\,[\mathbf{D}]$
ID 5	$\text{vec}\,[\mathbf{AD}] = (\mathbf{I}_s\otimes\mathbf{A})\text{vec}\,[\mathbf{D}]$
ID 6	$\mathbf{u}^\top\mathbf{Bx} = \text{vec}\,[\mathbf{B}]^\top(\mathbf{x}\otimes\mathbf{u})$
ID 7	$\text{vec}\left[\mathbf{x}^\top\otimes\mathbf{I}_m\right] = (\mathbf{x}\otimes\text{vec}\,[\mathbf{I}_m])$
ID 8	$\text{vec}\,[\mathbf{A}\otimes\mathbf{B}] = (\mathbf{I}_q\otimes\mathbf{S}_{p\times t}\otimes\mathbf{I}_s)\,(\text{vec}\,[\mathbf{A}]\otimes\text{vec}\,[\mathbf{B}])$

Dimensions of matrices used in the Kronecker product identities

$\mathbf{A}(p\times q)$	$\mathbf{D}(q\times s)$	$\mathbf{u}(s\times 1)$
$\mathbf{B}(s\times t)$	$\mathbf{G}(t\times u)$	$\mathbf{x}(t\times 1)$

B. Proof of Theorem 3

Inserting the polynomial forms of $\mathbf{f}(\mathbf{x})$ and $\mathbf{g}(\mathbf{x})$ from (7) into the HJB PDE (4) gives

$$\begin{aligned}0 =&\frac{\partial V^\top(\mathbf{x})}{\partial\mathbf{x}}\left[\mathbf{Ax}+\sum_{p=2}^{\ell}\mathbf{F}_p\mathbf{x}^{\textcircled{p}}\right]\\&-\frac{1}{2}\frac{\partial V^\top(\mathbf{x})}{\partial\mathbf{x}}\left[\sum_{p=1}^{\ell}\mathbf{G}_p\left(\mathbf{x}^{\textcircled{p}}\otimes\mathbf{I}_m\right)+\mathbf{B}\right]\times\\&\mathbf{R}^{-1}\left[\sum_{q=1}^{\ell}\left(\mathbf{x}^{\textcircled{q}\top}\otimes\mathbf{I}_m\right)\mathbf{G}_q^\top+\mathbf{B}^\top\right]\frac{\partial V(\mathbf{x})}{\partial\mathbf{x}}\\&+\frac{1}{2}\mathbf{x}^\top\mathbf{Qx}+\frac{1}{2}\sum_{p=3}^{\lambda}\mathbf{q}_p^\top\mathbf{x}^{\textcircled{p}}\end{aligned}\tag{16}$$

The gradient of the value function (9) in Kronecker product form is

$$\begin{aligned}\frac{\partial^\top V(\mathbf{x})}{\partial\mathbf{x}} =&\frac{1}{2}\big(2\mathbf{x}^\top\mathbf{V}_2\\&+\mathbf{v}_3^\top(\mathbf{I}_n\otimes\mathbf{x}\otimes\mathbf{x})+\mathbf{v}_3^\top(\mathbf{x}\otimes\mathbf{I}_n\otimes\mathbf{x})+\mathbf{v}_3^\top(\mathbf{x}\otimes\mathbf{x}\otimes\mathbf{I}_n)\\&+\mathbf{v}_4^\top(\mathbf{I}_n\otimes\mathbf{x}\otimes\mathbf{x}\otimes\mathbf{x})+\mathbf{v}_4^\top(\mathbf{x}\otimes\mathbf{I}_n\otimes\mathbf{x}\otimes\mathbf{x})\\&+\mathbf{v}_4^\top(\mathbf{x}\otimes\mathbf{x}\otimes\mathbf{I}_n\otimes\mathbf{x})+\mathbf{v}_4^\top(\mathbf{x}\otimes\mathbf{x}\otimes\mathbf{x}\otimes\mathbf{I}_n)+\cdots\big),\end{aligned}\tag{17}$$

