An inverse-free ADI algorithm for computing Lagrangian invariant subspaces

Volker Mehrmann* Federico Poloni†

04.08.14

The numerical computation of Lagrangian invariant subspaces of large scale Hamiltonian matrices is discussed in the context of the solution of Lyapunov and Riccati equations. A new version of the low-rank alternating direction implicit method is introduced, which in order to avoid numerical difficulties with solutions that are of very large norm, uses an inverse-free representation of the subspace and avoids inverses of ill-conditioned matrices. It is shown that this prevents large growth of the elements of the solution which may destroy a low-rank approximation of the solution. A partial error analysis is presented and the behavior of the method is demonstrated via several numerical examples.

Keywords: Lagrangian subspace, permuted Lagrangian subspace, Lyapunov equation, Riccati equation, low-rank ADI method, inverse-free arithmetic, permuted graph basis.

Mathematical Subject classification: 65F15, 65F50, 15A18, 15A22, 93B36, 93B40

1 Introduction

The computation of Lagrangian invariant subspaces of Hamiltonian matrices and the associated solution of Lyapunov and Riccati equations is an important task in the numerical solution of different control related problems, such as stabilization, robust control, Kalman filtering, subspace correction methods, model reduction and many other applications, see e.g. [1, 2, 10, 22, 23, 25, 29, 39, 50, 49, 54].

*Institut für Mathematik, MA 4-5, TU Berlin, Straße des 17. Juni 136, D-10623 Berlin, Fed. Rep. Germany. mehrmann@math.tu-berlin.de. Partially supported by DFG Research Center MATHEON 'Mathematics for key technologies' in Berlin.

†Università di Pisa, Dipartimento di Informatica, Largo B. Pontecorvo 3, 56127 Pisa, Italy. fpoloni@di.unipi.it. Partly supported by the A. von Humboldt Foundation and INDAM (Istituto Nazionale di Alta Matematica).

Given matrices $F, W = W^T \in \mathbb{R}^{n,n}$, the set of real $n \times n$ matrices, a *Lyapunov equation* is a symmetric linear matrix equation of the form

$$F^T X + XF + W = 0, \tag{1}$$

to be solved for a symmetric matrix $X = X^T \in \mathbb{R}^{n,n}$.

Lyapunov equations are a well-known tool for the characterization of the stability of dynamical systems [19, 30] and, furthermore, they arise as the linear systems in every step of the Newton-Kleinman algorithm [29, 39] for the solution of *algebraic Riccati equations*

$$F^T X + XF + W - XGX = 0, \tag{2}$$

with $F, G = G^T, W = W^T \in \mathbb{R}^{n,n}$. Lyapunov equations are just a special case of Riccati equations with vanishing coefficient G of the quadratic term.

1.1 Stabilization and optimal control

Let us briefly recall the origin of the Riccati equation in optimal control and stabilization of dynamical systems, see [22, 29, 39, 45, 47] for standard references. Consider a linear constant coefficient dynamical system of the form

$$\dot{x} = Fx + Bu, \qquad x(t_0) = x^0, \tag{3}$$

where x is the state, x^0 is an initial vector, u is the control input of the system and the matrices $F \in \mathbb{R}^{n,n}$, $B \in \mathbb{R}^{n,m}$ are constant. The classical *stabilization problem* is to find a *state feedback control law* $u = Kx$, such that the system matrix $A + BK$ of the *closed loop system*

$$\dot{x} = (F + BK)x. \tag{4}$$

is *stable*, i.e., it has all eigenvalues in the open left half plane. A common approach to compute such a *stabilizing feedback* in the solution of the stabilization problem but also for other control tasks is to solve an optimal control problem to minimize the cost functional

$$\int_{t_0}^{\infty} x^T W x + u^T R u \; dt \tag{5}$$

subject to (3), where $W = W^T \in \mathbb{R}^{n,n}$, is positive semidefinite and $R = R^T \in \mathbb{R}^{m,m}$ is positive definite.

The necessary optimality system [39] for this optimal control problem is given by the boundary value problem

$$\begin{bmatrix} 0 & I & 0 \\ -I & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \dot{\lambda} \\ \dot{x} \\ \dot{u} \end{bmatrix} = \begin{bmatrix} 0 & F & B \\ F^T & W & 0 \\ B^T & 0 & R \end{bmatrix} \begin{bmatrix} \lambda \\ x \\ u \end{bmatrix}, \; x(t_0) = x^0, \lim_{t \to \infty} \lambda(t) = 0, \tag{6}$$

where λ is a Lagrange multiplier. If (3) is *stabilizable*, i.e., $\mathrm{rk}[\lambda I - F, B] = n$ for all λ in the closed right half complex plane, then the optimal control u that gives the minimum of (5) is of the feedback form $u = Kx$ and it is a stabilizing feedback. The matrices W, R

in the cost functional are usually chosen to make the closed loop system (4) robust to perturbations or uncertainties, [54]. Since R is positive definite, one can solve the last equation in (6) as $u = -R^{-1}B^T\lambda$ and insert it into the other equations to obtain, after some permutation and scaling with -1, the *Hamiltonian boundary value problem*

$$\begin{bmatrix} \dot{x} \\ \dot{\lambda} \end{bmatrix} = \begin{bmatrix} F & -BR^{-1}B^T \\ -W & -F^T \end{bmatrix} \begin{bmatrix} x \\ \lambda \end{bmatrix}, \; x(t_0) = x^0, \lim_{t\to\infty} \lambda(t) = 0, \tag{7}$$

with the *Hamiltonian matrix*

$$H = \begin{bmatrix} F & -G \\ -W & -F^T \end{bmatrix}, \tag{8}$$

where we have introduced $G := BR^{-1}B^T$.

The most common approach for the solution of the boundary value problem (7), that is implemented in almost all design packages [11, 32] is to decouple the forward (for x) and backward (for λ) differential equation in (7) by solving the algebraic Riccati equation (2) or alternatively by computing an *orthogonal Lagrangian invariant subspace* of $\mathcal{H}$ associated with the eigenvalues in the left half complex plane. Here, an n-dimensional subspace $\mathcal{S} \subset \mathbb{R}^{2n}$ is called *Lagrangian* if $u^T\mathcal{J}_{2n}v = 0$ for all $u, v \in \mathcal{S}$, where

$$\mathcal{J}_{2n} := \begin{bmatrix} 0 & I_n \\ -I_n & 0 \end{bmatrix},$$

and to achieve backward numerical stability in the numerical computation of this subspace usually an orthogonal basis is used.

For small or medium size problems the desired orthogonal basis of this Lagrangian subspace can be computed with the backward stable methods of [13, 36, 52] and the stabilizability condition assures that such an orthogonal Lagrangian invariant subspace exists. Suppose that this subspace is spanned by the columns of a matrix

$$S = \begin{bmatrix} S_1 \\ S_2 \end{bmatrix},$$

with $S_1, S_2 \in \mathbb{R}^{n,n}$, i.e.,

$$HS = ST$$

where $T \in \mathbb{R}^{n,n}$ has only eigenvalues in the open left half plane. Under the stabilizability assumption, it is known, see e.g. [29, 39], that S_1 is invertible, and that $X = S_2S_1^{-1}$ is the unique symmetric positive semidefinite solution of the algebraic Riccati equation (2), while $K = -R^{-1}B^TX$ is the feedback matrix, and $C = F + BK = F - GX$ is the closed loop system matrix.

This means that the Riccati solution and thus also the feedback matrix is associated with the special representation of the Lagrangian invariant subspace

$$\begin{bmatrix} I \\ X \end{bmatrix} = SS_1^{-1} \tag{9}$$

and the closed loop system matrix is given by $C = S_1 T S_1^{-1}$.

If any basis matrix S of the invariant subspace $\mathcal{S}$ has been computed, then the positive semidefinite Riccati solution X can be determined by solving the linear system

$$XS_1 = S_2, \tag{10}$$

the feedback matrix K is obtained by solving the system

$$RKS_1 = -B^T S_2 \tag{11}$$

and the closed loop matrix C from the linear system

$$CS_1 = FS_1 - BR^{-1}B^T S_2. \tag{12}$$

Thus we can determine K and C directly from the invariant subspace without first computing the Riccati solution X. This observation is essential if large errors occur in the solution of the Riccati equation due ill-conditioning of the equation, which happens e.g. in optimal H_∞ control, see [8, 7, 33, 54], and when the system is close to a system that is non-stabilizable [14, 27, 39]. In these cases all three linear systems (10)–(12) may be difficult to solve and the solutions may be highly inaccurate even though the computation of the subspace $\mathcal{S}$ is still well-conditioned. This fact is exploited in [8, 7, 31, 33] to improve the computation of optimal H_∞ controllers, and it is obvious that first computing an inaccurate X and then determining C and K via this X is worse than computing C and K directly.

This observation is of particular importance in the low-rank approximation of Riccati solutions from Lagrangian subspaces which we discuss in the next subsection.

1.2 Low-rank approximation of Lagrangian subspaces

In the case of small-scale dense problems, there exist accurate structure preserving-methods of $O(n^3)$ complexity to compute an orthogonal basis of the desired Lagrangian invariant subspace, [13, 36, 52]. These methods are, however, limited in the problem size that can be handled, due to the complexity and the storage requirements ($O(n^2)$ usually), and hence infeasible for n large.

In many problems originating form model based control with models described by partial differential equations (after space discretization) the system matrices are large and sparse, and the matrices G, W are of low rank, given by the usually low number of inputs and outputs of the control system, respectively [9, 42]. Furthermore, often the eigenvalues of the Riccati solution X decay very quickly [43], so that X can be well-approximated by a low-rank matrix $X \approx ZZ^T$, where Z has very few (say k) columns, i.e., there exists an approximation to the stabilizing Lagrangian invariant subspace that is sparsely represented via

$$S_Z = \begin{bmatrix} I \\ ZZ^T \end{bmatrix}.$$

This sparse low-rank representation needs only storage of $O(nk)$ for the low-rank matrix $Z \in \mathbb{R}^{n,k}$. It is, however, directly connected to the Riccati solution and thus, in view

of the discussion in the last subsection, may be very inaccurate when the full Riccati solution is inaccurate, so that ZZ^T is of low rank but of very large norm.

A low-rank approximation of the positive semidefinite Riccati solution X and the feedback matrix K can be also be determined directly from a partial Lagrangian invariant subspace associated to some particular part of the spectrum of the Hamiltonian matrix H. It is however, in general an open problem to decide which low-dimensional partial Lagrangian subspace of $\mathcal{S}$ corresponds to the low-rank part of the Riccati solution, see [3]. If it were known in advance to which eigenvalues of H this subspace belongs then one could employ structure preserving shift-and-invert Krylov-subspace methods [4, 5, 35, 38] to determine approximations to the subspace associated with this part of the spectrum. From this partial Lagrangian invariant subspace we could then determine low-rank approximations, since from $ZZ^T \approx S_2 S_1^{-1}$ it follows that for a low-rank representation we would need $S_2 = ZL$ with some matrix $L = Z^T S_1$. Thus, if we would know which invariant subspace we should compute, we could directly compute Z and W from a Krylov subspace method, rather than computing the full representation $\mathcal{S}$. We will discuss the theoretical background, and how to obtain such a representation in Section 1.3.

The currently most widely used method for large scale problems in practice, however, uses the approximate sparse representation of the positive semidefinite Riccati solution. A classical method for this approach is the Newton-Kleinman algorithm [24, 29, 39], which for a given initial matrix $X_0 = X_0^T$ for which $X_0 - X$ is positive semidefinite, the iteration generates a monotonically decreasing sequence $X_{j+1} = X_j + Y_j$, $j = 0, 1, 2, \ldots$, where the correction term Y_j is obtained as the solution of the Lyapunov equation

$$0 = F_j^T Y_j + Y_j F_j + W_j,$$

with $F_j := F - GX_j$ being a low-rank approximation to the closed loop system matrix $C = F + GX$, and $W_j := W - X_j^T G X_j - X_j F - F^T X_j$ being the residual of the approximation. Furthermore, if the Riccati solution has a sparse low-rank approximation, then also the iterates X_j in the Newton-Kleinman algorithm can be effectively approximated by a low-rank matrix $X_j \approx Z_j Z_j^T$. This property is heavily exploited in the current numerical solution methods, which proceed by computing the approximate low-rank factor Z_j instead of the Lyapunov solution X_j, see [9, 12, 42, 43, 48] and the references therein.

