A POD Projection Method for Large-Scale Algebraic Riccati Equations

Boris Kramer
Department of Mathematics
Virginia Tech
Blacksburg, VA 24061-0123
Email: bokr@vt.edu

John R. Singler
Department of Mathematics and Statistics
Missouri University of Science and Technology
Rolla, MO 65409-0020
Email: singlerj@mst.edu

Abstract

The solution of large-scale matrix algebraic Riccati equations (AREs) is important for instance in control design and model reduction and remains an active area of research. We propose a projection method to obtain low rank solutions of AREs based on simulations of linear systems coupled with proper orthogonal decomposition (POD). The method can take advantage of existing (black box) simulation code to generate the projection matrices. Furthermore, simulation methods such as parallel computation, domain decomposition, and multigrid methods can be exploited to increase accuracy and efficiency and to solve large-scale problems. We present numerical results demonstrating that the proposed approach can produce highly accurate approximate solutions. We also briefly discuss making the proposed approach completely data-based so that one can use existing simulation codes without accessing system matrices.

1 Introduction

Riccati equations play an important role in a variety of problems, including optimal control and filtering. For instance, the solution to the algebraic Riccati equation (ARE) determines the optimal feedback control solving the linear quadratic regulator problem. Such feedback control laws are used to stabilize a dynamical system and to steer the dynamics to desired equilibrium states. Moreover, the problem of optimal state estimation from given measurements of a linear dynamical system also involves a solution to an algebraic Riccati equation. For details about control and estimation, see [50, Chapter 4] and [23, Chapter 12] and the many references therein. Solutions of AREs are also important for certain model reduction algorithms, such as LQG balanced truncation, [3, Section 7.5] and [70, 13]. Furthermore, optimizing sensor and actuator locations in optimal control problems can require the solution of many AREs; see, e.g., [27, 28, 45, 26] and the references therein. Hence, solving nonlinear matrix algebraic Riccati equations of the form

\[ A^T PE + E^T PA - E^T PBB^T PE + C^T C = 0 \] (1)

is a key step in many applications. Here, \( A, E \in \mathbb{R}^{n \times n} \), \( B \in \mathbb{R}^{n \times m} \), and \( C \in \mathbb{R}^{p \times n} \) are given matrices, and \( P \in \mathbb{R}^{n \times n} \) is the unknown matrix. Under reasonably weak assumptions on \( (E, A, B, C) \), the exact solution \( P \) of the ARE (1) exists and is symmetric positive semidefinite; e.g., [50, Theorem 3.7].

We are concerned with approximating the solution of the ARE (1) for large-scale systems, i.e., when \( n > 10,000 \). One instance where such AREs arise in the applications mentioned above is when the linear dynamical system is derived from a spatial discretization of a partial differential
equation using, e.g., finite element methods. The dimension $n$ can easily be on the order of $10^6$ or larger for such applications. Even if the matrices $E$ and $A$ are sparse and $m, p \ll n$, the exact solution $P$ of the ARE is a dense $n \times n$ matrix; therefore obtaining or even storing the exact solution is infeasible for many problems. Fortunately, the solution $P$ is often of low numerical rank when $p, m \ll n$ [17, 54] and many recent solution approaches exploit this by constructing factored low rank approximate solutions of the form $P \approx ZZ^T$.

Over the past 50 years many methods and techniques have been developed to efficiently solve small or moderate sized nonlinear matrix equations of Riccati type; see, e.g., [18] for an overview. A large amount of recent research has been devoted to the development and analysis of algorithms for large-scale AREs; see the recent survey [17]. Many of the approaches are inspired by computational linear algebra methods and include Krylov subspace methods in a projection framework [58, 40, 41, 42, 38, 63], Kleinman-Newton methods [47, 12, 21, 33, 16], and subspace iterations [52, 14].

There has also been interest in developing data-based algorithms that approximate the solution of the ARE using only simulation data. Such algorithms do not require direct access to the matrices $(E, A, B, C)$. This can be important if one has an existing (possibly complex) simulation code for which it is difficult or impossible to access the relevant matrices. In such a data-based setting, researchers have not typically focused on approximating the solution $P$ of the Riccati equation; instead, researchers have primarily focused on approximating quantities that depend on the Riccati solution. For example, one can attempt to approximate the feedback gain matrix $K = B^TPE \in \mathbb{R}^{m \times n}$ needed for optimal feedback control problems, or one can attempt to approximate the optimal control only (which is not in feedback form).

Data-based algorithms include approaches based on the Chandrasekhar equations [44, 53, 22, 20, 19, 1], iterative optimization algorithms [55, 59], and a vast variety of model reduction methods (see, e.g., [6, 5, 9, 8, 51]). All of these approaches have been used successfully on a variety of problems, but they can have drawbacks. For example, the Chandrasekhar equations are nonlinear and can be costly to simulate. Also, the iterative optimization algorithms only provide the optimal control which cannot be used for feedback purposes. Furthermore, these methods have typically not aimed to provide highly accurate approximations of the Riccati solution $P$.

We propose a new projection based method in Section 4 to solve AREs based on simulations of linear systems coupled with proper orthogonal decomposition (POD) and Galerkin projection. The proposed approach is a first step towards an accurate, completely data-based projection method for solution of AREs, and is particularly formulated for researchers familiar with POD. For readers not familiar with projection methods, we give a brief overview in Section 3. After describing the POD-projection approach, we present numerical results in Section 5 indicating that the proposed method can be used to accurately compute the feedback gain matrices and also provide an accurate very low rank approximate solution of the ARE, which can be used for model reduction applications, etc. In the conclusion, we briefly discuss transitioning the proposed approach into a completely data-based algorithm.

We note that a data-based POD approach has been previously coupled with a Kleinman-Newton iteration in [65] to approximately solve AREs. However, the recent work [63] indicates that projection methods outperform approaches based on the Kleinman-Newton iteration.

To relate our numerical findings to state-of-the-art low rank Riccati solution methods for large-scale problems, we choose an extended Krylov subspace projection method. In Section 5, we find that the proposed POD approach gives high accuracy for a fixed approximation rank and is tractable for large-scale applications. The proposed method generally is not as computationally and storage efficient as Krylov methods or other computational linear algebra approaches. However,

\footnote{A preliminary version of this algorithm can be found in [48].}
the primary goal of this work is not to produce the most efficient solver; instead, our goal is to move toward an efficient, accurate, completely data-based method that can take advantage of existing simulation codes. In addition, when the matrices come from discretizations of PDEs, efficient domain decomposition and multigrid techniques can be used to solve the linear systems, giving the method additional flexibility. Moreover, for systems with multiple outputs, the required simulations can be run in parallel.

2 Notation, Assumptions and Background

Throughout this paper, we consider linear time invariant systems \( \Sigma = (E, A, B, C) \) given by

\[
\begin{align*}
E \dot{x}(t) &= Ax(t) + Bu(t), \quad (2) \\
y(t) &= Cx(t), \quad (3)
\end{align*}
\]

where \( E, A \in \mathbb{R}^{n \times n}, \ B \in \mathbb{R}^{n \times m} \) and \( C \in \mathbb{R}^{p \times n} \) and \( m, p \ll n \). Here, \( \dot{x} = dx/dt \) denotes the usual time derivative of \( x \). We assume the matrix \( E \) is invertible.

For certain systems, such as those arising in the spatial discretization of the linearized Navier Stokes equations (e.g., \([2]\)), the system (2)-(3) is not stable in the sense that \( E^{-1}A \) has positive eigenvalues. A system is called asymptotically stable if all eigenvalues of \( E^{-1}A \) have negative real part. Stabilizing unstable systems is one important problem in feedback control applications. Whenever the system is unstable, the observability Gramian given by representation (6) below does not exist and certain ad hoc solution strategies for ARE can have poor performance. To make \( E^{-1}A \) stable, one first finds a stabilizing feedback gain \( K_1 \), for instance via the algebraic Bernoulli equations \([2]\) or by integrating the Chandrasekhar equations until a stabilizing feedback gain is obtained \([10]\). The new matrix \( E^{-1}(A - BK_1) \) is then stable.

For convenience we assume that \( E^{-1}A \) is stable. In light of the preceding discussion, we emphasize that this is not a necessary assumption. In fact, the assumption of stability of \( E^{-1}A \) solely implies that the stabilizing feedback has already been applied.

Many algorithms to solve Riccati equations exploit the intrinsic connection to the linear Lyapunov equation

\[
A^T X + X A + C^T C = 0. \quad (4)
\]

One should note that (4) is a linear matrix equation obtained from the algebraic Riccati equation (1) by ignoring the nonlinear term. The connection between the Riccati and Lyapunov equation can have important implications when devising algorithms to solve AREs. In this paper, we generate a subspace spanned by the singular vectors of an approximate Lyapunov solution to compute approximate solutions for AREs via projection.