where without loss of generality we assume that $\mathbf{V}_2$ is symmetric. After inserting the polynomial expressions for the dynamics (7) and the value functions (9), the HJB PDE (16) yields an algebraic equation for each of the coefficients $\mathbf{v}_i$ for $i = 2, 3, \ldots, d$. The collection of degree 2 terms in (16) is

$$0 =\mathbf{x}^\top\mathbf{V}_2\mathbf{Ax}-\frac{1}{2}\mathbf{x}^\top\mathbf{V}_2\mathbf{BR}^{-1}\mathbf{B}^\top\mathbf{V}_2\mathbf{x}+\frac{1}{2}\mathbf{x}^\top\mathbf{Qx}.$$

Differentiating twice with respect to $\mathbf{x}$ yields the algebraic Riccati equation (10) for $\mathbf{V}_2$.

The equation for $\mathbf{v}_k$ for $k = 3, 4, \ldots, d$ is obtained by collecting the terms of degree k in (16); recall that as a consequence of Theorem 2, these are linear algebraic

equations. Thus we separate the collection of degree k terms in (16) as $0 = -\mathit{LHS} + \mathit{RHS}$, where *LHS* denotes the terms containing the unknown $\mathbf{v}_k$ and *RHS* contains the sum of all of the remaining terms [2]. The collection of degree k terms containing $\mathbf{v}_k$ is

$$\mathit{LHS} \coloneqq -\frac{1}{2}\mathbf{v}_k^\top \left((\mathbf{I}_n \otimes \mathbf{x}^{\overbrace{k-1}}) + (\mathbf{x} \otimes \mathbf{I}_n \otimes \mathbf{x}^{\overbrace{k-2}}) + \cdots\right) \times (\mathbf{A} - \mathbf{B}\mathbf{R}^{-1}\mathbf{B}^\top \mathbf{V}_2)\mathbf{x}.$$

With careful algebraic manipulations, the Kronecker products can be reordered to rewrite this collection of terms using the *k-way Lyapunov matrix*

$$\mathit{LHS} = -\frac{1}{2}\mathbf{v}_k^\top \mathcal{L}_k(\mathbf{A} - \mathbf{B}\mathbf{R}^{-1}\mathbf{B}^\top \mathbf{V}_2)\mathbf{x}^{\textcircled{k}}. \tag{18}$$

The remaining terms in the HJB PDE, which make up the right-hand-side of the linear algebraic equation for $\mathbf{v}_k$, only contain coefficients $\mathbf{v}_2$ through $\mathbf{v}_{k-1}$, which have already been computed. As such, they are symmetric according to Definition 2. This allows us to rewrite the gradient of the value function (17) using ID 2 and Proposition 1 as

$$\begin{aligned}\frac{\partial^\top V(\mathbf{x})}{\partial \mathbf{x}} &= \frac{1}{2}\left(2\mathbf{v}_2^\top(\mathbf{I}_n \otimes \mathbf{x}) + 3\mathbf{v}_3^\top(\mathbf{I}_n \otimes \mathbf{x} \otimes \mathbf{x}) + \cdots\right) \\ &= \frac{1}{2}\sum_{i=2}^{k-1} i\mathbf{v}_i^\top(\mathbf{I}_n \otimes \mathbf{x}^{\overbrace{i-1}}).\end{aligned} \tag{19}$$