At each step of the Newton-Kleinman algorithm one has to solve a large scale Lyapunov equation and also this typically is approached by iterative methods such as the alternating direction implicit (ADI) method, see e.g. [20, 42, 43, 51] or other iterative techniques such as [21, 40]. Thus, for the Riccati equation and the optimal control problem, one obtains an iteration with an outer loop given by the Newton-Kleinman method and an inner iteration for the solution of the Lyapunov equation in every step. Furthermore, to reduce storage requirements and complexity, one ideally works only on low-rank representations of the current iterate.

1.3 Analysis of Riccati solutions and low-rank approximations

To analyze the properties of Riccati solutions, in particular, in the case of low-rank approximations to Riccati solutions, we use the following result of [41] on the *CS*-

decomposition of matrices representing orthogonal Lagrangian subspaces.

Lemma 1.1. *[CS-decomposition] If the columns of*

$$S = \begin{bmatrix} S_1 \\ S_2 \end{bmatrix} \in \mathbb{R}^{2n,n} \tag{13}$$

form an orthogonal basis of a Lagrangian subspace, then there exist real orthogonal matrices $U, V \in \mathbb{R}^{n,n}$ *such that*

$$U^T S_1 V = \Gamma = \mathrm{diag}(\gamma_1, \ldots, \gamma_n),\ U^T S_2 V = \Delta = \mathrm{diag}(\delta_1, \ldots, \delta_n), \tag{14}$$

with $\Gamma^2 + \Delta^2 = I_n$.

Applying the CS-decomposition of Lemma 14 to S, as in (13), spanning the stabilizing Lagrangian invariant subspace of a Hamiltonian matrix H as in (8), we see that the solution of the Riccati equation (2) $X = S_2 S_1^{-1}$ has the spectral decomposition

$$X = U^T \Delta \Gamma^{-1} U$$

with eigenvalues

$$\lambda_i = \delta_i \gamma_i^{-1} = \frac{\delta_i}{\sqrt{1 - \delta_i^2}},\ i = 1, \ldots, n,$$

and the small eigenvalues of X (which we can omit in the low rank representation) are related to δ_i being close to 0 while the large eigenvalues of X (which we would like to keep) are related to δ_i^2 being close to 1. Thus, to obtain a low-rank approximation $X = ZZ^T + \Psi$ with $\|\Psi\|_2 \leq \epsilon$, we can use the CS-decomposition and obtain

$$\Delta \Gamma^{-1} = (I - \Gamma^2)^{1/2} \Gamma^{-1} = U^T \Lambda_1 U + U^T \Lambda_2 U = U^T Z Z^T U + U^T \Psi U.$$

where Λ_1 is obtained by setting in $\Delta \Gamma^{-1}$ all those λ_i to zero that are smaller than $\epsilon \lambda_\ell$, with ℓ being an index where λ_i is maximal. A simple calculation shows that this is the case if

$$|(1 - {\gamma_i}^2){\gamma_\ell}^2| \leq \epsilon^2 |{\gamma_i}^2 (1 - {\gamma_\ell}^2)|. \tag{15}$$

We also have a measure for the error that we commit in this way which is given by $\|\Psi\| = \|\Lambda_2\|$.

Example 1.2. Let $\varepsilon \ll 1$ and consider the subspace

$$S = \begin{bmatrix} \gamma_1 & 0 & 0 \\ 0 & \gamma_2 & 0 \\ 0 & 0 & \gamma_3 \\ \delta_1 & 0 & 0 \\ 0 & \delta_2 & 0 \\ 0 & 0 & \delta_3 \end{bmatrix}, \quad \gamma_1 = \delta_3 = \varepsilon, \gamma_3 = \delta_1 = \sqrt{1 - \varepsilon^2}, \gamma_2 = \delta_2 = \frac{\sqrt{2}}{2}.$$

The associated Riccati solution is

$$X = \begin{bmatrix} \frac{\sqrt{1-\varepsilon^2}}{\varepsilon} & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & \frac{\varepsilon}{\sqrt{1-\varepsilon^2}} \end{bmatrix}.$$

A perturbation in X of magnitude $O(\varepsilon^2)\|X\|_2 = O(\varepsilon)$ will translate into a perturbation of magnitude $O(\varepsilon^2)$ on the largest eigenvalue, of magnitude $O(\varepsilon)$ on the second one, and $O(1)$ on the smallest eigenvalue. On the other hand, a perturbation in S of magnitude $O(\varepsilon^2)\|S\|$ will give rise to a perturbation of magnitude $O(\varepsilon^2)$ in the second eigenvalue and $O(\varepsilon)$ on the other two. So using S as a representation to compute and store the invariant subspace will give a solution that is less accurate on the dominant eigenvalues, but overall more faithful to the original one if one keeps into account all the components. For instance, numerical experiments with an artificial perturbation on all the matrix entries, $\varepsilon = 10^{-6}$, for an input coefficient $B = \begin{bmatrix} \varepsilon & 1 & \varepsilon \end{bmatrix}^T$ and a weight matrix $R = 1$ show that the feedback matrix computed with a perturbed X and $K = -R^{-1}B^TX$ satisfies

$$K - \tilde{K} = \begin{bmatrix} O(\varepsilon) & O(\varepsilon) & O(\varepsilon) \end{bmatrix},$$

while an artificially perturbed S and (11) give

$$K - \tilde{K} = \begin{bmatrix} O(\varepsilon) & O(\varepsilon^2) & O(\varepsilon^2) \end{bmatrix}.$$

The difference in accuracy is not evident in a norm-wise bound, but the difference in precision between the smallest components is significant.

We also see from the CS-decomposition when large ill-conditioning of the matrix S_1 arises. Since the diagonal elements δ_j in the bottom block are bounded by 1 in absolute value, large eigenvalues of X arise from small elements γ_i (in absolute value) in the top block. These are, however, also responsible for the ill-conditioning of the top block which may lead to inaccuracies in the Riccati solution. The CS-decomposition allows to determine when this ill-conditioning happens and shows how large inaccuracies can be avoided. If the orthogonal basis of the Lagrangian invariant subspace is computed with small error, i.e., the computed matrices $\tilde{S}_1, \tilde{S}_2$ are close to the exact S_1, S_2 then the CS-decomposition presents an ideal method to compute accurate approximations and to determine the error in the computed Riccati solution $\tilde{X}$, the computed closed loop matrix $\tilde{C}$ and the computed feedback matrix $\tilde{K}$.

The same analysis can also be applied when (in the large scale case) only a subspace of the Lagrangian subspace is computed, which would be the case if an isometric Krylov-subspace method is used to compute the space associated with a particular subset of eigenvalues of the Hamiltonian matrix H. As mentioned before, it is not known in advance, which eigenvalues of the Hamiltonian matrix are associated with the large eigenvalues of X and thus with the best low-rank solution. Approximations of these desired eigenvalues would be ideal candidates for the shifts in the shift-and-invert isometric Arnoldi methods of [35, 38].

1.4 Permuted graph representations

Due to the unresolved difficulty with the use of Krylov subspace methods, in this paper we discuss an improvement of the Newton-Kleinman iteration for the case that the Riccati solution is ill-conditioned. We mainly discuss the inner loop, i.e., the iterative solution of a Lyapunov equation of the form (1) that has a positive semidefinite solution X which can be well-approximated by a low-rank matrix $X = ZZ^T$. The low-rank alternating directions implicit (LR-ADI) method [9, 20, 42, 43, 51] approaches the solution of (1) by alternating the approximate, low-rank solution of linear systems with $A - p_i I$ and $(A - p_i I)^T$ for specially chosen ADI parameters p_i. One of the challenges in the practical use of the LR-ADI method is the choice of these parameters as well as the control of the convergence speed and the approximation quality of the low-rank approximation, see [9].

In view of the previous discussion, for the computation of Lagrangian invariant spaces or subspaces associated with low-rank approximations, we have two major alternative approaches. We can either compute a low-rank approximate solution of the algebraic Riccati equation with the potential danger of large inaccuracy due to the ill-conditioning of the representation, or we can compute an orthogonal basis of the associate Lagrangian subspace with the disadvantage of high storage requirements. Also at current we do not know how to extract a low-rank representation from this subspace without computing and storing the whole subspace first.

A compromise between these two extremes has recently been proposed in [33], for small-scale dense control problems, where a special basis of the Lagrangian subspace, called *permuted graph representation*, has been introduced. It has been shown that for every Lagrangian subspace $\mathcal{S}$ there always exist an orthogonal matrix Π (in fact, a permutation matrix except for some entries 1 being replaced by -1) and a symmetric matrix N with bounded norm such that

$$\mathcal{S} = \operatorname{im} \Pi^T \begin{bmatrix} I \\ N \end{bmatrix}.$$

This choice of the basis combines the advantages of both approaches: like an orthogonal basis, its condition number is moderate, which means that the column space is stable with respect to perturbations in the matrix entries, and the Lagrangian structure is preserved exactly, since it is equivalent to the symmetry of N.

In this paper we show that similar techniques can be used in the large and sparse case: the low-rank ADI algorithm can be modified to use a similar implicit representation of the iterates, returning in the end a basis matrix for $\mathcal{S}$ of the form

$$U = \begin{bmatrix} I_n - U_1 U_2^T \\ U_3 U_4^T \end{bmatrix}, \tag{16}$$

where U_1, U_2, U_3, U_4 are long thin matrices with bounded norm, and $\kappa(U)$ can be bounded explicitly.

The property of the ADI method of working with low-rank representations during the iteration process can be used while at the same time avoiding the potentially large norm growth of the Riccati solution due to ill-conditioning.

Our construction is based on the well-known fact, see e.g. [16], that computing the eigenvalues and eigenvectors of a square matrix pencil $\lambda E - A$, with $E, A \in \mathbb{R}^{n,n}$, is numerically much better than computing the eigenvalues of the matrix representation $E^{-1}A$. Not only is it possible to treat problems in which E (or the whole pencil) is singular, but even when E is invertible, the resulting accuracy that can be achieved is higher. The pencil representation not only allows to avoid forming $E^{-1}A$, in [6] this idea of a *pencil arithmetic* has been extended to all basic arithmetic operations on matrices without computing the inverses explicitly.

In the context of this pencil arithmetic, however, the pair (E, A) is highly non-unique; for every nonsingular $M \in \mathbb{R}^{n,n}$, the pair (ME, MA) represents the same matrix. A natural choice to achieve some kind of uniqueness is to require that the columns of $[E \quad A]^T$ are orthonormal, see [6], which however is usually a non-sparse representation.

Recently in [33, 34] the permuted graph representation was employed to achieve a different choice, by requiring that

$$\begin{bmatrix} E^T \\ A^T \end{bmatrix} = \Pi^T \begin{bmatrix} I_n \\ N \end{bmatrix}, \tag{17}$$

where $\Pi \in \mathbb{R}^{2n,2n}$ is a permutation matrix and $N \in \mathbb{R}^{n,n}$ is bounded in norm. In this representation the deviation from orthonormality can be controlled, while at the same time getting a more sparse representation in which also the property of the subspace to be Lagrangian is easier to control.

The pencil arithmetic has been applied to the solution of dense small-scale algebraic Riccati equations [6, 33]. If care is taken in the representation of the non-unique pair (E, A) and in the implementation of the arithmetic operations, then these algorithms can be significantly more stable than their counterparts that work on $M = E^{-1}A$ directly [33].

1.5 Aim and content

The aim of this paper is to demonstrate that the use of inverse-free and permuted graph representations can be beneficial in the computation of Lagrangian subspaces and the solution of large-scale Lyapunov and Riccati equations via the LR-ADI method. To this purpose, we use similar representations for tall and skinny matrices M, and modify the main LR-ADI loop to use them. We show how to apply sparse operators to them, and add new operations such as horizontal concatenation and transpositions to the framework of [6], while keeping the computational cost linear in the larger dimension. The representation (17) plays an important role in the final part of the algorithm, as it allows easier control of sparsity or approximate sparsity than an orthogonal representation for both rectangular and square matrices.