Assuming that \( E^{-1}A \) is stable, the solution to Lyapunov equation admits a closed form. To see this, rewrite the Lyapunov equation (4) above as

\[
\bar{A}^T \bar{X} + \bar{X} \bar{A} + \bar{C}^T \bar{C} = 0, \quad (5)
\]

where

\[
\bar{X} = E^T X E, \quad \bar{A} = E^{-1} A, \quad \bar{C} = C.
\]

It is well known that the solution of the transformed Lyapunov equation (5) has the following integral representation:
Theorem 2.1. ([3, Proposition 4.27]) If $\tilde{A} = E^{-1}A$ is a stable matrix, then the Lyapunov equation (5) has a unique solution $\tilde{X}$, called the observability Gramian, which has the representation

$$\tilde{X} = \int_0^\infty e^{t\tilde{A}^T} \tilde{C} \tilde{C}^T e^{t\tilde{A}} dt. \tag{6}$$

The solution of the original Lyapunov equation (4) can be expressed in a similar form.

Corollary 2.2. If $\tilde{A} = E^{-1}A$ is a stable matrix, then the Lyapunov equation (4) has a unique solution $X$ given by

$$X = \int_0^\infty e^{tE^{-T}A^{T}E^{-T}C} \tilde{C} e^{tE^{-1}} dt.$$  

Proof. The above theorem gives

$$X = \int_0^\infty E^{-T} e^{t(E^{-1}A)^T} C \tilde{C} e^{tE^{-1}} dt.$$  

Using $M^{-1}e^NM = e^{M^{-1}NM}$ for $M = E^T$, $N = AT_t$ and also $M = E$, $N = At$ gives the result.  

We denote the standard Euclidean inner product and norm by $(x,y) = y^T x$ and $\|x\| = (x,x)^{1/2} = (x^T x)^{1/2}$. We also use the standard matrix 2-norm defined by $\|A\|_2 = \sup_{\|x\|=1} \|Ax\|_2 = \sigma_1(A)$ and the Frobenius norm given by $\|A\|_F^2 = \text{trace}(A^T A) = \sum_{i=1}^n \sigma_i^2$. Here, $\sigma_1 \geq \sigma_2 \geq \cdots \geq 0$ denote the singular values of $A$.

In some problems, it is more natural to work with weighted norms on $\mathbb{R}^n$. For example, in many PDE systems the natural state spaces are the Hilbert spaces $L^2$ or $H^1$. Thus, when the system (2)-(3) arises from a standard finite element spatial discretization of a partial differential equation system, the matrix $E$ is symmetric positive definite and the $E$-weighted norm of a vector equals the $L^2$ norm of the corresponding finite element function. In general, for a symmetric positive definite matrix $W \in \mathbb{R}^{n \times n}$, let $(x,y)_W = y^T W x$ denote the $W$-weighted inner product and let $\|x\|_W = (x,x)_W^{1/2} = (x^T W x)^{1/2}$ denote the $W$-weighted norm.

3 Direct Projection Methods

Projection methods have been demonstrated to be efficient methods to compute solutions of Riccati and Lyapunov equations [40, 41, 42, 43, 61, 38, 63], and can be divided in two steps. The first step is the computation of the approximate solution of the ARE via a specific algorithm. The second step is a computation of a matrix residual norm (or other criteria) to test the accuracy of the approximate solution obtained in the first step. Below, we give an overview of the projection approach, discuss the choice of a projection matrix, and review residual norms and approximation errors.

3.1 Overview

For a symmetric positive definite weight matrix $W \in \mathbb{R}^{n \times n}$, assume we have a matrix $V_r$ with full column rank such that

$$V_r = [v_1, v_2, \ldots, v_r] \in \mathbb{R}^{n \times r}, \quad V_r^T W V_r = I_r, \tag{7}$$

Preprint
where $v_i \in \mathbb{R}^n$ for each $i$ and $r \ll n$ is the reduced order dimension. The matrix $V_r$ is called a projection matrix and is used to obtain a reduced order model $\Sigma_r = (E_r, A_r, B_r, C_r)$ of the system $\Sigma = (E, A, B, C)$ via

$$ E_r = V_r^T E V_r, \quad A_r = V_r^T A V_r, \quad B_r = V_r^T B, \quad C_r = C V_r. \quad (8) $$

The reduced order matrices give the following low order projected ARE

$$ A_r^T \Pi_r E_r + E_r^T \Pi_r A_r - E_r^T \Pi_r B_r B_r^T \Pi_r E_r + C_r^T C_r = 0 \in \mathbb{R}^{r \times r}, \quad (9) $$

which can alternatively be obtained by imposing a Galerkin condition on the residual matrix. Assuming the low order ARE is well posed, the solution $\Pi_r \in \mathbb{R}^{r \times r}$ can be computed using the well developed methods for moderate sized AREs; e.g., the direct solver care in Matlab. Having solved the low order ARE (9), one defines a low rank approximate solution to the large-scale ARE (1) as

$$ P_r := V_r \Pi_r V_r^T \approx P. \quad (10) $$

Projection methods automatically yield low rank factored solutions. As noted earlier, the solution to the low rank algebraic Riccati equation $\Pi_r$ is symmetric positive semidefinite. Thus, the eigenvalue decomposition $\Pi_r = U_r S_r U_r^T$ is used to define the low rank factor

$$ Z_r = V_r U_r S_r^{1/2}, \quad (11) $$

which in turn gives $P_r = Z_r Z_r^T$. In practice, only the low rank factors are stored and used where the solution $P_r$ would be needed.

Steps (7)-(11) are common to all Galerkin projection methods. The distinctive feature of a method is the structure and generation of the projection matrix $V_r$. As we see later, if the columns of $V_r$ are in the span of the solution to the Lyapunov equation (4) then the projection based method for solving an ARE can be very accurate. A discussion about measures of accuracy of approximate solutions to ARE is given in Section 3.3 below.

### 3.2 The Choice of the Projection Matrix

Below, we give a brief overview of available methods available for computing $V_r$, as they distinguish the various projection methods. In the next chapter, we propose a new POD based algorithm to compute the projection matrix.

#### 3.2.1 Regular Krylov Methods

Direct projection methods for solving a Lyapunov equation of the form (4) with $E = I$ were first considered in [58] and variations, including a GMRES method on the matrix residual can be found in [40]. Recall, that the explicit solution (6) with $E = I$ to the Lyapunov equation contains $e^{A^T} C^T$ in the integrand. The idea is that a good approximation of the integrand combined with a suitable quadrature rule should be sufficient for convergence of the Gramian. Therefore, the authors in [58] used the standard Krylov subspace

$$ \mathcal{K}_r(A^T, C^T) := \text{span}\{C^T, A^T C^T, \ldots, (A^T)^{r-1} C^T\} \quad (12) $$

to achieve $e^{A^T} C^T \approx p_{r-1}(A^T) C^T$, where $p_{r-1}(A^T)$ is a matrix polynomial of maximum degree $r - 1$. Note, that this does not necessarily imply that the matrix exponential is well approximated by the matrix polynomial, but only imposes a lower bound of the form

$$ ||e^{A^T} C^T - p_{r-1}(A^T) C^T||_2 \leq ||e^{A^T} - p_{r-1}(A^T)||_2 ||C^T||. $$
More sophisticated quadrature rules are considered in [36], where it was shown that under rather mild assumptions a quadrature with ‘sinc’ quadrature points and appropriate weights yields good low rank approximations of \( X \). It has been widely noted that the projection matrix \( V_r \) has to be of high rank for those methods to converge.

### 3.2.2 Extended Krylov Subspace Method

We next describe an improved Krylov subspace method which is widely used for solving large AREs and Lyapunov equations. Thus, consider the extended Krylov space

\[
\mathcal{K}_r(A^T, C^T) := \mathcal{K}_{r/2}(A^T, C^T) + \mathcal{K}_{r/2}(A^{-T}, C^T),
\]

with \( E = I \) and \( W = I \), and \( A^{-T} := (A^{-1})^T \). For notational convenience, we assume that \( r \) is an even integer. In [30], it was shown that the enriched Krylov subspaces yield more accurate approximations for \( p_{r-1}(A^T)C^T \) than the standard Krylov space \( \mathcal{K}_r(A^T, C^T) \) in (12). This result has been exploited in the design of an iterative method for the solution of the Lyapunov equation, see [61]. The proposed Extended Krylov Subspace Method (EKSM) was found to outperform other methods in terms of CPU-time and memory requirements. Moreover, the direct projection method using the extended Krylov subspace above has been used to solve Riccati equations in [38]. For certain examples, the projection based EKSM was found to be better in terms of CPU-time and memory requirement than the Cholesky factorized-ADI method. When the matrix \( E \) is not the identity matrix, there are two ways of adapting this Krylov subspace. First, one can work in changed variables through a Cholesky decomposition of the mass matrix, or explicitly solve linear systems with the mass matrix whenever \( A^T \) is applied to a vector, see [61, 17].