Rewriting the remaining terms in the HJB PDE (16) with the new gradient of the value function (19), the *RHS* terms can be written as

$$\mathit{RHS} \coloneqq \frac{1}{2}\left[\sum_{i=2}^{k-1} i\mathbf{v}_i^\top(\mathbf{I}_n \otimes \mathbf{x}^{\overbrace{i-1}})\right]\left[\sum_{p=2}^{\ell}\mathbf{F}_p\mathbf{x}^{\textcircled{p}}\right] + \frac{1}{2}\sum_{p=3}^{\lambda}\mathbf{q}_p^\top\mathbf{x}^{\textcircled{p}} \tag{20a}$$

$$-\frac{1}{8}\left[\sum_{i=2}^{k-1} i\mathbf{v}_i^\top(\mathbf{I}_n \otimes \mathbf{x}^{\overbrace{i-1}})\right]\left[\sum_{p=1}^{\ell}\mathbf{G}_p\left(\mathbf{x}^{\textcircled{p}} \otimes \mathbf{I}_m\right) + \mathbf{B}\right] \times \tag{20b}$$

$$\mathbf{R}^{-1}\left[\sum_{q=1}^{\ell}\left(\mathbf{x}^{\textcircled{q}^\top} \otimes \mathbf{I}_m\right)\mathbf{G}_q^\top + \mathbf{B}^\top\right]\left[\sum_{j=2}^{k-1}(\mathbf{I}_n \otimes \mathbf{x}^{\overbrace{j-1}^\top})\mathbf{v}_j j\right]. \tag{20c}$$

Again using careful algebraic manipulations, these terms can be manipulated to a more desirable form; specifically, the goal is to put these in the form $(\cdot)\,\mathbf{x}^{\textcircled{k}}$ so that we may match coefficients of the same polynomial degree. Beginning with the terms from (20a) containing $\mathbf{F}_p$

$$\frac{1}{2}i\mathbf{v}_i^\top(\mathbf{I}_n \otimes \mathbf{x}^{\overbrace{i-1}})\mathbf{F}_p\mathbf{x}^{\textcircled{p}}, \quad \text{with} \quad p + i - 1 = k,$$

a similar trick to that used in (18) allows us to rewrite this using the *k-way Lyapunov matrix* as

$$\frac{1}{2}\mathbf{v}_i^\top \mathcal{L}_i(\mathbf{F}_p)\mathbf{x}^{\textcircled{k}}. \tag{21}$$

The terms containing $\mathbf{q}_p$ are already in the desired form

$$\frac{1}{2}\mathbf{q}_p^\top\mathbf{x}^{\textcircled{p}} \qquad \text{with} \qquad p = k. \tag{22}$$

[2]Eventually, *LHS* will be the left-hand-side of the linear algebraic equations for $\mathbf{v}_k$, and *RHS* will be the right-hand-side, i.e. $\mathit{LHS} = \mathit{RHS}$.

The terms containing $\mathbf{B}$ are

$$\frac{1}{8}i\mathbf{v}_i^\top(\mathbf{I}_n \otimes \mathbf{x}^{\overbrace{i-1}})\mathbf{B}\mathbf{R}^{-1}\mathbf{B}^\top(\mathbf{I}_n \otimes \mathbf{x}^{\overbrace{j-1}^\top})\mathbf{v}_j j, \tag{23}$$

with $i + j - 2 = k$. After algebraic manipulations, these terms are written as

$$\frac{1}{8}ij\mathrm{vec}\left[\mathbf{V}_i^\top\mathbf{B}\mathbf{R}^{-1}\mathbf{B}^\top\mathbf{V}_j\right]^\top\mathbf{x}^{\textcircled{k}}. \tag{24}$$

The remaining terms, which are those containing $\mathbf{G}_p$, are

$$\begin{aligned}\frac{1}{8}i\mathbf{v}_i^\top(\mathbf{I}_n \otimes \mathbf{x}^{\overbrace{i-1}})\mathbf{G}_p(\mathbf{x}^{\textcircled{p}} \otimes \mathbf{I}_m)\times \\ \mathbf{R}^{-1}(\mathbf{x}^{\textcircled{q}^\top} \otimes \mathbf{I}_m)\mathbf{G}_q^\top(\mathbf{I}_n \otimes \mathbf{x}^{\overbrace{j-1}^\top})\mathbf{v}_j j,\end{aligned} \tag{25}$$