We do not wish to propose the algorithm presented here as an ultimate solver for sparse control problems; our goal in this paper is to demonstrate that these techniques can be useful also for large and sparse matrices, at least in some cases: we give a proof of concept showing that in some examples they give better numerical accuracy than the standard methods.

The paper proceeds as follows. In Section 2 we recall some basic results on permuted graph representations. In Section 4 we describe the low-rank version of the ADI method using permuted graph bases. We discuss the different representations and their residuals in Section 6 and present a partial error analysis in Section 7.

Numerical experiments in Section 8 are followed by a Conclusion.

2 Inverse-free arithmetic and permuted graph representations

In this section, we recall a few tools from [6] and [33, 34], although using a novel presentation that is more suitable to our exposition.

Given a matrix $M \in \mathbb{R}^{n,m}$, we call a pair $(A, E) \in \mathbb{R}^{n,m} \times \mathbb{R}^{m,m}$ such that $M = AE^{-1}$ a *(right-looking) inverse-free representation* of M. We use the notation

$$\mathrm{IF}(M) := \begin{bmatrix} E \\ A \end{bmatrix} \in \mathbb{R}^{n+m,m}$$

to denote any block matrix $\begin{bmatrix} E^T & A^T \end{bmatrix}^T$ whose blocks satisfy this condition $M = AE^{-1}$. Note that this is a slight abuse of notation, since the matrix $\begin{bmatrix} E^T & A^T \end{bmatrix}^T$ is not uniquely defined by this condition, but only the subspace $\operatorname{im} \begin{bmatrix} E^T & A^T \end{bmatrix}^T$ is.

Let $\sigma_{\min}(M)$ and $\sigma_{\max}(M)$ denote the smallest and largest singular value of a matrix M, see [16], and let $\kappa(M) := \sigma_{\max}(M)/\sigma_{\min}(M)$ denote the spectral norm condition number of M. The condition number of the column space $\operatorname{im} M$ as a function of the matrix M is given by $\kappa(M)$ (see [18] and Lemmas 7.1 and 7.3 in the following). We call a matrix with full column rank whose columns form a basis for a subspace $\mathcal{U}$ a *basis matrix* for $\mathcal{U}$.

When computing an inverse-free representation of a matrix M, we usually aim for a small value of its subspace condition number $\kappa(\mathrm{IF}(M))$. Obviously, the best choice is to use an orthogonal basis matrix. The following result gives a different well-conditioned representation.

Theorem 2.1 ([26])**.** *Let $U \in \mathbb{R}^{n+m,m}$ have full column rank, and let $\tau \geq 1$.*

1. *There exists an invertible submatrix $T \in \mathbb{R}^{m,m}$ of U, a permutation matrix $\Pi \in \mathbb{R}^{n+m,n+m}$ and $N = [N_{i,j}] \in \mathbb{R}^{n,m}$ with $|N_{ij}| \leq \tau$ for all $i = 1, \ldots, n$, $j = 1, \ldots, m$ such that*

$$\hat{U} = UT^{-1} = \Pi^T \begin{bmatrix} I_m \\ N \end{bmatrix}. \tag{18}$$

2. *If $\tau > 1$, then the matrices Π, N and T can be computed in $O\left(m^2(n+m)\frac{\log m}{\log \tau}\right)$ floating point operations.*

3. The condition numbers of $\hat{U}$ and T satisfy

$$\kappa\left(\hat{U}\right) = \kappa\left(\Pi^T \begin{bmatrix} I_m \\ N \end{bmatrix}\right) \leq \sqrt{mn\tau^2 + 1},$$
$$\kappa(T) \leq \kappa(U)\sqrt{mn\tau^2 + 1}.$$

Hence we can always choose a matrix $\mathrm{IF}(M)$ in the form (18) so that its conditioning is controlled by the choice of τ. A typical choice for τ is a small constant such as 2, or, if one is worried by the logarithmic factors in the computational cost, a small power of n like $n^{1/3}$.

The next trick that we need is a method to compute an inverse-free representation of the transpose of a matrix.

Lemma 2.2. *Let $M \in \mathbb{R}^{n,m}$, and let $U = \mathrm{IF}(M)$ for some $U \in \mathbb{R}^{n+m,m}$. Let $\begin{bmatrix} \hat{A}^T & -\hat{E}^T \end{bmatrix}^T$ be any basis matrix for $\ker U^T$, with $\hat{E} \in \mathbb{R}^{n\times n}$, $\hat{A} \in \mathbb{R}^{m\times n}$, then $\begin{bmatrix} \hat{E}^T & \hat{A}^T \end{bmatrix}^T = \mathrm{IF}(M^T)$.*

Proof. Let $U = \begin{bmatrix} E^T & A^T \end{bmatrix}^T$, with $E \in \mathbb{R}^{m\times m}$, $A \in \mathbb{R}^{n\times m}$. First note that U has full column rank, since its submatrix E is nonsingular, and hence a basis matrix for $\ker U^T$ has indeed n columns. By the definition of the kernel, $E^T\hat{A} = A^T\hat{E}$, i.e., $\hat{A} = (AE^{-1})^T\hat{E} = M^T\hat{E}$. If $\hat{E}$ were singular, then $\hat{E}v = 0$ for some vector $v \in \mathbb{R}^m$, $v \neq 0$, hence also $\hat{A}v = M^T\hat{E}v = 0$ and $\begin{bmatrix} \hat{A}^T & -\hat{E}^T \end{bmatrix}^T v = 0$, which is impossible since $\begin{bmatrix} \hat{A}^T & -\hat{E}^T \end{bmatrix}^T$ must have full column rank by construction. So $\hat{E}$ is invertible and $\hat{A}\hat{E}^{-1} = M^T$, as asserted. □

If $U = \mathrm{IF}(M)$ is in the format given by Theorem 2.1, then we can determine a basis $\mathrm{IF}(M^T)$ in the same format.

Theorem 2.3. *Consider $U = \mathrm{IF}(M)$ in the format given by Theorem 2.1, i.e.,*

$$\Pi^T \begin{bmatrix} I_m \\ N \end{bmatrix} = \mathrm{IF}(M).$$

Then, there exist a permutation matrix $\widehat{\Pi}$ and two real diagonal matrices $D_1 \in \mathbb{R}^{n,n}$, $D_2 \in \mathbb{R}^{m,m}$ with ± 1 on the diagonal such that

$$\mathrm{IF}(M^T) = \widehat{\Pi}^T \begin{bmatrix} I_n \\ D_2 N^T D_1 \end{bmatrix}. \tag{19}$$

Proof. Since

$$\begin{bmatrix} 0 & -I_n \\ I_m & 0 \end{bmatrix} \Pi \begin{bmatrix} 0 & I_m \\ -I_n & 0 \end{bmatrix}$$

is nothing more than a permutation with some additional sign changes, we can find a permutation matrix $\widehat{\Pi}$ and ± 1 diagonal matrices D_1, D_2 such that

$$\begin{bmatrix} 0 & -I_n \\ I_m & 0 \end{bmatrix} \Pi \begin{bmatrix} 0 & I_m \\ -I_n & 0 \end{bmatrix} \widehat{\Pi}^T \begin{bmatrix} D_1 & 0 \\ 0 & D_2 \end{bmatrix} = I.$$

We now employ Lemma 2.2. Since the matrix in the left-hand side of (19) has full column rank, we only have to verify that

$$\begin{bmatrix} I_m & N^T \end{bmatrix} \Pi \begin{bmatrix} 0 & I_m \\ -I_n & 0 \end{bmatrix} \widehat{\Pi}^T \begin{bmatrix} I_n \\ D_2 N^T D_1 \end{bmatrix} = 0.$$

We have indeed

$$\begin{aligned} 0 &= \begin{bmatrix} -N^T & I_m \end{bmatrix} \begin{bmatrix} I_n \\ N^T \end{bmatrix} D_1 \\ &= \begin{bmatrix} -N^T & I_m \end{bmatrix} \begin{bmatrix} 0 & -I_n \\ I_m & 0 \end{bmatrix} \Pi \begin{bmatrix} 0 & I_m \\ -I_n & 0 \end{bmatrix} \widehat{\Pi}^T \begin{bmatrix} D_1 & 0 \\ 0 & D_2 \end{bmatrix} \begin{bmatrix} I_n \\ N^T \end{bmatrix} D_1 \\ &= \begin{bmatrix} I_m & N^T \end{bmatrix} \Pi \begin{bmatrix} 0 & I_m \\ -I_n & 0 \end{bmatrix} \widehat{\Pi}^T \begin{bmatrix} I_n \\ D_2 N^T D_1 \end{bmatrix}. \end{aligned}$$

□

Note that the presented proof of Theorem 2.3 is completely constructive and can be turned into an algorithm to compute $\widehat{\Pi}$ and $D_2 N_1^T D_1$, which form a permuted graph representation for $\mathrm{IF}(M^T)$.

The ideas in the proofs of Lemma 2.2 and Theorem 2.3 have been developed in the literature (in different forms) for the case $m = n$, but we argue that the more interesting situation is when $m \ll n$. Indeed, in this case the basis 19 is sparse, while using orthogonal matrix machinery to determine the kernel in Lemma 2.2 would yield a more dense orthogonal matrix.

3 The low-rank ADI method

For the computation of low-rank approximations of Lagrangian invariant subspaces, using the Newton-Kleinman iteration for the Riccati solution and in each step for solution of the Lyapunov equation, we will employ the low-rank ADI (LR-ADI) method [9, 43]. We assume in the following that the Lyapunov equation (1) has a factored matrix $W = BB^T$, where $B \in \mathbb{R}^{n,m}$ and $m \ll n$. The low-rank ADI method, described in Algorithm 1, needs a sequence of *shifts* p_i. In some special cases optimal shifts are known and in general there are several heuristic techniques available to choose these shifts [12, 42, 43, 51]. The column compression in Line 6 of Algorithm 1 can be performed with the help of a rank revealing QR-decomposition [16], $Z^T = QRP$ with Q orthogonal, P a permutation matrix that is used for pivoting, and R upper triangular. We can replace Z_i with the

Input: $F \in \mathbb{R}^{n,n}$, $B \in \mathbb{R}^{n,m}$, a sequence of shifts $p_1, p_2, \ldots, p_{i_{\max}}$ such that $\operatorname{Re} p_i < 0$ for each $i = 1, 2, \ldots, i_{\max}$

Output: Z such that $ZZ^T \approx X$, where X solves (1)

$V \leftarrow \sqrt{-2 \operatorname{Re} p_1}(F^T + p_1 I_n)B$;

$Z \leftarrow V$;

for $i = 2, 3, \ldots, i_{\max}$ **do**

  $V \leftarrow \sqrt{\frac{\operatorname{Re} p_i}{\operatorname{Re} p_{i-1}}} \left(V - (p_i + \bar{p}_{i-1})(F^T + p_i I_n)^{-1} V\right)$;

  $Z \leftarrow \begin{bmatrix} Z & V \end{bmatrix}$;

  (optionally) $Z \leftarrow$ a matrix $Z^{(1)}$ with less columns than Z s.t. $Z^{(1)}(Z^{(1)})^T \approx ZZ^T$;

end

leading columns of $P^T R^T$, eliminating all those columns whose norm is below a certain threshold. This operation is typically performed only every few steps, to amortize the cost of the QRP decomposition. All the other lines of Algorithm 1 are self-explaining. The majority of the computational cost comes from the solution of the sparse linear systems with the matrices $F^T + p_i I_n$ and this can be effectively solved via a Krylov subspace method [46].

As discussed before, in many practically relevant examples the solution X is well approximated by a low-rank matrix ZZ^T and thus Z can be effectively compressed to have only a small number of columns.