### 3.2.3 Gramian Based Projection

The relationship between the ARE (1) and the Lyapunov equation (4), motivates the following projection matrix. The solution of the Lyapunov equation can be computed with \texttt{lyap} in \texttt{Matlab} as

\[
X = \texttt{lyap}(A^T, C^T, [], E^T).
\]

\texttt{Matlab} uses the SLICOT SB03MD routine for this problem. At first, the Schur decomposition of \( A^T \) is computed and then the new system solved by forward substitution. The algorithm is backward stable and requires \( O(n^3) \) operations, therefore becoming unfeasible for large \( n \). We next compute the singular value decomposition of the observability Gramian

\[
X = V\Sigma W^T,
\]

where the columns of \( V = [v_1 v_2 \ldots v_n] \) span the range space of \( X \). Truncation of \( V \) after \( r \) columns yields the projection matrix as \( V_r = [v_1 v_2 \ldots v_r] \). Finally, the projected Riccati equation (9) is solved. We include this method for testing on small problems since it is closely related to the POD projection method proposed in Section 4.

### 3.3 Residual Computations and Approximation Errors

In order to set stopping criteria or to compare various methods regarding accuracy, it is necessary to define a quality measure for approximate solutions. A relative residual is often used as a stopping criteria to select the rank of the approximate solution. Many existing approaches to solving the ARE (1) are linked with specialized algorithms that rapidly compute (or estimate) the residual.
We do not attempt to review these algorithms here; see the survey paper [17] for some details and references. Below, we present a QR algorithm from [15] to compute the residual that is applicable for many low rank ARE solution methods. We use this approach for the computations in this paper. This residual approach is not as computationally cheap as the specialized algorithms mentioned above; however, it is still computationally tractable and scales to large problems.

Let \( P_r \in \mathbb{R}^{n \times n} \) be a symmetric positive semidefinite approximate solution of the ARE (1). The residual is defined as

\[
\mathcal{R}(P_r) := A^T P_r E + E^T P_r A - E^T P_r B B^T P_r E + C^T C.
\]  

(14)

We assume \( P_r \) has a low rank factorization \( P_r = Z_r Z_r^T \), where \( Z_r \in \mathbb{R}^{n \times r} \) and \( r \ll n \). Then the residual can be rewritten as

\[
\mathcal{R}(P_r) = F G F^T, \quad F = \begin{bmatrix} C^T & A^T Z_r & E^T Z_r \end{bmatrix}, \quad G = \begin{bmatrix} I & 0 & 0 \\ 0 & I & 0 \\ 0 & I & -(Z_r B) (Z_r B)^T \end{bmatrix}.
\]

We have \( F \in \mathbb{R}^{n \times (2 r + p)} \) and \( G \in \mathbb{R}^{(2 r + p) \times (2 r + p)} \), where \( C \in \mathbb{R}^{p \times n} \). Let \( F = QR \) be the thin QR decomposition of \( F \), see [35, page 230]. Since \( Q^T Q = I \), we have

\[
\|\mathcal{R}(P_r)\|_2 = \|R G R^T\|_2, \quad \|\mathcal{R}(P_r)\|_F = \|R G R^T\|_F.
\]

Therefore, computing the norm of the residual matrix \( \mathcal{R}(P_r) \) (e.g., of size \( n \geq 10^5 \)) can be reduced to the norm computation of a small square matrix (e.g., of size \( 2 r + p \approx O(10) \) or \( O(100) \)) using a thin QR decomposition.

Next, we review the connection between the residual and the approximation error. Let \( P_r \) be any approximation to the exact solution of the ARE (1). Ideally, one should use the actual approximate error

\[
\varepsilon = \|P - P_r\|,
\]  

(15)

to assess the accuracy of the approximate solution, where \( \| \cdot \| \) either denotes the matrix 2-norm or some other matrix norm. Of course, the exact solution is almost never available and, as mentioned above, the residual is often used instead. To connect the residual norm to the error in the solution, the authors in [46, 68] gave error bounds relating (15) to the residual when \( E = I \). We state these theorems below. First, define the linear operator \( \Omega_{P_r} : \mathbb{R}^{n \times n} \to \mathbb{R}^{n \times n} \) by

\[
\Omega_{P_r}(X) := (A - B B^T P_r)^T X + X (A - B B^T P_r).
\]

This operator has the structure of the “Lyapunov operator” in equation (4). Moreover, if \( (A - B B^T P_r) \) is stable, then Theorem 2.1 implies that \( \Omega_{P_r} \) is invertible (i.e. the Lyapunov equation has a unique solution), which is crucial for proving the next theorem.

**Theorem 3.1.** ([46, Theorem 2']) Let \( P_r \geq 0 \) be an approximation to the unique symmetric positive semidefinite solution \( P \) to the ARE (1). If \( (A - B B^T P_r) \) is stable and

\[
\|P - P_r\|_2 \leq \frac{1}{3 \|\Omega_{P_r}^{-1}\|_2 \|B\|_2^2} \quad \text{and} \quad 4 \|\Omega_{P_r}^{-1}\|_2 \|\mathcal{R}(P_r)\|_2 \|B\|_2^2 < 1,
\]

then

\[
\|P - P_r\|_2 \leq \frac{2 \|\Omega_{P_r}^{-1}\|_2 \|\mathcal{R}(P_r)\|_2}{1 + \sqrt{1 - 4 \|\Omega_{P_r}^{-1}\|_2 \|\mathcal{R}(P_r)\|_2 \|B\|_2^2}} \leq 2 \|\Omega_{P_r}^{-1}\|_2 \|\mathcal{R}(P_r)\|_2.
\]
Rewriting the above result yields
\[ \| P - P_r \|_2 \leq c(P_r) \| \mathcal{R}(P_r) \|_2, \]
where \( c(P_r) = 2\| \Omega_{P_r}^{-1} \|_2 \) depends on the approximate solution \( P_r \). Theoretically, this inequality bounds the error in the approximate solution by the matrix residual. The term \( c(P_r) \) can be viewed as a condition number of the algebraic Riccati equation. However, computing the quantity \( \| \Omega_{P_r}^{-1} \|_2 = \sup_{X \neq 0} (\| \Omega_{P_r}^{-1}(X) \|/\| X \|) \) is not straightforward and only computable upper bounds were stated in [46, Sec.3]. Furthermore, Sun [68] sharpened the above error bound while simultaneously relaxing the assumptions. A statement of those results is beyond the scope of this paper. For the extended Krylov approach discussed above, a similar error bound is given in [38, Theorem 3.2].

4 POD Projection Method

Proper Orthogonal Decomposition (POD) is a widely used model reduction method in the engineering and mathematics communities. Depending on the field, it is also known as Karhuenen-Loeve expansion or principal component analysis. POD has been used to create low order models of complex systems in order to speed up simulation, optimization, control, and many other applications; see, e.g., [39, 24, 49, 37, 25, 29, 72, 71] and the references therein.

We propose using POD in the projection framework to approximate solutions to AREs in an accurate and computationally efficient manner. First, a POD method is employed to approximate the dominant eigenvectors of the observability Gramian \( X \) solving the Lyapunov equation (4). Those vectors are used to construct a projection matrix \( V_r \) which is used in a projection framework to solve the ARE.

In [73], a snapshot based approach was used to approximate solutions of the Lypunov equation in \( n \) dimensions (see also [58, 56]). In particular, the authors suggested using snapshots of simulations of linear systems to compute the observability Gramian (6). In [66], this idea was extended to infinite dimensional Lyapunov equations and a rigorous convergence theory was presented. Specifically, error bounds and convergence of the low rank, finite dimensional solution to the infinite dimensional Gramian were obtained. We follow this approach and first compute an approximation of the observability Gramian \( X \) and then we project the ARE using the approximate dominant left singular vectors of \( X \). The dimension of the necessary singular value decomposition is limited by \( \min(n, ps) \), with \( s \) being the number of snapshots collected during the simulation of a linear system.

In control and systems theory, the dual equations of the underlying optimal control problem are
\[
\begin{align*}
E^T \hat{z}_i(t) &= A^T z_i(t), \\
E^T z_i(0) &= c_i^T,
\end{align*}
\]
for all \( i = 1, \ldots, p \), where \( C^T = [c_1^T, c_2^T, \ldots, c_p^T] \). By the theory of ordinary differential equations, the unique solution to (16)-(17) is given by \( z_i(t) = e^{tE^TA^T} E^{-T} c_i^T \), for \( i = 1, \ldots, p \). Due to the representation of the Lyapunov solution in Corollary 2.2, the authors in [73] thus suggested to use simulations of the dual equations to approximate the solution of the observability Gramian \( X \).
When $E^{-1}A$ is stable, the Gramian can be rewritten as follows:

$$X = \int_{0}^{\infty} e^{tE^{-T}A^T} E^{-T} C T C E^{-1} e^{tA E^{-1}} dt$$

$$= \int_{0}^{\infty} e^{tE^{-T}A^T} E^{-T} [c_1^T, \ldots, c_p^T] \begin{bmatrix} c_1 \\ \vdots \\ c_p \end{bmatrix} E^{-1} e^{tA E^{-1}} dt$$

$$= \int_{0}^{\infty} e^{tE^{-T}A^T} E^{-T} [c_1^T, \ldots, c_p^T] \left( e^{tE^{-T}A^T} E^{-T} [c_1^T, \ldots, c_p^T] \right)^T dt$$

$$= \int_{0}^{\infty} [z_1(t), \ldots, z_p(t)] [z_1(t), \ldots, z_p(t)]^T dt$$

$$= \int_{0}^{\infty} [Z(t)] [Z(t)]^T dt.$$ 

We approximate the observability Gramian by the finite time integral

$$X \approx X_T := \int_{0}^{T} [Z(t)] [Z(t)]^T dt,$$

where $T$ specifies a final time, chosen so that a good approximation of the infinite integral is obtained. The finite time integral can be approximated by quadrature, such that

$$X \approx X_{T,\delta} := \sum_{i=1}^{s} \delta_i [Z(t_i)] [Z(t_i)]^T, \quad (18)$$

using positive weights $\delta_i$ and a time grid $0 = t_1 < t_2 < \ldots < t_s = T$. Here, let $s_i = t_{i+1} - t_i$ be the step size for the numerical integration scheme at the $i^{th}$ time step. In matrix form, the approximation reads as

$$X \approx X_{T,\delta} := Z \Delta Z^T,$$

where $\Delta$ is the diagonal matrix of weights and $Z$ contains snapshots of simulations, as outlined below. The method of snapshots [67] is used for the POD computations, as briefly reviewed below. We refer the reader to, e.g., [72, Chapters 2–3] and the references above for more detail. We note that the POD computations can be performed using other approaches; see, e.g., [32, 11].