with $p \in [0, o]$, $o \in [1, 2\ell]$, $q = o - p$, and $i + j + o = k + 2$. Here the manipulations are much more involved, but ultimately they just require careful application of the identities in Table III. After these manipulations, the terms containing $\mathbf{G}_p$ are written as

$$\begin{aligned}\frac{1}{8}ij\mathrm{vec}\Bigg[\left(\mathbf{I}_{n^p} \otimes \mathrm{vec}\left[\mathbf{I}_m\right]^\top\right)\Bigg(\mathrm{vec}\left[\mathbf{G}_q^\top\mathbf{V}_j\right]^\top \otimes \\ \left(\mathbf{G}_p^\top\mathbf{V}_i \otimes \mathbf{R}^{-1}\right)\Bigg)\left(\mathbf{I}_{n^{j-1}} \otimes \mathbf{S}_{n^{i-1}\times n^q m} \otimes \mathbf{I}_m\right) \times \\ \left(\mathbf{I}_{n^{k-p}} \otimes \mathrm{vec}\left[\mathbf{I}_m\right]\right)\Bigg]^\top\mathbf{x}^{\textcircled{k}}.\end{aligned} \tag{26}$$

The quantities (21), (22), (24) and (26) represent single degree k terms containing $\mathbf{F}_p, \mathbf{q}_p, \mathbf{B}$, and $\mathbf{G}_p$ contributions, respectively. Introducing summations over the appropriate combinations of indices, the HJB PDE (16), written now as $\mathit{LHS} = \mathit{RHS}$, can be expanded as

$$\begin{aligned}-\frac{1}{2}\mathbf{v}_k^\top\mathcal{L}_k(\mathbf{A} - \mathbf{B}\mathbf{R}^{-1}\mathbf{B}^\top\mathbf{V}_2)\mathbf{x}^{\textcircled{k}} &= \frac{1}{2}\sum_{\substack{i,p\geq 2\\ i+p=k+1}}\mathbf{v}_i^\top\mathcal{L}_i(\mathbf{F}_p)\mathbf{x}^{\textcircled{k}} \\ &+ \frac{1}{2}\mathbf{q}_k^\top\mathbf{x}^{\textcircled{k}} - \frac{1}{8}\sum_{\substack{i,j>2\\ i+j=k+2}} ij\ \mathrm{vec}(\mathbf{V}_i^\top\mathbf{B}\mathbf{R}^{-1}\mathbf{B}^\top\mathbf{V}_j)\mathbf{x}^{\textcircled{k}} \\ -\frac{1}{8}\sum_{o=1}^{2\ell}\Bigg(\sum_{\substack{p,q\geq 0\\ p+q=o}}\Bigg(\sum_{\substack{i,j\geq 2\\ i+j=k-o+2}} ij\ \mathrm{vec}\Bigg[\left(\mathbf{I}_{n^p} \otimes \mathrm{vec}\left[\mathbf{I}_m\right]^\top\right) \times \\ \left(\mathrm{vec}\left[\mathbf{G}_q^\top\mathbf{V}_j\right]^\top \otimes \left(\mathbf{G}_p^\top\mathbf{V}_i \otimes \mathbf{R}^{-1}\right)\right) \times \\ \left(\mathbf{I}_{n^{j-1}} \otimes \mathbf{S}_{n^{i-1}\times n^q m} \otimes \mathbf{I}_m\right)\left(\mathbf{I}_{n^{k-p}} \otimes \mathrm{vec}\left[\mathbf{I}_m\right]\right)\Bigg]^\top\Bigg)\Bigg)\Bigg)\mathbf{x}^{\textcircled{k}}\end{aligned}$$

We require this to hold for all $\mathbf{x}$. Pulling out the factor of $\mathbf{x}^{\textcircled{k}}$ from every term, multiplying by negative two, and transposing the entire equation results in (11) and completes the proof for Theorem 3. ■