4 Inverse-free LR-ADI

The goal of this paper is to perform the LR-ADI iteration using only an implicit inverse-free representation of the iterates. Namely, we store at each step $\mathrm{IF}(V_i)$ and $\mathrm{IF}(Z_i)$ instead of V_i and Z_i. For this to work, we need new methods to perform each step in Algorithm 1 on an inverse-free representations directly. These new methods are described in the following subsections.

After each operation, the representation is converted to an equivalent representation by replacing $\mathrm{IF}(V_i)$ or $\mathrm{IF}(Z_i)$ with their orthogonal QR-factor. Here and in the following, when we refer to the QR-factorization of a rectangular matrix $M = QR$, we use the so-called *thin QR*, i.e., the version in which Q is rectangular and R square. For simplicity, we define the operator $Q = \operatorname{orth}(M)$ that maps a matrix to its orthogonal QR-factor.

4.1 Initialization

As a starting inverse-free representation, we take

$$IF(V_1) = \operatorname{orth}\left(\begin{bmatrix} I_p \\ \sqrt{-2 \operatorname{Re} p_1}(F^T + p_1 I_n)B \end{bmatrix}\right).$$

4.2 Solution of sparse linear systems

For the operation in Line 4 of Algorithm 1, let E, A be such that $\begin{bmatrix} E \\ A \end{bmatrix} = IF(V_{i-1})$. We factor out E on the right to obtain

$$IF(V_i) = \operatorname{orth}\left(\begin{bmatrix} E \\ \sqrt{\frac{\operatorname{Re} p_i}{\operatorname{Re} p_{i-1}}}\left(A - (p_i + \bar{p}_{i-1})(F^T + p_i I_n)^{-1}A\right) \end{bmatrix}\right).$$

In principle, one could try to avoid the solution of linear systems with $F^T + p_i I_n$ as well, by replacing it with an equivalent inverse-free computation. However, this seems challenging to do in a computationally feasible way, due to the large size of the matrices involved.

4.3 Horizontal stacking

The operation in Line 5 of Algorithm 1 needs to be extended to inverse-free representations as well. Let $\begin{bmatrix} E_Z \\ A_Z \end{bmatrix} = \mathrm{IF}(Z_{i-1})$, $\begin{bmatrix} E_V \\ A_V \end{bmatrix} = \mathrm{IF}(V_{i-1})$, then it is simple to verify that we can choose

$$\mathrm{IF}(Z_i) = \mathrm{IF}\left(\begin{bmatrix} Z_{i-1} & V_{i-1} \end{bmatrix}\right) = \operatorname{orth}\left(\begin{bmatrix} E_Z & 0 \\ 0 & E_V \\ A_Z & A_V \end{bmatrix}\right).$$

Again, we re-orthogonalize after the computation. If $IF(Z_{i-1})$ is already an orthogonal matrix, then the first block column of this matrix is already orthogonal, so there is only a small amount of extra columns to orthogonalize against the previous columns in each step.

4.4 Column compression

Column compression in the usual LR-ADI algorithm is performed using a rank-revealing RQ decomposition. One determines an orthogonal matrix $Y \in \mathbb{R}^{m,m}$ such that $Z_i Y = \begin{bmatrix} Z^{(1)} & Z^{(2)} \end{bmatrix}$, with $\|Z^{(2)}\|_2 < \varepsilon \|Z_i\|_2$, and then replaces Z_i with $Z^{(1)}$, which gives

$$\|Z_i Z_i^T - Z^{(1)}(Z^{(1)})^T\|_2 = \|Z^{(2)}(Z^{(2)})^T\|_2 = \|Z^{(2)}\|_2^2 \leq \varepsilon^2 \|Z_i\|_2^2 = \varepsilon^2 \|Z_i Z_i^T\|_2.$$

In our extension based on inverse-free arithmetic, we use the generalized singular value decomposition (GSVD) instead, which is essentially an inverse-free version of the singular value decomposition.

Theorem 4.1 (Generalized SVD, [16, Theorem 8.7.4])**.** *Let $E \in \mathbb{R}^{m,m}$, $A \in \mathbb{R}^{n,m}$. There exist square orthogonal matrices $U \in \mathbb{R}^{m,m}$, $P \in \mathbb{R}^{n,n}$ and a nonsingular $V \in \mathbb{R}^{m,m}$ such that $E = U\Gamma V$, $A = P\Delta V$, with $\Gamma = [\gamma_{ij}] \in \mathbb{R}^{m,m}$, $\Delta = [\delta_{ij}] \in \mathbb{R}^{n,m}$ diagonal.*

For a rectangular Δ, being diagonal means that all entries Δ_{ij} with $i \neq j$ are zero. Note that if $\begin{bmatrix} E^T & A^T \end{bmatrix}^T$ were a (partial) Lagrangian subspace then this would just be the CS-decomposition of Lemma 1.1 with $P = Q$ and V orthogonal.

The following result shows how to perform column compression in inverse-free form.

Theorem 4.2. *Let* $IF(Z) = \begin{bmatrix} E \\ A \end{bmatrix}$, *with* $E \in \mathbb{R}^{m \times m}$, *and let* $E = U\Gamma V$, $A = P\Delta V$ *be a GSVD of the pair* E, A, *with the elements in* Γ *and* Δ *ordered so that the ratios* $\sigma_i = |\delta_{ii}\gamma_{ii}^{-1}|$ *are decreasing, Let* k *be such that* $\sigma_h \leq \varepsilon\sigma_1$ *if and only if* $h > k$, *and partition* $P = \begin{bmatrix} P_1 & P_2 \end{bmatrix}$, $\Delta = \begin{bmatrix} \Delta_1 & 0 \\ 0 & \Delta_2 \end{bmatrix}$, $\Gamma = \begin{bmatrix} \Gamma_1 & 0 \\ 0 & \Gamma_2 \end{bmatrix}$ *so that the size of the first block is* k. *Let* $Z^{(1)}$ *be the matrix such that*

$$\mathrm{IF}(Z^{(1)}) = \begin{bmatrix} \Gamma_1 \\ P_1\Delta_1 \end{bmatrix}.$$

Then, $\|ZZ^T - Z^{(1)}(Z^{(1)})^T\|_2 \leq \varepsilon^2\|ZZ^T\|_2$.

Proof. We have

$$\begin{aligned} \|ZZ^T - Z^{(1)}(Z^{(1)})^T\|_2 &= \|P\Delta\Gamma^{-2}\Delta^T P^T - P_1\Delta_1\Gamma_1^{-2}\Delta_1^T P_1^T\|_2 \\ &= \|P_2\Delta_2\Gamma_2^2\Delta_2^T P_2^T\|_2 = \|\Delta_2\Gamma_2^2\Delta_2^T\| \\ &= \max_{h>k} \sigma_i^2 \leq \varepsilon^2\sigma_1^2 = \|ZZ^T\|_2. \end{aligned}$$

□

5 Representation of the Lagrangian invariant subspace

Algorithm 1 computes (A, E) such that $Z = AE^{-1}$ is a good approximation of the low-rank factor of X. In principle, one can now evaluate the matrix product AE^{-1} to get Z. However, this defeats part of the purpose of the inverse-free computation. It is better to consider the application (stabilization, optimal control, etc) where this factor is used, and to see if we can do this next step without inversion. For instance, when using Z to compute a the feedback matrix in stabilization or linear-quadratic optimal control, then we need to compute the product $B^T ZZ^T$ and the formulation $B^T AE^{-1}E^{-T}A^T$) may be numerically more stable to evaluate if, for instance $B^T A$ is very small in norm. Similarly, in the context of model reduction by balanced truncation [2], we need the low-rank factors Z_1 and Z_2 of the solutions of two Lyapunov equations, and then we compute a singular value decomposition of $Z_2^T Z_1$. Again, working directly on the product form, for instance using algorithms for product eigenvalue problems [15, 53], is usually more numerically stable.

In the next sections, we show that a special representation of the subspace im $\begin{bmatrix} I \\ X \end{bmatrix}$ can be constructed directly from the pair (A, E), without constructing X or otherwise passing through an explicit inversion.

5.1 A low-rank representation for $\mathcal{S}$

Let $Z \in \mathbb{R}^{n,m}$ be the (approximate) low-rank factor of the solution X of a Lyapunov equation. In the following, we show how to compute, starting from $\mathrm{IF}(Z) = \begin{bmatrix} E \\ A \end{bmatrix}$, matrices $U_1, U_2 \in \mathbb{R}^{n,r}$ (for some $r \leq m$) and $U_3, U_4 \in \mathbb{R}^{n,m}$ such that

$$\mathrm{IF}(X) = \begin{bmatrix} I_n - U_1 U_2^T \\ U_3 U_4^T \end{bmatrix}. \tag{20}$$

1. Let $M = A(E^T E)^{-1}$; then clearly $\mathrm{IF}(M) = \begin{bmatrix} E^T E \\ A \end{bmatrix}$. Given a threshold $\tau > 1$, using the method in Theorem 2.1, compute a permutation matrix $\mathit{\Pi} \in \mathbb{R}^{n+m,n+m}$ and $N \in \mathbb{R}^{n,m}$ such that $\mathrm{IF}(M) = \mathit{\Pi}^T \begin{bmatrix} I_m \\ N \end{bmatrix}$.

2. Using the method in the proof of Theorem 2.3, compute $\widehat{\mathit{\Pi}} \in \mathbb{R}^{n+m,n+m}$ and $\widehat{N} = D_2 N^T D_1 \in \mathbb{R}^{m,n}$ such that $\mathrm{IF}(M^T) = \widehat{\mathit{\Pi}}^T \begin{bmatrix} I_n \\ \widehat{N} \end{bmatrix}$.

3. Let $\widehat{E} \in \mathbb{R}^{n,n}$ and $\widehat{A} \in \mathbb{R}^{m,n}$ such that $\widehat{\mathit{\Pi}}^T \begin{bmatrix} I_n \\ \widehat{N} \end{bmatrix} = \begin{bmatrix} \widehat{E} \\ \widehat{A} \end{bmatrix}$. Since $\widehat{\mathit{\Pi}}^T$ acts by permuting rows, there are at most $r \leq m$ rows of $\widehat{E}$ which are not rows of I_n. This means that there exists a permutation matrix $\widetilde{\mathit{\Pi}}$ such that $E\widetilde{\mathit{\Pi}}$ and I_n differ only in these r rows. Hence, $I_n - E\widetilde{\mathit{\Pi}}$ is a rank-r matrix, which we can factor as $U_1 U_2^T$, where $U_1, U_2 \in \mathbb{R}^{n,r}$. In particular, we can choose U_1 to be a submatrix of I_n and U_2 with elements of magnitude at most $\tau + 1$. Note that U_1, U_2 and $\widetilde{\mathit{\Pi}}$ can be computed from $\widehat{N}$ and (a compact representation of) $\widehat{\mathit{\Pi}}$ only, without forming full $n \times n$ matrices.

4. We have

$$\begin{aligned} X &= ZZ^T = AE^{-1}E^{-T}A^T = AN^T \\ &= A\widehat{A}\widehat{E}^{-1} = A\widehat{A}\widetilde{\mathit{\Pi}}\left(\widehat{E}\widetilde{\mathit{\Pi}}\right)^{-1} = A\widehat{A}\widetilde{\mathit{\Pi}}\left(I_n - U_1 U_2^T\right)^{-1}, \end{aligned}$$

   hence it suffices to set $U_3 = A \in \mathbb{R}^{n,m}$ and $U_4 = (\widehat{A}\widetilde{\mathit{\Pi}})^T \in \mathbb{R}^{n,m}$ to obtain (20).

The cost of the whole algorithm is $O(nm^2 \frac{\log m}{\log \tau})$, which is linear in n, hence this construction is feasible in the large-scale, low-rank case, i.e. if m is small.

5.2 Conditioning of the special right-looking representation

In view of the discussion in Section 2, to make sure that our special right-looking representation (20) is a well-conditioned basis for its column space, we need to bound its condition number. For this we start with a lemma.