Some technical details for the implementation of the POD snapshot based approach to approximate the Gramian are listed below.

- For accurate simulation of the dual system (16)-(17), a proper set of time steps has to be chosen a priori, or adaptively during the time stepping. In this work, we simulated the test problems with Matlab's adaptive time stepping solvers ode45 and ode23s, with default absolute and relative error tolerances. In most cases, the snapshots are selected at the time steps chosen by the adaptive solver.

- In case of highly stiff problems, the time steps $s_i$ are small, which results in a larger set of snapshots than is needed for computation of $V_r$. In this case, a subset of snapshots from the previous step is selected, and the singular value decomposition computed from this smaller set of vectors. In practice, we found this approach to not lose significant accuracy, compared to keeping all time snapshots.
The final time $T$ is the only parameter that needs to be fixed for the POD based approach. One possible approach is to choose $T$ so that the norms of each solution $z_i(t)$ are below a certain tolerance. (Since the system is stable, the solutions must tend to zero for large enough time.) For certain simulation codes, it is possible to choose the tolerance and the simulation will determine the time $T$. Also, we note that different final times can be taken for each simulation, but for simplicity we use one final time $T$.

In this work, the weights for the approximation of the integral are chosen by the trapezoidal rule, which yielded high accuracy in the projection framework as demonstrated in Section 5. The remainder of this section focuses on the construction of the projection matrix $V_r$ so that it is orthogonal with respect to a $W$-weighted inner product, i.e., $V_r^T W V_r = I$, where $W$ is a symmetric positive definite matrix. For a given initial condition $E^{-T} c_1^T$, simulate system (16)-(17) and assemble the time snapshots in a matrix

$$Z_1 = [z_1(t_1) \ldots z_1(T)] \in \mathbb{R}^{n \times s}.$$  

Simulations starting with every column of $E^{-T} C^T$ are concatenated in the matrix

$$Z = [Z_1, Z_2, \ldots, Z_p] \in \mathbb{R}^{n \times ps}.$$  

Further, the approximate observability Gramian in the new variables can be factored as

$$X_{T,\delta} = Y Y^T, \quad Y = Z \Delta^{1/2} \in \mathbb{R}^{n \times ps}.$$  

The projection method only requires the $W$-orthogonal eigenvectors of $X_{T,\delta}$ to construct $V_r$, so there is no need to form the approximate Gramian explicitly, and we can work with the factor $Y$ instead.

For large systems in which the state space dimension exceeds the number of snapshots, the well known method of snapshots to compute the eigenvectors proceeds as follows. First, compute the eigenvalue decomposition

$$Y^T W Y = \Phi \Lambda \Phi^T \in \mathbb{R}^{ps \times ps},$$  

and rescale (if necessary) the eigenvectors so that they are orthonormal with respect to the standard inner product, i.e., $\Phi^T \Phi = I$. Here, $\Lambda = \text{diag}(\lambda_1, \lambda_2, \ldots)$, and the eigenvalues are ordered so that $\lambda_1 \geq \lambda_2 \geq \cdots \geq 0$. The projection matrix $V_r$ is given by the matrix consisting of the first $r$ eigenvectors of $X_{T,\delta}$, which is given by

$$V_r = Y \Phi_r \Lambda_r^{-1/2},$$  

where $\Phi_r \in \mathbb{R}^{n \times r}$ denotes the matrix consisting of the first $r$ columns of $\Phi$, and $\Lambda_r \in \mathbb{R}^{r \times r}$ denotes the matrix $\Lambda_r = \text{diag}(\lambda_1, \ldots, \lambda_r)$. The procedure is summarized in pseudocode in Algorithm 1. Note that the loop computation can be executed in parallel.

In exact arithmetic, the projection matrix $V_r$ is orthonormal with respect to the $W$-weighted inner product whenever $\lambda_r > 0$. This can be seen by direct computation: since $\Phi_r^T \Phi_r = I$, we have

$$V_r^T W V_r = \Lambda_r^{-1/2} \Phi_r^T (Y^T W Y) \Phi_r \Lambda_r^{-1/2} = \Lambda_r^{-1/2} \Phi_r^T (\Phi_r \Lambda_r \Phi_r^T) \Phi_r \Lambda_r^{-1/2} = I.$$  

However, as is well known, in finite precision arithmetic accuracy can be lost due to forming the required matrix products. In the large-scale problem considered in Section 5 below, we found that $V_r^T W V_r$ began to deviate significantly from the identity matrix as $r$ increased.

To deal with the loss of $W$-orthogonality, we use a $W$-weighted stabilized Gram-Schmidt procedure to form a new $W$-orthonormal matrix $V_r$ with the same span as the $V_r$ matrix constructed above. To be complete, we present a simple implementation in Algorithm 2 below. We include a reorthogonalization step as this is well known to give greater accuracy in finite precision arithmetic; see, e.g., [34, 57] and the references therein.
Algorithm 1: POD method to compute projection matrix

Input: $E, A, W \in \mathbb{R}^{n \times n}$, $B \in \mathbb{R}^{n \times m}$, $C \in \mathbb{R}^{p \times n}$, final time $T$, maximal order $r_{\text{max}}$.
Output: Projection matrix $V_{r_{\text{max}}}$.
1: for $i = 1, 2, \ldots, p$ do
2: Simulate $E^T \dot{z}_i = A^T z_i$, $E^T z_i(0) = c_i^T$ and place time snapshots of solutions in the matrices $Z_i = [z_i(t_1^i), z_i(t_2^i), \ldots, z_i(t_s^i) = T]$.
3: Compute quadrature weights (see (18)), and let $\Delta_i = \text{diag}(\delta_1^i, \delta_2^i, \ldots, \delta_s^i)$.
4: end for
5: $Y = [Z_1, \ldots, Z_p] \cdot \text{diag}(\Delta_1, \ldots, \Delta_p)^{1/2}$.
6: Compute the singular value decomposition $Y^TWY = \Phi \Lambda \Phi^T$.
7: $V_{r_{\text{max}}} = Y \Phi_1 : r_{\text{max}} \Lambda_1^{-1/2}$.

Algorithm 2: $W$-Weighted Stabilized Gram-Schmidt with Reorthogonalization

Input: $V_r = [v_1, v_2, \ldots, v_r] \in \mathbb{R}^{n \times r}$, $W \in \mathbb{R}^{n \times n}$.
Output: $V_r^{(\ell)} \in \mathbb{R}^{n \times r}$ such that $\text{span}(V_r)$ remains unchanged and $V_r^{(\ell)}WV_r^{(\ell)} = I \in \mathbb{R}^{r \times r}$.
1: for $\ell = 1, 2$ (reorthogonalization) do
2: for $i = 1, \ldots, r$ do
3: $v_i = v_i/(v_i^TWv_i)^{1/2}$.
4: for $j = i + 1, \ldots, r$ do
5: $v_j = v_j - (v_j^TWv_i)v_i$.
6: end for
7: end for
8: end for

4.1 Parallels to Standard and Rational Krylov Subspaces

Rational Krylov subspace techniques (and the related ADI iteration) have been demonstrated to be efficient for solving Lyapunov equations; see [31] and the recent survey [62]. Moreover, in [63] rational Krylov subspace methods were compared with the EKSM as direct projection methods for the ARE and it was found that both methods show a similar convergence behavior. In this short section, we highlight some of the similarities and differences between the projection spaces obtained from the POD method and rational Krylov subspace methods. A thorough analysis is left for future work.

Recall, the integral representation of the solution of the Lyapunov equation in Corollary 2.2 motivated us to consider approximating the integral via quadrature using snapshots of the dual projection spaces. It was found that both methods show a similar convergence behavior. In this short section, we highlight some of the similarities and differences between the projection spaces obtained from the POD method and rational Krylov subspace methods. A thorough analysis is left for future work.

Preprint 11

B. Kramer and J. Singler

POD Projection for Large-Scale Riccati Equations
of dimension $n_s$. Note that $\text{span}(S_{BE}) = \text{span}(\mathcal{K}_{n_s}(E^{-T}A^T, x_0, s^{-1}))$ is a rational Krylov subspace, and $s^{-1} = [s_1^{-1}, s_2^{-1}, \ldots, s_{n_s}^{-1}]$ is the vector of inverted time steps.

If we arrange the columns of $S_{BE}$ in a matrix $X$ and take its singular value decomposition, then the left singular vectors form an orthonormal basis for the range of $X$. The POD approach only uses the $r$ dominant left singular vectors to generate $V_r$, and therefore the span of $V_r$ would only approximate the span of $S_{BE}$. Alternatively, one can perform a block Arnoldi algorithm on the columns of $S_{BE}$ to obtain $V_r$. Thus, for the same set of parameters $s$, both the POD based approach as well as the rational Krylov subspace method yield similar results. In other words, the POD approach with a stable backward Euler scheme approximates a rational Krylov subspace.

However, there are significant differences between the two approaches in practice:

- One would likely use time stepping schemes with higher accuracy than backward Euler in the POD approach. Certain time stepping schemes might possibly lead to snapshot sets $S_{BE}$ of a type not usually considered in standard rational Krylov subspace approaches.

- The POD approach generates the large “Krylov subspace” $S_{BE}$ first and then builds $V_r$ by extracting the dominant singular vectors of the matrix with the same columns as $S_{BE}$. In contrast, a rational Krylov approach increases the rank of the Krylov subspace and $V_r$ iteratively; a residual is used to stop updating the construction.

The last item shows that, in general, the POD approach requires more computational cost and storage. However, adaptive time stepping algorithms can often achieve high accuracy with a low number of snapshots. In Examples 2 and 3 in Section 5 the POD approach with $n_s \approx c \cdot r$, where $c \lesssim 5$, gave high accuracy. For instance, in Example 3, $n_s = 130$ and at order $r = 40$ the residual is approximately $10^{-13}$.

It is interesting to note that with a forward Euler scheme we obtain another rational Krylov subspace, this time with a different structure. In fact, a forward Euler approximation yields the recurrence

$$x_{k+1} = (I + s_i E^{-T}A^T)x_k,$$

so that the collection of snapshots is

$$S_{FE} = \left\{ x_0, (I + s_1 E^{-T}A^T)x_0, \ldots, \prod_{i=1}^{n_s-1} (I + s_i E^{-T}A^T)x_0 \right\}.$$

One can easily see that

$$\text{span}(S_{FE}) = \text{span} \{ x_0, (E^{-T}A^T)x_0, \ldots, (E^{-T}A^T)^{n_s-1}x_0 \},$$

which is the span of the standard Krylov subspace $\mathcal{K}_{n_s}(E^{-T}A^T, x_0)$.

5 Numerical Results

In this section, we present numerical results for three different test problems. The first problem is a benchmark problem from structural dynamics, whereas the second and third problems arise from spatial discretizations of heat transfer and fluid dynamical problems. The second and third problems have symmetric positive definite mass matrices $E$.

Table 1 contains specifics and parameters of the test models. The quantity $\lambda_{\max}(A, E)$ denotes the (approximate) largest generalized eigenvalue of $A$. Also, $\omega(A, E)$ denotes the (approximate) generalized numerical abscissa of $A$, which is the largest generalized eigenvalue of $(1/2)(A + A^T)$. 

Preprint 12
These quantities were computed in Matlab using the generalized eigenvalue problems $Ax = \lambda Ex$ and $(A + A^T)x = \lambda Ex$, respectively.

| Problem    | $n$  | $m$  | $p$  | $\lambda_{max}(A,E)$ | $\omega(A,E)$ | $A = A^T$? | $E = I$? |
|------------|------|------|------|-----------------------|----------------|-------------|----------|
| space station | 270  | 3    | 3    | $-4.8 \cdot 10^{-2}$  | $7.65 \cdot 10^{-9}$ | no          | yes      |
| diffusion   | 9900 | 1    | 1    | $-0.62$               | $-0.6150$      | yes         | no       |
| conv.-diff. | 122150 | 1    | 1    | $-0.16$               | $0.7525$       | no          | no       |

All of the examples are stable, however the values of the numerical abscissa indicate that the uncontrolled problems are quite different from each other. The numerical abscissa indicates whether solutions of the system (2)-(3) with no control (i.e., $u(t) = 0$) can experience transient growth before decaying to zero; this transient growth is possible only if the numerical abscissa is positive; e.g., [69, Sections 14 and 17]. Also see [7] for more about the numerical abscissa and related matrix Lyapunov equations.

Recall, the first step for solving ARE with projection methods consists of generating a projection matrix $V_r \in \mathbb{R}^{n \times r}$. Next, the reduced order matrices $(E_r, A_r, B_r, C_r)$ are computed via projection and the Riccati equation (9) is solved in $r$ dimensions with care in Matlab. This direct solution routine solves the Hamiltonian eigenvalue problem associated with ARE and is further described in [4]. The low rank approximation of $P$ is then given by $P_r = Z_r Z_r^T$. A general projection algorithm following the steps in Section 3 is used to compare the methods, see Algorithm 3. In all cases, following [61, 38] and the remark in [17, p.9], $tol = 10^{-12}$ was used for truncation of the singular values of the low rank solution $P_r$. Therefore, a rank reduction is already applied at the reduced order level, which is in accordance with the low rank structure of the Riccati solution.

**Remark 5.1.** We also briefly tested setting $tol = 0$ in two versions of Example 3 below with smaller values of $n$ (generated using coarser finite element meshes). Of course, there is no further rank reduction in this case. In these tests, the matrix residual for the POD approach (not shown here) levels off at order $10^{-15}$ instead of order $10^{-13}$ when we use the above tolerance. However, this increases the rank of $P_r$.

**Algorithm 3 :** Projection based Riccati solver with residual computation

**Input:** $E, A \in \mathbb{R}^{n \times n}$, $B \in \mathbb{R}^{n \times m}$, $C \in \mathbb{R}^{p \times n}$, $V_r \in \mathbb{R}^{n \times r}$, a tolerance $tol > 0$.

**Output:** The low rank factor $Z_l$ of $P_l = Z_l Z_l^T$, and residual norm vector $[\gamma_1, \ldots, \gamma_r]$.

1. for $l = 1, 2, \ldots, r$ do
2. Let $V_l = V_l(:, 1 : l)$. Compute $E_l = V_l^T E V_l$, $A_l = V_l^T A V_l$, $B_l = V_l^T B$, $C_l = C V_l$.
3. Solve $A_l^T \Pi_l E_l + E_l^T \Pi_l A_l - E_l^T \Pi_l B_l B_l^T \Pi_l E_l + C_l^T C_l = 0$.
4. Compute (svd or evd): $\Pi_l = U \Sigma U^T$, where $\Sigma = \text{diag}[\sigma_1, \ldots, \sigma_l]$ and $\sigma_1 \geq \ldots \geq \sigma_l$.
5. Determine $k$ such that $\sigma_{k+1} < tol < \sigma_k$; set $\Sigma_k = \text{diag}[\sigma_1, \ldots, \sigma_k]$, $U_k = U(:, 1:k)$ and compute $S_l = U_k S_k^{1/2}$.
6. $Z_l = V_l S_l$, i.e., the low rank factor of $P_l = Z_l Z_l^T$.
7. Compute $\gamma_l = \|\Re(P_l)\|$ as in Subsection 3.3.
8. end for

Since the first problem is small, for comparison purposes we compute the solution $P$ via a direct solver (care) and compute the actual error in the solution, $\|P - P_r\|$, for increasing $r$. Moreover,
we examine the convergence of the feedback gains using the relative error $\|K - K_r\|/\|K\|$, where $K = B^TPE$ and $K_r = B^T P_r E$. For large problems, we consider the convergence behavior of the residual norm $\|R(P_r)\|$ and the relative change in the feedback gains $\|K_r - K_{r-1}\|/\|K_r\|$. Note that the full matrix $P_r$ is never explicitly used or stored, and we only work with the low rank factor $Z_r$ from equation (11). In particular, the residual computation is performed as in Subsection 3.3 and the feedback gain matrices are evaluated as $K_r = B^T P_r E = (B^T Z_r)(Z^T_r E)$.

The first two problems were computed on a 2010 MacBook Pro with a 2.66 GHz Intel Core i7 Processor and 4GB RAM. Matlab was used as a software in the version of R2012b. The convection diffusion problem was solved on a computer cluster with two 6-core Intel Xeon X5680 CPU’s at 3.33GHz. The cluster has a Random-Access Memory of 48GB and runs on Scientific Linux 6.4 with Matlab in the Version of 2013b. Machine precision on both machines is in accordance with IEEE double precision standard, $\text{eps} = 2.210^{-16}$.