Lemma 5.1. *Let $A, \widehat{A}^T \in \mathbb{R}^{n,m}$, and two invertible matrices $E \in \mathbb{R}^{m,m}$, $\widehat{E} \in \mathbb{R}^{n,n}$ be given, such that*

$$E^{-1}E^{-T}A^T = \widehat{A}\widehat{E}^{-1}.$$

Suppose that

$$\sigma_{\min}\left(\begin{bmatrix}\widehat{E}\\ \widehat{A}\end{bmatrix}\right) = \widehat{\sigma}, \quad \sigma_{\min}\left(\begin{bmatrix}E\\ A\end{bmatrix}\right) = \sigma.$$

Then,

$$\sigma_{\min}\left(\begin{bmatrix}\widehat{E}\\ A\widehat{A}\end{bmatrix}\right) \geq \frac{\sigma\widehat{\sigma}}{\left(\max\left(1,\sigma^2\right)+\frac{1}{2}\right)^{1/2}}.$$

Proof. Let v be any vector with $\|v\| = 1$. By the properties of the smallest singular value, we have

$$\|\widehat{E}v\|^2 + \|\widehat{A}v\|^2 \geq \widehat{\sigma}^2.$$

Similarly, working this time with the vector $\widehat{A}v$, we have

$$\|A\widehat{A}v\|^2 + \|E\widehat{A}v\|^2 \geq \sigma^2\|\widehat{A}v\|^2.$$

Moreover, we have

$$\|E\widehat{A}v\|^2 = \|v^T\widehat{A}^TE^TE\widehat{A}v\| = \|v^T\widehat{A}^TA^T\widehat{E}v\| \leq \|A\widehat{A}v\|\|\widehat{E}v\| \leq \frac{1}{2}\left(\|A\widehat{A}v\|^2 + \|\widehat{E}v\|^2\right).$$

Combining the three inequalities, we get

$$\frac{3}{2}\|A\widehat{A}v\|^2 + \left(\sigma^2+\frac{1}{2}\right)\|\widehat{E}v\|^2 \geq \sigma^2\widehat{\sigma}^2,$$

from which the assertion follows. □

Using Lemma 5.1, we obtain the following result.

Theorem 5.2. *For the special low-rank representation $U = \begin{bmatrix} I - U_1U_2^T \\ U_3U_4 \end{bmatrix}$ constructed in Section 5.1, we have*

$$\kappa(U) \leq \frac{\sqrt{3}}{\sqrt{2}}(mn\tau^2 + n\tau).$$

Proof. We obtain the assertion directly from the definition $\kappa(U) = \frac{\sigma_{\max}(U)}{\sigma_{\min}(U)}$. It is easy to bound the largest singular value as $\sigma_{\max}(U) \leq mn\tau^2 + n\tau$. To bound the smallest singular value, we use Lemma 5.1. The matrix $\begin{bmatrix}E^T & A^T\end{bmatrix}^T$ is orthogonal, hence $\sigma = 1$, while $\begin{bmatrix}\widehat{E}^T & \widehat{A}^T\end{bmatrix}^T$ contains I_n as a submatrix and thus $\widehat{\sigma} \geq 1$. The matrix $\widetilde{\Pi}$ is orthogonal and does not change the singular values, hence $\sigma_{\min}(U) \geq \frac{\sqrt{2}}{\sqrt{3}}$. □

6 Subspaces and residuals

To check the accuracy of a computed solution $X = ZZ^T$ to the Riccati equation (2) (or the Lyapunov equation (1), i.e., when $G = 0$), the standard choice is to compute the relative residual

$$K_e = \frac{\|F^T X + XF + W - XGX\|}{\|F^T\|\|X\| + \|X\|\|F\| + \|W\| + \|X\|^2\|G\|},$$

where the norm is either the Frobenius or the spectral norm. In the typical case in which W factored as $W = C^T C$ and $G = BR^{-1}B^T$ have low rank, in [9] an efficient method is presented to compute the numerator in the relative residual in cost $O(nm^2)$, without assembling the full $n \times n$ matrix, so this computation is feasible even in the large-scale case. The method relies on the identity

$$F^T X + XF + W - XGX = \mathcal{R}\mathcal{L}\mathcal{R}^T,\ \mathcal{R} = \begin{bmatrix} F^T Z & Z & C^T & XB \end{bmatrix},\ \mathcal{L} = \begin{bmatrix} 0 & I & 0 & 0 \\ I & 0 & 0 & 0 \\ 0 & 0 & I & 0 \\ 0 & 0 & 0 & -I \end{bmatrix},$$

where the matrix $\mathcal{R}$ is tall and thin, so one can compute its QR-factorization $\mathcal{R} = QT$, and ignore the Q factors, since the norms we are interested in are orthogonally invariant. Hence

$$\|F^T X + XF + W - XGX\| = \left\|T\mathcal{L}T^T\right\|. \tag{21}$$

In view of our goal to avoid the computation of the Riccati solution but of rather using the Lagrangian invariant subspace, an alternative in our setting is to compute the residual of a representation of the invariant subspace of the Hamiltonian matrix $\mathcal{H}$, see [37], i.e., for an orthonormal basis matrix $Q = \mathrm{IF}(X) \in \mathbb{R}^{2n,n}$, we can compute the quantity

$$K_s = \frac{\left\|\mathcal{H}Q - Q(Q^T\mathcal{H}Q)\right\|}{\|\mathcal{H}\|}, \tag{22}$$

which is exactly zero when Q spans an exact invariant subspace (hence the matrix X obtained from this subspace is a solution of the Riccati equation). This gives a direct measure of the quality of the invariant subspace Q, however, in the large-scale case, this is again unfeasible, as we would need an orthogonal basis.

Once again, inverse-free representations provide an alternative. Given any right-looking representation $X = AE^{-1}$, we can compute without inversions a generalized residual as

$$\begin{aligned} K_g &= \frac{\|E^T F^T A + A^T FE + E^T WE - A^T GA\|}{\|E^T\|\|F^T\|\|A\| + \|A^T\|\|F\|\|E\| + \|E^T\|\|W\|\|E\| + \|A^T\|\|G\|\|A\|} \\ &= \frac{\left\|\begin{bmatrix} E^T & A^T \end{bmatrix} \mathcal{J}_{2n}\mathcal{H} \begin{bmatrix} E \\ A \end{bmatrix}\right\|}{\begin{bmatrix} \|E^T\| & \|A^T\| \end{bmatrix} \begin{bmatrix} \|F\| & \|G\| \\ \|W\| & \|F^T\| \end{bmatrix} \begin{bmatrix} \|E\| \\ \|A\| \end{bmatrix}}. \end{aligned}$$

Note that this residual coincides with K_e when $\begin{bmatrix} E^T \\ A^T \end{bmatrix}^T = \begin{bmatrix} I \\ X^T \end{bmatrix}^T$, and is zero for a Riccati solution, since its numerator equals $E^T(F^TX + XF + W - XGX)E$. If the right-looking representation has the special form (20), then a trick analogous to the one in (21) can be used to compute this numerator. We first determine

$$E^TF^TA+A^TFE+E^TWE-A^TGA = \hat{\mathcal{R}}\mathcal{L}\hat{\mathcal{R}}^T, \quad \hat{\mathcal{R}} = \begin{bmatrix} E^TF^TU_3 & U_4 & E^TC^T & U_3U_4^TB \end{bmatrix},$$

and continue as before in the non-inverse-free case.

To see, how good K_g is as an estimate of K_s, we perform a QR-factorization

$$\begin{bmatrix} E \\ A \end{bmatrix} = QR.$$

Since X is symmetric and the subspace is invariant, it is well-known see [28, 39] that the subspace spanned by Q is Lagrangian and thus

$$\begin{bmatrix} Q & \mathcal{J}_{2n}Q \end{bmatrix}$$

is an *orthogonal symplectic* matrix. Thus, in particular,

$$I_{2n} - QQ^T = -\mathcal{J}_{2n}QQ^T\mathcal{J}_{2n},$$

and therefore,

$$(I_{2n} - QQ^T)\mathcal{H}Q = -\mathcal{J}_{2n}QR^{-T}\begin{bmatrix} E^T & A^T \end{bmatrix}\mathcal{J}_{2n}\mathcal{H}\begin{bmatrix} E \\ A \end{bmatrix}R^{-1}.$$

Using the orthogonality, and denoting by N_s and N_g the numerators of K_s and K_g, respectively, it follows that $N_s \leq N_g\|R^{-1}\|^2$. Including the denominators as well, we obtain

$$K_s \leq 4\left(\|R^{-1}\|\|R\|\right)^2 K_g = 4\kappa\left(\begin{bmatrix} E \\ A \end{bmatrix}\right)K_g,$$

where the (non-important) factor 4 arises from the nested norms in the denominator. As a consequence, for a basis matrix with bounded condition number, small K_g implies small K_s, and hence the subspace is a good approximation to the exact invariant subspace. If, however, $\|X\|$ is large, then $\kappa\left(\begin{bmatrix} I & X^T \end{bmatrix}^T\right)$ is large as well, and hence the usual Riccati residual K_e may not be a good measure.

7 Partial error analysis

A complete error analysis of the LR-ADI method is not available in the literature, and it looks like a daunting task. Here we present a small part of the analysis that allows a better understanding under which conditions our inverse-free approach will perform better than the classical approach. For this, we simplify our analysis in several ways using the symbol $a \dot{\leq} b$ to denote $a \leq p(m)b + O(\mathbf{u}^2)$, where $p(m)$ is a polynomial in the dimension m and $\mathbf{u}$ is the machine precision.

- We ignore second-order terms in the perturbation and terms dependent on the low-rank subspace dimension m (but not those depending on the full subspace dimension n, which we assume to be significantly larger).
- We focus only on the accuracy of the computation of V_{k+1}. We denote an appropriate orthogonal basis for the computed subspace in finite precision by $\tilde{V}_{k+1}$. Note that this orthonormal basis does not depend on the sequence of low-rank factors $\{Z_i\}$.
- With this in mind, we can reduce the analysis to the computation of the map $V_k \mapsto \tilde{V}_{k+1}$. Indeed, the computational strategies that we are comparing (standard and inverse-free ADI) can be represented by taking two different routes in the following diagram:

$$\begin{array}{ccccccccc} V_0 & \to & V_1 & \to & V_2 & \to & \dots & \to & V_{k-1} & \to & V_k \\ \downarrow & & & & & & & & & & \downarrow \\ \tilde{V}_0 & \to & \tilde{V}_1 & \to & \tilde{V}_2 & \to & \dots & \to & \tilde{V}_{k-1} & \to & \tilde{V}_k \end{array}$$

  If we can prove that the computational error in taking one single step is higher if one takes the upper road $V_{i-1} \to V_i \to \tilde{V}_i$ rather than the lower road $V_{i-1} \to \tilde{V}_{i-1} \to \tilde{V}_i$, then the same holds (at least in first order approximation) for the whole computation.

To achieve a partial error analysis we use the following bounds.

Lemma 7.1 ([49])**.** *Let $A = QR$ and $A + \delta A = (Q + \delta Q)(R + \delta R)$ be QR-decompositions, then*

$$\|\delta Q\|_F \dot{\leq} \|\delta A\|_F \|R^{-1}\|_2.$$

Lemma 7.2 (Follows from [18, Section 19.3])**.** *Let $\hat{Q}$ be the computed QR-factor of a matrix A. Then,*

$$\|Q - \hat{Q}\|_F \dot{\leq} cn^{3/2} \|A\|_F \|R^{-1}\|_2.$$

for a small integer constant c.

In some of our analysis we will compare orthogonal matrices $\hat{Q}$ and Q that are not close to each other, but span the same subspace. For this, it will be useful to convert the bound of Lemmas 7.1 and 7.2 to bounds in terms of a *subspace distance*. The distance between two subspaces [16] is defined as $d(Q, \hat{Q}) = \|P_Q - P_{\hat{Q}}\|_2$, where $P_Q = QQ^T$ is the orthogonal projector on the subspace $\operatorname{im} Q$ and we have the following result.