### 5.1 Example 1: ISS1R Flex Model

This model describes a structural subsystem of the International Space Station (ISS) and is taken from [3]. Many flexible structures, operational modes, and control systems result in a complex dynamical system. Solving the algebraic Riccati equation for this model is important for model reduction, such as LQG balanced truncation, as well as optimal control design for the overall plant.

The system matrix $A$ is a nonsymmetric dense matrix with 63843 nonzero elements and has 88% fill-in. The mass matrix is the identity, $E = I$. The condition number of $A$ is $1.0 \cdot 10^2$. For $r = 2, 4, \ldots, 60$, we computed low rank Riccati solutions with each of the described methods and plotted the norm of the matrix residual in Figure 1. For the POD based projection method, the dual system was simulated from $t = 0$ to $T = 15s$ and snapshots were taken every 0.02s. This amounts to a combined 2253 snapshots of the three simulations of the system. The residual norm for the solution $P$ produced by the direct care solver is $1.8 \cdot 10^{-12}$. The reader should observe that not even the direct solver achieves machine precision for the residual norm.

Figure 1, left, shows the residual norm $\|R(P_r)\|_F$ and the actual error in the solution $\|P - P_r\|_F$ for the EKSM, POD based method, and the Gramian based approach. The approach based on computing the Gramian with a direct solver and then projecting performs very well in terms of accuracy. As mentioned earlier, this approach is only feasible for small scale problems. The POD based method shows a rather monotone convergence behavior of the residual norm, yet it is several orders of magnitude larger than the Gramian based approach. The extended Krylov subspace method shows oscillations in the residual norm and has the largest residual norms of the three methods. A monotone decreasing error is desirable, for it guarantees that extra work for computing an increased size reduced order model is not wasted. The reader should observe that in this example the residual is a good indicator of the accuracy of solutions. In general, this does not have to be true; see the discussion in [74, Section 5].

Figure 1, right, shows the convergence of the feedback gain matrices. The gains show a similar convergence behavior compared to the residual norm and solution error, since $K_r = B^T P_r E$. Note that the convergence in both plots is slow for the POD and EKSM method, which indicates the numerical rank of $P$ is rather large. In fact, the rank of $P$ is 267 (computed with rank in Matlab) and the decay of the singular values for determination of the numerical rank is plotted in Figure 3 and commented on below.

Recall that the POD method approximates the observability Gramian by a numerical quadrature. The number and location of the time samples $t_j$ and weights $\delta_j$ is important for the quadrature to be accurate. Increasing both the final time $T$ when terminating simulations and the number of snapshots should yield a more accurate approximation of the Gramian. Figure 2, left, plots the...
As we have seen above, the Gramian approach with the \texttt{lyap} solver as described in Section 3.2.3 performs very well. In other words, the left singular vectors of the observability Gramian provide a rich subspace for solving ARE with a projection method. To further investigate this, Figure 3, left, shows a plot of the singular values of the Lyapunov and Riccati solution, obtained using \texttt{lyap} and \texttt{care}. The singular values of both solutions are close to each other and only separate at approximately $10^{-10}$. This could explain the excellent behavior of the Gramian based approach.

Figure 3 plots the quotient of actual error in the approximation, $\|P - P_r\|_F$, and residual, $\|\mathcal{R}(P_r)\|_F$. From theoretical results in Section 3.3, it is known that $\|P - P_r\|_F \leq c \cdot \|\mathcal{R}(P_r)\|_F$, where $c = \|\Omega_P^{-1}\|_F$ depends on the current solution. From a theoretical perspective, it is sufficient for $c$
to be bounded from above. However, when the residual is used to judge about the accuracy of a computation, the magnitude of the constant does play a crucial role. For instance, consider the constant \( c = 1000 \). Clearly, the actual error could be up to 1000 times higher than the residual would suggest. Therefore, we view the residual as an error “indicator” and are aware of the fact that the actual error can be nominally higher than the residual norm. It is demonstrated in [74, Section 5] that residuals can sometimes fail to be a good indicator of the accuracy of solutions for Lyapunov equations.

Figure 3: ISS1R model. (left): Normed singular values of the observability Gramian \( X \) and Riccati solution \( P \) using direct solvers \texttt{lyap} and \texttt{care}. (right): The relation between residual and actual approximation error, \( \|P - P^r\|_F / \|R(P^r)\|_F \).

### 5.2 A diffusion problem in 2D

We consider the diffusion problem in 2D:

\[
\frac{\partial w}{\partial t} = \mu (w_{xx} + w_{yy}) + b(x, y)u(t)
\]

on \( \Omega = [0, 1] \times [0, 1] \) with Dirichlet boundary conditions on the bottom, right and top walls:

\[
w(t, x, 0) = 0, \quad w(t, 1, y) = 0, \quad w(t, x, 1) = 0,
\]

and Neumann boundary condition on the left wall:

\[
w_x(t, 0, y) = 0.
\]

We choose \( b(x, y) = 5 \) if \( x \geq 1/2 \) and \( b(x, y) = 0 \) otherwise. The outputs are taken to be

\[
\eta(t) = \int_\Omega 5w(t, x, y)dxdy.
\]

After semi-discretization by a standard piecewise bilinear finite element method, a finite dimensional system with representation \((E, A, B, C)\) is obtained, where \( n = 9900, \quad m = p = 1 \). In this example, the system matrix \( A \) is symmetric and has a condition number of \( 6.17 \cdot 10^3 \). The dual system was simulated for 15s using \texttt{ode23s} in \texttt{Matlab} with default error tolerances and options \texttt{mass} and \texttt{jacobian} set as \( E \) and \( A^T \), respectively. The norm of the final snapshot was \( 3.8 \cdot 10^{-4} \). The adaptive time stepping routine returned \( n_s = 104 \) snapshots of solutions.

As mentioned earlier, the \( E \)-weighted norm of a vector equals the \( L^2 \) norm of the corresponding finite element function. The \( L^2 \) norm of a function is a natural measure of magnitude for such a
problem; therefore, we use the $E$-weighted inner product and norm for this example. Then, the projection matrix $V_r$ (approximately) satisfies $V_r^T E V_r = I$, and so we simply use an identity matrix in place of the projected matrix $E_r = V_r^T E V_r$.

Figure 4, left, shows the residual norm of solutions to ARE, using the EKSM and POD based method, respectively. The residual norm for the POD method decreases quickly and then levels off at $10^{-13}$ when $r = 35$. As a point of comparison, at this projection order the residual norm for the extended Krylov space method is at the order of $10^{-9}$. Figure 4, right, shows the relative refinement of the feedback gains with increasing projection order. A similar convergence history as previously observed before can be seen. Note, that the feedback gain vectors continue to refine past $r = 35$, although the residual matrix stagnates.

Figure 4: Diffusion model. (left): Residual norm $\|R(P_r)\|_F$ as the rank of $P_r$ increases. (right): Convergence of the feedback gain vectors for increasing $r$.

5.3 A convection diffusion problem in 2D

With this example, we consider a convection diffusion equation, which is a common partial differential equation model arising in fluid dynamics and a variety of other application areas. The problem is given by

$$w_t = \mu(w_{xx} + w_{yy}) - c_1(x, y)w_x - c_2(x, y)w_y + b(x, y)u(t),$$

where $c_1(x, y) = x \sin(2\pi x) \sin(\pi y)$ and $c_2(x, y) = y \sin(\pi x) \sin(2\pi y)$. The boundary conditions, the function $b(x, y)$, and the output are chosen as in the previous example. This time, the semi-discretization by a standard piecewise bilinear finite element method yields a non-symmetric $A$ matrix with a condition number of $1.0 \cdot 10^6$. The state dimension is $n = 122150$, and $m = p = 1$. The dual system was simulated using ode23s in Matlab with default error tolerances and options mass and jacobian set as $E$ and $A^T$, respectively.

Since the state space dimension is large for this example, the Gramian approach as well as the computation of the Riccati solution with direct methods are not feasible on a standard desktop computer. In fact, it was not even feasible on our computer cluster with specifications given above. Note that for this model, the largest real part of the eigenvalues of $A$ is approximately $-1.6 \cdot 10^{-6}$, so simulations were performed from $t = 0$ to $T = 50s$. The generalized eigenvalues are listed in Table 1. The norm of the final snapshot was $8.8 \cdot 10^{-3}$. The adaptive ODE solver returned $n_s = 130$ snapshots of the solution. The mass matrix $E$ is well conditioned, $\text{cond}(E) \approx 14.1$, however the same is false for the matrix $A$ since $\text{cond}(A) \approx 10^6$. 

Preprint 17
Figure 5, left, shows the residual norm of the ARE solutions obtained from both the POD based method and the extended Krylov subspace method. The residual norm for the POD method again decreases quickly and then levels out at order $10^{-13}$ when the projection order is $r = 40$. Again, to give a point of comparison, at this projection order the residual norm for EKSM is at the order of $10^{-7}$. Figure 5, right, shows the relative change in the gain vectors computed for every additional two basis vectors, i.e., $\|K_r - K_{r-2}\|/\|K_{r-2}\|$. When using $r = 50$ modes for the POD projection method, the relative change in the gains is approximately $10^{-13}$. Again, the relative change in the gains shows similar trends to the matrix residual convergence.

![Figure 5](https://example.com/fig5.png)

Figure 5: Convection diffusion model. (left): Residual norm $\|R(P_r)\|_F$ as the rank of $P_r$ increases. (right): Convergence of the feedback gain vectors for increasing $r$.

6 Conclusion

We presented a new POD based projection approach to compute solutions of algebraic Riccati equations. The method relies on proper orthogonal decomposition to compute an approximation of the solution to the related Lyapunov equation via the algorithm in [73, 66]. The resulting dominant left singular vectors are used in the projection framework to solve algebraic Riccati equations. Numerical results demonstrate that this POD basis is sufficiently rich for the projection approach to produce accurate solutions at low solution rank. Currently, no convergence theory is available for this method and this will be part of future work, since the numerical results are very promising.

To put our numerical results in perspective to state-of-the-art low rank ARE solution methods, we compared this POD projection method to the extended Krylov subspace method [38]. The POD approach will generally require more computational effort and storage compared to EKSM, ADI and other similar approaches. It was demonstrated that the POD projection approach can be efficiently computed and can give high accuracy at a low solution rank. Again, we emphasize that our primary goal is not to develop the most efficient solver, but to move toward an efficient, highly accurate, completely data-based algorithm. We also note that the proposed approach may be naturally implemented by many researchers in a variety of fields who are already familiar with POD computations.

In our numerical experiments, we used an adaptive ODE solver in Matlab to approximate the solutions of the required ordinary differential equations. Adaptive time stepping methods can be found in other existing simulation packages, such as IFISS [60]. The POD approach will often still be tractable for large-scale systems even if an adaptive ODE solver is not used; in the literature,
POD computations are frequently performed using ODE solvers with a constant time step. The computational effort and required storage will usually be larger when the time steps are not chosen adaptively.

POD-based approaches can often be directly applied to systems governed by partial differential equations. For such problems, one can use existing simulation codes without accessing the matrix approximations of the operators; see, e.g., the discussion in [66]. A rigorous convergence theory for POD-based approximations to infinite dimensional Lyapunov solutions is available; see [66]. We believe the ideas in [66] coupled with the present approach will enable direct approximation of operator Riccati equations arising from partial differential equation systems. It may be possible to extend the existing theory to obtain rigorous error estimates for Riccati solutions.

It may also be possible to exploit the parallels between the POD method and rational Krylov subspace techniques (as discussed in Section 4.1) to obtain a thorough analysis of the method.

As mentioned in the introduction, for certain applications it is of interest to make the proposed approach completely data-based so that access to system matrices is not required. This can sometimes be done directly for AREs arising from parabolic partial differential equations (such as the second and third example problems considered in Section 5). Briefly, assume the simulation code for the parabolic PDE is based on a bilinear form as in finite element methods and other Galerkin methods. Then one only needs to be able to use the existing simulation code to evaluate the bilinear form acting on the relevant POD modes to project the $A$ operator (see [64, Section 3.1]). Also, it may be possible to modify the techniques from [64] to make the proposed algorithm completely data-based when the bilinear form is not available or the ARE does not arise from a parabolic PDE. We intend to explore these issues in more detail elsewhere.

Acknowledgements

B. Kramer would like to thank Professors John A. Burns and Daniel B. Szyld for helpful discussions on this topic. J. R. Singler acknowledges the support in part by the National Science Foundation under grant DMS-1217122.

References

[1] I. Akhtar, J. Borggaard, M. Stoyanov, and L. Zietsman. On commutation of reduction and control: linear feedback control of a von Kármán street. pages 1–14, 2010. 1

[2] L. Amodei and J.-M. Buchot. A stabilization algorithm of the Navier–Stokes equations based on algebraic Bernoulli equation. *Numerical Linear Algebra with Applications*, 19(4):700–727, 2012. 2

[3] A. C. Antoulas. *Approximation of Large-Scale Dynamical Systems*. Advances in Design and Control. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2005. 1, 2.1, 5.1

[4] W. F. Arnold and A. J. Laub. Generalized eigenproblem algorithms and software for algebraic Riccati equations. *Proceedings of the IEEE*, 72(12):1746–1754, 1984. 5

[5] J. A. Atwell, J.T. Borggaard, and B.B. King. Reduced order controllers for Burgers’ equation with a nonlinear observer. *International Journal of Applied Mathematics and Computer Science*, 11:1311–1330, 2001. 1
[6] J. A. Atwell and B. B. King. Proper orthogonal decomposition for reduced basis feedback controllers for parabolic equations. *Math. Comput. Modelling*, 33(1-3):1–19, 2001. 1

[7] J. Baker, M. Embree, and J. Sabino. Fast singular value decay for Lyapunov solutions with nonnormal coefficients. arXiv:1410.8741, 2015. 5

[8] H. T. Banks, S. C. Beeler, G. M. Kepler, and H. T. Tran. Reduced order modeling and control of thin film growth in an HPCVD reactor. *SIAM J. Appl. Math.*, 62(4):1251–1280, 2002. 1

[9] H. T. Banks, Ricardo C. H. del Rosario, and Ralph C. Smith. Reduced-order model feedback control design: numerical implementation in a thin shell model. *IEEE Trans. Automat. Control*, 45(7):1312–1324, 2000. 1

[10] H. T. Banks and K. Ito. A numerical algorithm for optimal feedback gains in high-dimensional linear quadratic regulator problems. *SIAM J. Control Optim.*, 29(3):499–515, 1991. 2

[11] C. A. Beattie, J. Borggaard, S. Gugercin, and T. Iliescu. A domain decomposition approach to POD. In *Proceedings of the 45th IEEE Conference on Decision and Control*, 2006. 4

[12] P. Benner. Accelerating Newton’s method for discrete-time algebraic Riccati equations. In *Proceedings MTNS 98*, pages 569–572, 1998. 1

[13] P. Benner. Balancing-related model reduction for parabolic control systems. In *1st IFAC Workshop on Control of Systems Governed by Partial Differential Equations*, pages 257–262. IFAC-PapersOnLine, 2014. 1

[14] P. Benner and Z. Bujanović. On the solution of large-scale algebraic Riccati equations by using low-dimensional invariant subspaces. Preprint MPIMD/14-15, Max Planck Institute Magdeburg, August 2014. 1

[15] P. Benner, J.-R. Li, and T. Penzl. Numerical solution of large-scale Lyapunov equations, Riccati equations, and linear-quadratic optimal control problems. *Numer. Linear Algebra Appl.*, 15(9):755–777, 2008. 3.3

[16] P. Benner and J. Saak. A Galerkin-Newton-ADI method for solving large-scale algebraic Riccati equations. Preprint SPP1253-090, DFG Priority Programme 1253 Optimization with Partial Differential Equations, 2010. 1

[17] P. Benner and J. Saak. Numerical solution of large and sparse continuous time algebraic matrix Riccati and Lyapunov equations: a state of the art survey. *GAMM-Mitteilungen*, 36(1):32–52, 2013. 1, 3.2.2, 3.3, 5

[18] D. Bini, B. Iannazzo, and B. Meini. *Numerical Solution of Algebraic Riccati Equations*. Fundamentals of algorithms. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, USA, 2012. 1

[19] J. Borggaard and M. Stoyanov. An efficient long-time integrator for Chandrasekhar equations. In *Proceedings of the 47th IEEE Conference on Decision and Control*, pages 3983–3988, 2008. 1

[20] J. A. Burns and K. P. Hulsing. Numerical methods for approximating functional gains in LQR boundary control problems. *Math. Comput. Modelling*, 33(1-3):89–100, 2001. Computation and Control, VI (Bozeman, MT, 1998). 1
[21] J. A. Burns, E. W. Sachs, and L. Zietsman. Mesh independence of Kleinman-Newton iterations for Riccati equations in Hilbert space. *SIAM J. Control Optim.*, 47(5):2663–2692, 2008. 1

[22] J.A. Burns, K. Ito, and R.K. Powers. Chandrasekhar equations and computational algorithms for distributed parameter systems. In *Proceedings of the 23rd IEEE Conference on Decision and Control*, volume 23, pages 262–267, 1984. 1

[23] John A. Burns. *Introduction to the Calculus of Variations and Control—With Modern Applications*. CRC Press, Boca Raton, FL, 2014. 1

[24] D. H. Chambers, R. J. Adrian, P. Moin, D. S. Stewart, and H. J. Sung. Karhunen-Loève expansion of Burgers model of turbulence. *Phys. Fluids*, 31:2573–2582, 1988. 4

[25] S. Chaturantabut and D. C. Sorensen. Discrete empirical interpolation for nonlinear model reduction. In *Proceedings of the 48th IEEE Conference on Decision and Control*, pages 4316–4321. IEEE, 2009. 4

[26] K. K. Chen and C. W. Rowley. Fluid flow control applications of $H_2$ optimal actuator and sensor placement. In *Proceedings of the American Control Conference*, pages 4044–4049, 2014. 1

[27] K.K. Chen and C. W. Rowley. $H_2$ optimal actuator and sensor placement in the linearised complex Ginzburg-Landau system. *J. Fluid Mech.*, 681:241–260, 2011. 1