Lemma 7.3. *Let $Q, Q + \delta Q \in \mathbb{R}^{n,m}$ be orthogonal. Then,*

$$d(Q, Q + \delta Q) \leq \|\delta Q\|_2 \leq \sqrt{m} \|\delta Q\|_F.$$

Proof. Let Q_2 be chosen such that $\begin{bmatrix} Q & Q_2 \end{bmatrix}$ is a square, orthogonal matrix. It is known [16, Theorem 2.6.1] that $d(Q, Q+\delta Q) = \|Q_2^T(Q+\delta Q)\|_2$. Hence,

$$d(Q, Q+\delta Q) = \|Q_2^T(Q+\delta Q)\|_2 = \|Q_2^T(\delta Q)\|_2 \leq \|\delta Q\|_2.$$

□

For our analysis we focus on computing $W = \tilde{V}_{i+1}$ from $V = V_i$. For simplicity, let us drop the subscript i and consider the operation in line 4 of Algorithm 1 as a single matrix product $V \leftarrow DV$, with

$$D := \sqrt{\frac{\operatorname{Re} p_i}{\operatorname{Re} p_{i-1}}} \left(I - (p_i + \overline{p_{i-1}})(F^T + p_i I)^{-1} \right).$$

We first show that the inverses of several triangular factors in QR-decompositions are available explicitly to use in Lemma 7.1. Consider the subspace representation $\tilde{V} = [E^T \quad A^T]^T$ of $[I \quad V^T]^T$ which we obtain from the QR-decomposition

$$\begin{bmatrix} E \\ A \end{bmatrix} T = \begin{bmatrix} I \\ V \end{bmatrix},$$

implying that $E = T^{-1}$. Consider now the QR-decomposition of $[E^T \quad (DA)^T]^T$ and let Y be the inverse of the triangular factor. Then, with this choice, $[(EY)^T \quad (DAY)^T]^T$ is orthogonal, and E, Y are upper triangular, hence EY is the inverse of the triangular factor in the QR-decomposition of $[I \quad (DV)^T]^T$.

We are now ready to estimate the computational error in the two procedures. The error η_R in the computation of the Riccati solution comes from two sources, the numerical error in the product DV and the numerical error in the orthogonalization of $[I \quad (DV)]^T$.

Since we are only performing a first-order error analysis, we can consider both errors separately and then sum up their contributions, see [18, Section 3.8].

For the error in computing DV we can write the computed product as $DV + \Delta_{DV}$ and have the bound $\|\Delta_{DV}\|_F \leq \gamma \|D\|_F \|V\|_F$ for an unspecified constant γ. If this were an ordinary product between dense matrices stored in memory, one would have $\gamma = n$, but in fact the expression of D involves a sparse system solution, multiplications and a subtraction. Hence, the true value of c is complicated to estimate. Nevertheless, we may expect it to scale, as stated, with the norms of the matrices. The orthogonal matrix $\tilde{W}_1$ computed with this error then satisfies (using Lemmas 7.1 and 7.3)

$$d(W, \tilde{W}_1) \dot{\leq} \|W - \tilde{W}_1\|_F \dot{\leq} \Delta_{DV} \|EY\|_2 \leq \gamma \|D\|_F \|V\|_F \|EY\|_2.$$

For the error in the last orthogonalization, the matrix $\tilde{W}_2$ produced with machine precision orthogonalization satisfies

$$d(W, \tilde{W}_2) \dot{\leq} \|W - \tilde{W}_2\|_F \dot{\leq} cn^{3/2} \left\| \begin{bmatrix} I \\ DV \end{bmatrix} \right\|_F \|EY\|_2 = cn^{3/2} \sqrt{\|DV\|_F^2 + n}\, \|EY\|_2,$$

where we have used Lemmas 7.2 and 7.3.

In contrast to this analysis, taking the route of using first an orthogonalization and then proceeding has three possible sources of error, the numerical error in the first orthogonalization producing $\tilde{V}$, the numerical error in the product DA, and the numerical error in the last orthogonalization step producing W. Denoting the sum of these three individual errors by η_O we have the following terms.

The first orthogonalization produces $\tilde{W}_3$ with an individual error that can be estimated via

$$d(W, \tilde{W}_3) \dot{\leq} \|D\|_F c n^{3/2} \sqrt{\|V\|_F^2 + n}\ \|E\|_2 \|Y\|_2. \tag{23}$$

The individual error in the product DA produces a $\tilde{W}_4$ satisfying

$$d(W, \tilde{W}_4) \dot{\leq} \gamma \|D\|_F \|A\|_F \|Y\|_2 \dot{\leq} \gamma \|D\|_F \|Y\|_2,$$

where we have used that $\|A\|_F \leq \sqrt{m}$, because A is a submatrix of a matrix with orthonormal columns. The final orthogonalization step produces a $\tilde{W}_5$ with an individual error satisfying

$$d(W, \tilde{W}_5) \dot{\leq} c n^{3/2} \left\| \begin{bmatrix} E \\ DA \end{bmatrix} \right\|_F \|Y\|_2 \leq c n^{3/2} \sqrt{\|D\|_F^2 + n} \|Y\|_2$$

Thus, we have the following two estimates for the errors in the two alternative routes.

$$\eta_R = \gamma \|D\|_F \|V\|_F \|EY\|_2 + c n^{3/2} \sqrt{\|DV\|_F^2 + n}\ \|EY\|_2, \tag{24}$$

$$\begin{aligned} \eta_O = {} & \|D\|_F c n^{3/2} \sqrt{\|V\|_F^2 + n}\ \|E\|_2 \|Y\|_2 \\ & + \gamma \|D\|_F \|Y\|_2 + c n^{3/2} \sqrt{\|D\|_F^2 + n}\ \|Y\|_2. \end{aligned} \tag{25}$$

Assuming that $\|D\|_F$, $\|V\|_F$ are not significantly smaller than n, the two estimates are essentially equivalent apart from a factor $\|EY\|_2 \leq \|E\|_2 \|Y\|_2 \leq \|Y\|_2$.

However, looking at the two estimates, at first sight it seems that the new strategy of performing first an orthogonalization is never better than the procedure used in the standard LR-ADI method, and that η_R would be even be superior to η_O, if the orthogonalization of $[I \quad (DV)^T]^T$, which accounts for the factor $\|EY\|_2$, happens to be better-conditioned than that of $[E^T \quad (DA)^T]^T$, which accounts for the factor $\|Y\|_2$. This could happen due to subtractive cancelation for instance if $\|DV\|$ happens to be significantly smaller than $\|D\|_F \|V\|_F$. However, performing several numerical experiments, it seems that the bound η_R is essentially accurate, while the η_O, and more precisely, (23) is often a large overestimate.

Let us first give an intuitive argument of why (23) often overestimates the error. The computed orthogonal basis matrix $U = \tilde{V}$ is affected by an error $\delta U = \begin{bmatrix} \delta E \\ \delta A \end{bmatrix}$ of magnitude $\|\delta U\|_F \dot{\leq} c n^{3/2} \sqrt{\|V\|_F^2 + n}$. The subspace $U + \delta U$ is then multiplied with a matrix $\hat{D} = \mathrm{diag}(I_m, D)$. The bound (23) considers the resulting error $\hat{D}\delta U$ as a generic

error of norm $\|\hat{D}\|_F \|\delta U\|_F$. This is, however, not entirely true, as both $\hat{D}U$ and $\hat{D}\delta U$, in the generic case, will be approximately associated with the singular vectors of $\hat{D}$ of the dominant singular values. Consider for instance the extreme case in which $\hat{D}$ has rank m. In this case, no matter how large δU is, the resulting subspace will be still $\mathrm{im}(\hat{D})$.

We can turn this intuition into a more formal bound, using a probabilistic analysis.

Theorem 7.4. *Let $U \in \mathbb{R}^{n\times r}$ be a random matrix with Gaussian distributed entries, and let $\hat{D} \in \mathbb{R}^{n\times n}$ be a given matrix with singular values $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_n$. Let $U + \delta U$ be a small perturbation of U and let $\hat{D}U = QR$ and $\hat{D}(U + \delta U) = \tilde{Q}\tilde{R}$ be QR-decompositions of these matrices, respectively. Then*

$$\mathbb{E}\left[\frac{d(Q,\tilde{Q})}{\|\delta U\|_F \|\tilde{R}^{-1}\|_2}\right] \leq \sqrt{1+r}\left(\sum_{j>r-1}\sigma_j^2\right)^{1/2}. \tag{26}$$

Proof. The main part of the proof follows by applying [17, Theorem 10.5], which for the special choice $p = 2$ and the notation adapted to our case states that

$$\mathbb{E}\left[\|(I_n - QQ^T)\hat{D}\|_F\right] \leq \sqrt{1+r}\left(\sum_{j>r-1}\sigma_j^2\right)^{1/2}. \tag{27}$$

Let Q_2 be a matrix such that $[Q \quad Q_2]$ is square orthogonal. Then we have

$$\begin{aligned} d(Q,\tilde{Q}) &= \|Q_2^T\tilde{Q}\|_2 = \|Q_2Q_2^T\tilde{Q}\|_2 = \|(I - QQ^T)\tilde{Q}\|_2 = \|(I - QQ^T)\hat{D}(U + \delta U)\tilde{R}^{-1}\|_2 \\ &= \|(I - QQ^T)\hat{D}(\delta U)\tilde{R}^{-1}\|_2 \leq \|(I - QQ^T)\hat{D}\|_F \|\delta U\|_F \|\tilde{R}^{-1}\|_2. \end{aligned}$$

Combining this inequality with (27) then gives (26). □

Theorem 7.4 states that, for generic values of V, we can replace the factor $\|D\|_F$ with $\sqrt{1+r}\left(\sum_{j>r-1}\sigma_j^2\right)^{1/2}$, i.e., instead of summing the squares of all the singular values of $\hat{D}$, we start from the $(r-1)$st. Hence the modified version of (25) does not contain anymore terms that depend on the product $\|D\|_F\|V\|_F$. This reduction typically is more pronounced than the one resulting from $\|Y\|_2$ instead of $\|EY\|_2$. Of course, this intuitive analysis is far from being rigorous (mainly because U is not random in our setting, but comes from the previous steps of the algorithm), but it captures the numerical behavior of the method in many cases.

8 Numerical experiments

In this section we present some numerical experiments. We start with a standard example, to show that there is little difference between the two algorithms in normal conditions, with bounded X and well-conditioned matrices such as those coming from finite-difference discretizations on regular grids, often used in ADI benchmarks [43].

We take the demo problem `demo_l1` of Lyapack [44], with a Lyapunov equation resulting from the discretization of a 2D convection-diffusion equation on a square grid. In detail, we consider a 25×25 finite-difference discretization of the operator $\Delta(u) - 10x\frac{d}{dx} - 100y\frac{d}{dy}$ and of the right-hand side function

$$g(x) = \begin{cases} 1 & 0.1 \leq x \leq 0.3, \\ 0 & \text{otherwise.} \end{cases}$$

All parameters are unchanged with respect to the demo; in detail, we use $\sqrt{\mathbf{u}}$, the square root of the machine precision, as column compression tolerance, and 15 shift values computed using 50 Ritz values for F and 25 for F^{-1}.

We report in Figure 1 the residual obtained at each iteration (using the different measures K_e, K_s, K_g introduced in Section 6) and the number of columns in the approximation Z at each ADI step. For this problem, the norm of the solution is moderate ($\|Z\| \approx 0.27$) and there is basically no difference between the results of the two algorithms. The only notable difference is that the inverse-free method attains a slightly lower accuracy in the equation residual K_e (although still below 10^{-15}).

As a test with a significantly more ill-conditioned matrix, we created a sparse symmetric, negative-definite matrix with the following MATLAB commands

```
n=400;
reset(RandStream.getGlobalStream);
F=-sprandsym(n,4/n,1e-10,1);
```

We chose 400×400 as the size of F to ensure that the subspace residual to be computed in a reasonable time. In our experiments, a similar behavior arises also for larger sizes. The matrix F has condition number $\approx 1.5 \times 10^{10}$ and norm $\approx 3 \times 10^{0}$.

We tested Lyapunov equations with this matrix and three different right-hand sides:

R1 a random right-hand side, generated with `C=randn(2,n);`

R2 a right-hand side that has a norm approximately equal to the smallest eigenvalues of F; in this way, the pair (F, B) is close to uncontrollable and large entries appear in Z starting from the very first iteration. We generated the coefficients with

```
[V D]=eigs(A,4,'sm');
mu=1e-6;
V=V+mu*randn(size(V));
C=V';
```

R3 the same as R2, but with `mu=1e-9`.