[28] N. Darivandi, K. Morris, and A. Khajepour. An algorithm for LQ optimal actuator location. *Smart Materials and Structures*, 22(3):035001, 2013. 1

[29] B. T. Dickinson and J. R. Singler. Nonlinear model reduction using group proper orthogonal decomposition. *Int. J. Numer. Anal. Model.*, 7(2):356–372, 2010. 4

[30] V. Druskin and L. Knizhnerman. Extended Krylov subspaces: approximation of the matrix square root and related functions. *SIAM Journal on Matrix Analysis and Applications*, 19(3):755–771, 1998. 3.2.2

[31] V. Druskin, L. Knizhnerman, and V. Simoncini. Analysis of the rational Krylov subspace and ADI methods for solving the Lyapunov equation. *SIAM Journal on Numerical Analysis*, 49(5):1875–1898, 2011. 4.1

[32] M. Fahl. Computation of POD basis functions for fluid flows with Lanczos methods. *Math. Comput. Modelling*, 34(1-2):91–107, 2001. 4

[33] F. Feitzinger, T. Hylla, and E. W. Sachs. Inexact Kleinman-Newton method for Riccati equations. *SIAM J. Matrix Analysis Applications*, 31(2):272–288, 2009. 1

[34] L. Giraud, J. Langou, M. Rozložník, and J. van den Eshof. Rounding error analysis of the classical Gram-Schmidt orthogonalization process. *Numer. Math.*, 101(1):87–100, 2005. 4

[35] G.H. Golub and C. F. Van Loan. Matrix computations. *Johns Hopkins University, Press, Baltimore, MD, USA*, 1996. 3.3

[36] L. Grasedyck. Existence of a low rank or $H$-matrix approximant to the solution of a Sylvester equation. *Numer. Linear Algebra Appl*, 11:371–389, 2004. 3.2.1
[37] A. Hay, J.T. Borggaard, and D. Pelletier. Local improvements to reduced-order models using sensitivity analysis of the proper orthogonal decomposition. *J. Fluid Mech.*, 629:41–72, 2009.

[38] M. Heyouni and K. Jbilou. An extended block Arnoldi algorithm for large-scale solutions of the continuous-time algebraic Riccati equation. *Electron. Trans. Numer. Anal.*, 33:53–62, 2009.

[39] P. Holmes, J. L. Lumley, G. Berkooz, and C. W. Rowley. *Turbulence, Coherent Structures, Dynamical Systems and Symmetry*. Cambridge Monographs on Mechanics. Cambridge University Press, Cambridge, second edition, 2012.

[40] I.M. Jaimoukha and E.M. Kasenally. Krylov subspace methods for solving large Lyapunov equations. *SIAM J. Numer. Anal.*, 31(1):227–251, 1994.

[41] K. Jbilou. Block Krylov subspace methods for large algebraic Riccati equations. *Numerical Algorithms*, 34(2-4):339–353, 2003.

[42] K. Jbilou. An Arnoldi based algorithm for large algebraic Riccati equations. *Applied Mathematics Letters*, 19(5):437–444, 2006.

[43] K. Jbilou and A.J. Riquet. Projection methods for large Lyapunov matrix equations. *Linear Algebra and its Applications*, 415:344 – 358, 2006. Special Issue on Order Reduction of Large-Scale Systems.

[44] T. Kailath. Some Chandrasekhar-type algorithms for quadratic regulators. In *Proceedings of the 1972 IEEE Conference on Decision and Control and 11th Symposium on Adaptive Processes.*, volume 11, pages 219–223, 1972.

[45] D. Kasinathan and K. Morris. $H_{\infty}$-optimal actuator location. *IEEE Trans. Automat. Control*, 58(10):2522–2535, 2013.

[46] C. Kenney, A.J. Laub, and M. Wette. Error bounds for Newton refinement of solutions to algebraic Riccati equations. *Mathematics of Control, Signals and Systems*, 3(3):211–224, 1990.

[47] D.L. Kleinman. On the iterative technique for Riccati equation computations. *IEEE Transactions on Automatic Control*, 13:114–115, 1968.

[48] B. Kramer. Solving algebraic Riccati equations via proper orthogonal decomposition. In *Proceedings of the 19th IFAC World Congress*, pages 7767–7772, 2014.

[49] K. Kunisch and S. Volkwein. Galerkin proper orthogonal decomposition methods for a general equation in fluid dynamics. *SIAM Journal on Numerical analysis*, 40(2):492–515, 2002.

[50] H. Kwakernaak and R. Sivan. *Linear Optimal Control Systems*. Wiley-Interscience, New York, 1972.

[51] C. H. Lee and H. T. Tran. Reduced-order-based feedback control of the Kuramoto-Sivashinsky equation. *J. Comput. Appl. Math.*, 173(1):1–19, 2005.

[52] Y. Lin and V. Simoncini. A new subspace iteration method for the algebraic Riccati equation. *Numerical Linear Algebra with Applications*, 2014. doi: 10.1002/nla.1936.
[53] A. Lindquist. A new algorithm for optimal filtering of discrete-time stationary processes. *SIAM Journal on Control*, 12(4):736–746, 1974. 1

[54] M. Opmeer. Decay of singular values of the Gramians of infinite-dimensional systems. Preprint. 1

[55] J.O. Pralits and P. Luchini. Riccati-less optimal control of bluff-body wakes. In *Seventh IUTAM Symposium on Laminar-Turbulent Transition*, pages 325–330. Springer Netherlands, 2010. 1

[56] C.W. Rowley. Model reduction for fluids, using balanced proper orthogonal decomposition. *International Journal of Bifurcation and Chaos*, 15(03):997–1013, 2005. 4

[57] M. Rozložník, M. Tůma, A. Smoktunowicz, and J. Kopal. Numerical stability of orthogonalization methods with a non-standard inner product. *BIT*, 52(4):1035–1058, 2012. 4

[58] Y. Saad. Numerical solution of large Lyapunov equations. In *Signal Processing, Scattering and Operator Theory, and Numerical Methods (Amsterdam, 1989)*, volume 5 of *Progr. Systems Control Theory*, pages 503–511. Birkhäuser Boston, Boston, MA, 1990. 1, 3.2.1, 4

[59] O. Semeraro, J. O. Pralits, C.W. Rowley, and D.S. Henningson. Riccati-less approach for optimal control and estimation: an application to two-dimensional boundary layers. *Journal of Fluid Mechanics*, 731:394–417, 2013. 1

[60] D. Silvester, H. Elman, and A. Ramage. Incompressible Flow and Iterative Solver Software (IFISS), version 3.3, October 2013. 6

[61] V. Simoncini. A new iterative method for solving large-scale Lyapunov matrix equations. *SIAM J. Sci. Comput*, 29:1268–1288, 2007. 3, 3.2.2, 5

[62] V. Simoncini. Computational methods for linear matrix equations. Submitted, January 2014. 4.1

[63] V. Simoncini, D. B. Szyld, and M. Monsalve. On two numerical methods for the solution of large-scale algebraic Riccati equations. *IMA Journal of Numerical Analysis*, 2013. doi: 10.1093/imanum/drt015. 1, 3, 4.1

[64] J. R. Singler. Balanced POD for model reduction of linear PDE systems: convergence theory. *Numer. Math.*, 121(1):127–164, 2012. 6

[65] J. R. Singler and B. A. Batten. Balanced POD for linear PDE robust control computations. *Comp. Opt. and Appl.*, 53(1):227–248, 2012. 1

[66] J.R. Singler. Convergent snapshot algorithms for infinite-dimensional Lyapunov equations. *IMA Journal of Numerical Analysis*, 31(4):1468–1496, 2011. 4, 6

[67] L. Sirovich. Turbulence and the dynamics of coherent structures. I. Coherent structures. *Quart. Appl. Math.*, 45(3):561–571, 1987. 4

[68] J. Sun. Residual bounds of approximate solutions of the algebraic Riccati equation. *Numerische Mathematik*, 76(2):249–263, 1997. 3.3, 3.3

[69] L.N. Trefethen and M. Embree. *Spectra and Pseudospectra*. Princeton University Press, Princeton, NJ, 2005. 5
[70] E.I. Verriest. Low sensitivity design and optimal order reduction for the LQG-problem. In Proc. 24th Midwest Symp. Circ. Syst., Albuquerque, NM, USA, pages 365–369, 1981. 1

[71] S. Volkwein. Proper orthogonal decomposition for linear-quadratic optimal control. Lecture Notes, University of Konstanz, 2013. 4

[72] S. Volkwein. Proper orthogonal decomposition: Theory and reduced-order modelling. Lecture Notes, University of Konstanz, 2013. 4, 4

[73] K. Willcox and J. Peraire. Balanced model reduction via the proper orthogonal decomposition. AIAA Journal, pages 2323–2330, 2002. 4, 4, 6

[74] T. Wolf, H. Panzer, and B. Lohmann. On the residual of large-scale Lyapunov equations for Krylov-based approximate solutions. In Proceedings of the American Control Conference, pages 2606–2611, 2013. 5.1, 5.1