The other parameters are the same as in the other cases, with the exception of the column compression tolerance which has been set to 10^{-12}. The approximate low-rank factor Z of the solution has norm $\approx 10^{5}$ in all three cases. We depict the results in Figures 2, 3 and 4.

Figure 1: Comparison of ADI and PG-ADI, finite-difference discretization

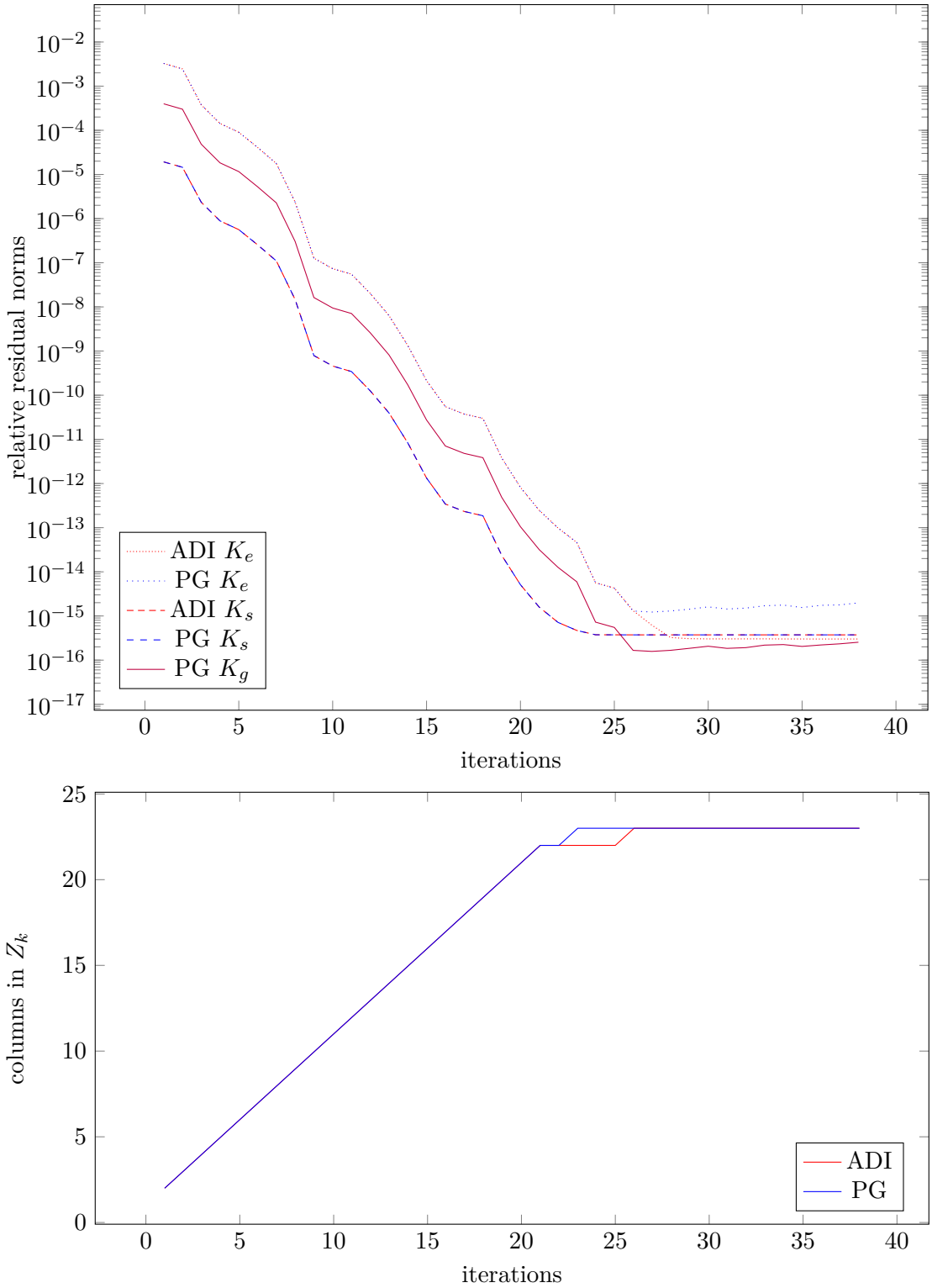


Figure 2: Comparison of ADI and PG-ADI, random matrix and right-hand side R1

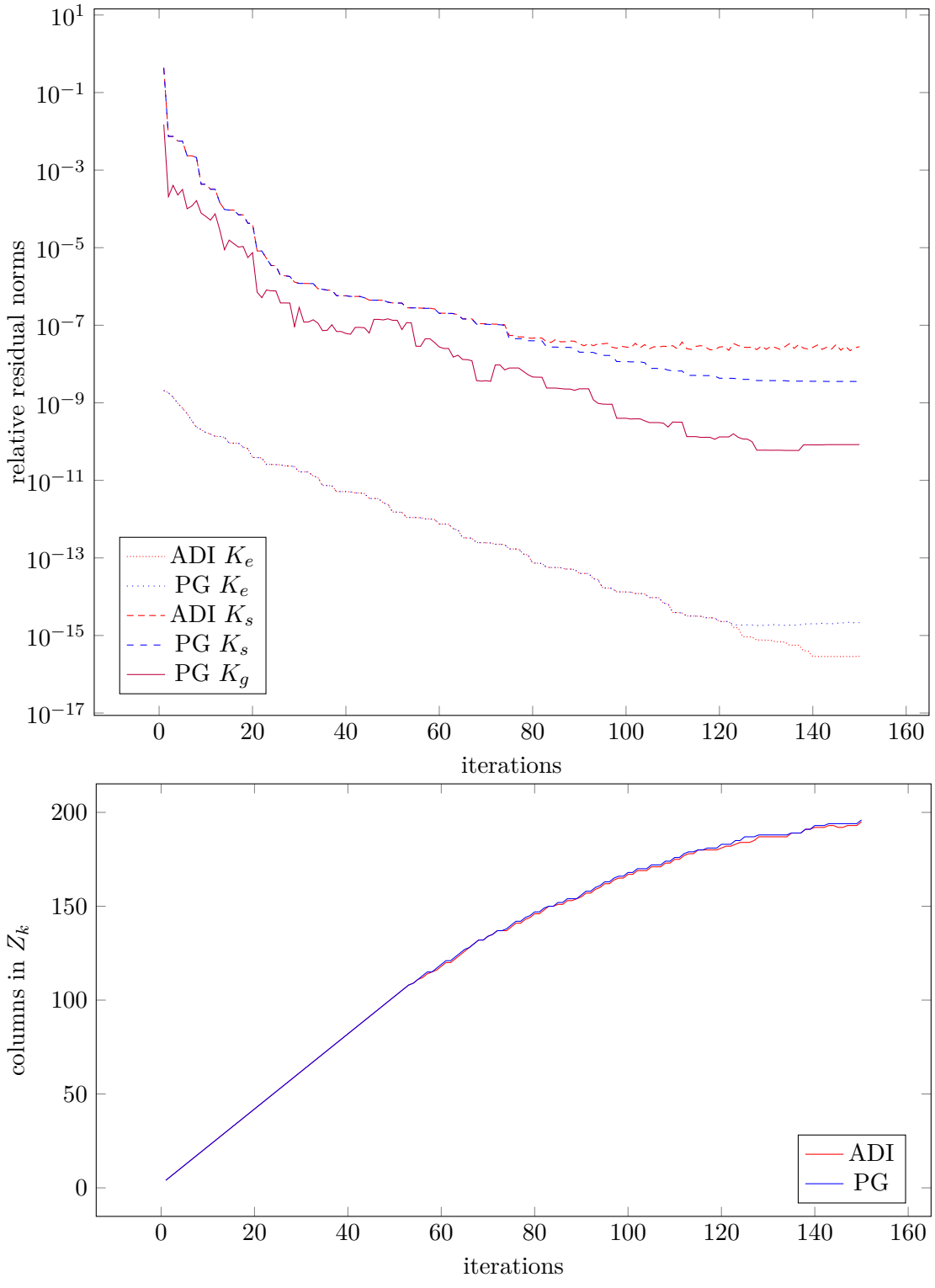


Figure 3: Comparison of ADI and PG-ADI, random matrix, RHS from eigs with $\mu = 10^{-6}$

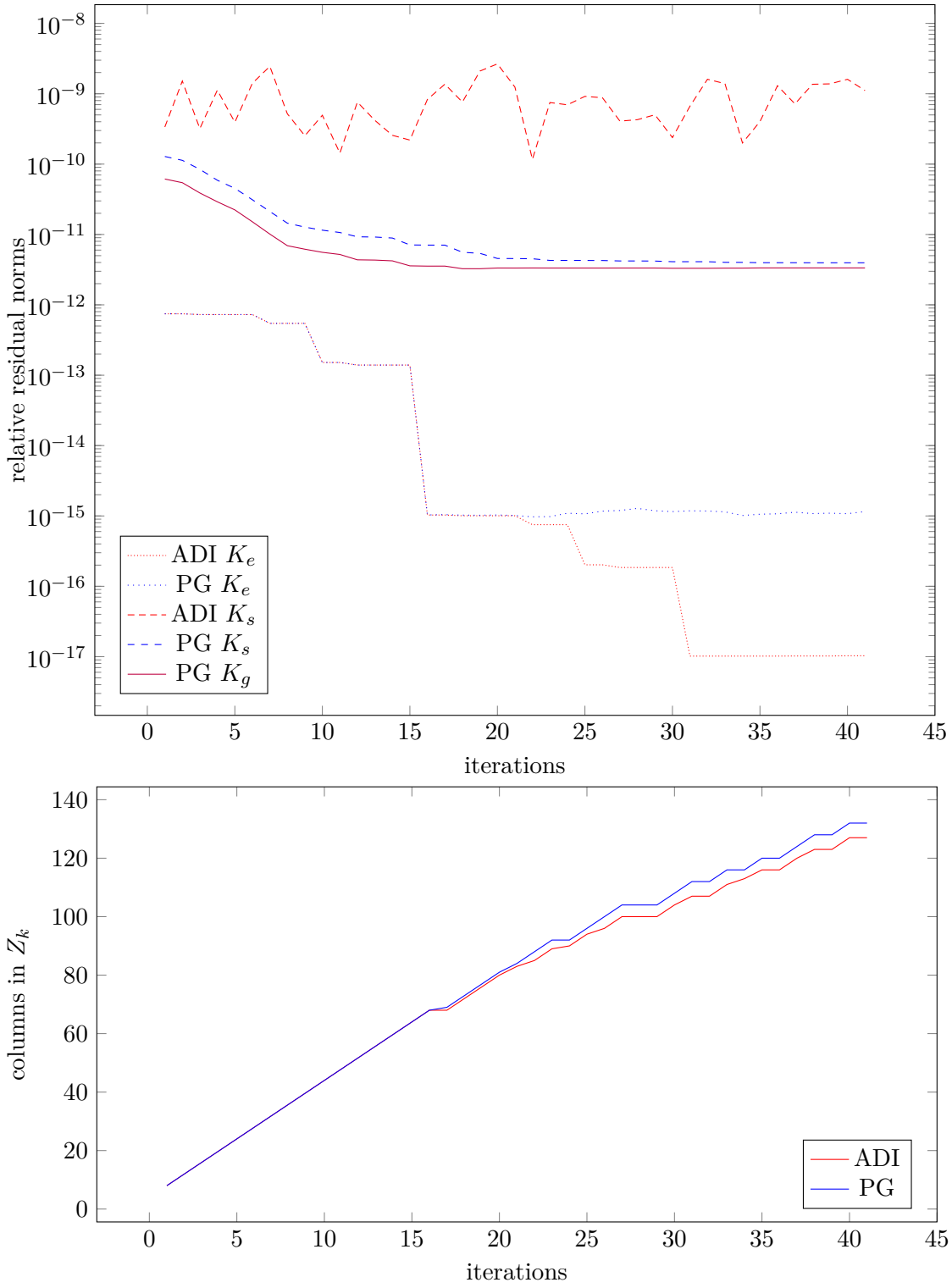
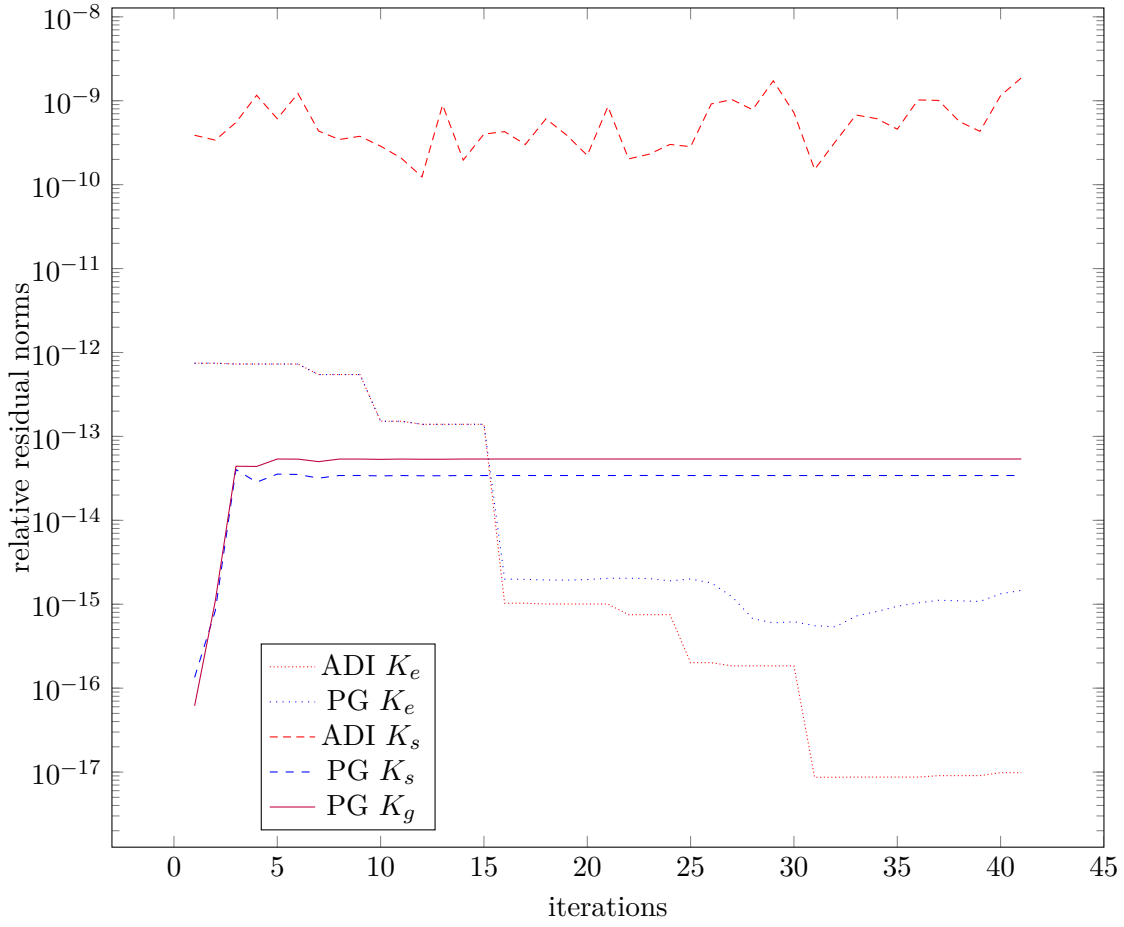
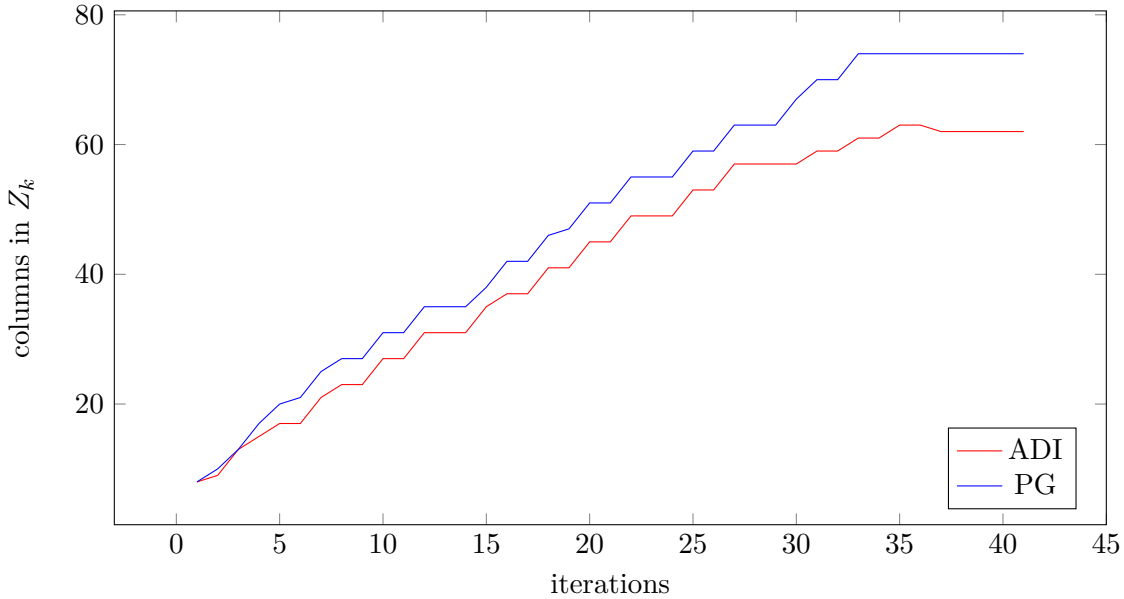


Figure 4: Comparison of ADI and PG-ADI, random matrix, RHS from eigs with $\mu = 10^{-9}$





The experiments show that the permuted graph variant of LR-ADI computes a more accurate basis for the invariant subspace associated with $X = ZZ^T$, at the expense of dropping fewer columns of Z in the column compression. Indeed, the used compression step drops only components of magnitude smaller than $O(\varepsilon)$, not $O(\|Z\|\varepsilon)$. Nevertheless, it is not only a question of using a different tolerance, the new algorithm gets better residuals because it is immune from element growth in Z, and thus the same quantities can be computed more accurately.

Another interesting observation is that the generalized residual K_g in general achieves better results as an estimator of K_s rather than the Riccati residual.

9 Conclusions

In this paper, we have presented an inverse-free version of the low-rank ADI algorithm. We have shown that this technique, that was so far only used in iterative solvers for small-scale dense Lyapunov and Riccati equations can be successfully employed also in medium and large scale applications. Permuted graph bases allow the efficient representation of sparse matrices and subspaces. While an expression based on a low rank representation of graph subspace is perfectly manageable with low-rank arithmetic, an orthonormal basis for the same subspace would in general be highly impractical.

The residual heuristic measure K_g suggested in Section 6 shows a numerical behavior which is more similar to the real subspace residual K_s than the usual matrix equation residual K_e, and it is possibly useful also independently from the remainder of the algorithm.

Inverse-free algorithms for the basic matrix operations (sum, products, inverses) appeared in [6]. As a useful by-product of our analysis we have shown how to perform several new nontrivial tasks using these matrix representations: horizontal stacking, column compression, and converting between left- and right-looking representations in a large-scale low-rank context. The obtained results show that inverse-free arithmetic is a viable tool for dealing with large low-rank matrices, and opens the way to new possible uses for this technique.

References

[1] H. Abou-Kandil, G. Freiling, V. Ionescu, and G. Jank. *Matrix Riccati Equations in Control and Systems Theory.* Birkhäuser, Heidelberg, 2003.

[2] A. C. Antoulas. *Approximation of large-scale dynamical systems*, volume 6 of *Advances in Design and Control.* Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 2005.

[3] A. C. Antoulas, D. C. Sorensen, and Y. Zhou. On the decay rate of Hankel singular values and related issues. *Systems Control Lett.*, 46(5):323–342, 2002.

[4] P. Benner and H. Faßbender. On the numerical solution of large-scale sparse discrete-time Riccati equations. *Adv. Computational Math.*, 35:119–147, 2011.

[5] P. Benner, H. Faßbender, and M. Stoll. A Hamiltonian Krylov-Schur-type method based on the symplectic Lanczos method. *Linear Algebra Appl.*, 435:578–600, 2011.

[6] P. Benner and R. Byers. An arithmetic for matrix pencils: theory and new algorithms. *Numer. Math.*, 103(4):539–573, 2006.

[7] P. Benner, R. Byers, P. Losse, V. Mehrmann, and H. Xu. Robust formulas for optimal h_∞ controllers. *Automatica*, 47:2639–2646, 2011.

[8] P. Benner, R. Byers, V. Mehrmann, and H. Xu. A robust numerical method for the γ-iteration in H_∞-control. *Linear Algebra Appl.*, 425:548–570, 2007.

[9] 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.

[10] P. Benner, V. Mehrmann, and D. Sorensen (Editors). *Dimension Reduction of Large-Scale Systems*, volume 45 of *LNSCE*. Springer-Verlag, Heidelberg, 2005. ISBN 3-540-24545-6.

[11] P. Benner, V. Mehrmann, V. Sima, S. Van Huffel, and A. Varga. SLICOT - a subroutine library in systems and control theory. *Applied and Computational Control, Signals and Circuits*, 1:505–546, 1999.

[12] 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*, 6:32–52, 2013.

[13] D. Chu, X. Liu, and V. Mehrmann. A numerical method for computing the Hamiltonian Schur form. *Numer. Math.*, 105(3):375–412, 2007.

[14] G. Freiling, V. Mehrmann, and H. Xu. Existence, uniqueness and parametrization of Lagrangian invariant subspaces. *SIAM J. Matrix Anal. Appl.*, 23:1045–1069, 2002.

[15] Gene Golub, Knut Sølna, and Paul Van Dooren. Computing the SVD of a general matrix product/quotient. *SIAM J. Matrix Anal. Appl.*, 22(1):1–19 (electronic), 2000.

[16] G.H. Golub and C.F. Van Loan. *Matrix computations*. Johns Hopkins Studies in the Mathematical Sciences. Johns Hopkins University Press, Baltimore, MD, third edition, 1996.

[17] N. Halko, P. G. Martinsson, and J. A. Tropp. Finding structure with randomness: probabilistic algorithms for constructing approximate matrix decompositions. *SIAM Rev.*, 53(2):217–288, 2011.

[18] N.J. Higham. *Accuracy and stability of numerical algorithms*. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, second edition, 2002.

[19] D. Hinrichsen and A. J. Pritchard. *Mathematical Systems Theory I. Modelling, State Space Analysis, Stability and Robustness*. Springer-Verlag, New York, NY, 2005.

[20] D.Y. Hu and R. Reichel. Krylov subspace methods for the Sylvester equation. *Linear Algebra Appl.*, 172:283–313, 1992.

[21] I.M. Jaimoukha and E.M. Kasenally. Implicitly restarted Krylov subspace methods for stable partial realizations. *SIAM J. Matrix Anal. Appl.*, 18:633–652, 1997.

[22] T. Kailath. *Systems Theory*. Prentice-Hall, Englewood Cliffs, NJ, 1980.

[23] R.E. Kalman, P.L. Falb, and M.A. Arbib. *Topics in Mathematical System Theory*. McGraw-Hill, New York, 1969.

[24] D. L. Kleinman. On an iterative technique for Riccati equation computations. *IEEE Trans. Automat. Control*, AC-13:114–115, 1968.

[25] H.W. Knobloch and H. Kwakernaak. *Lineare Kontrolltheorie*. Springer-Verlag, Berlin, 1985. in German.

[26] D. E. Knuth. Semioptimal bases for linear dependencies. *Linear and Multilinear Algebra*, 17(1):1–4, 1985.

[27] M.M. Konstantinov, D.W. Gu, V. Mehrmann, and P.Hr. Petkov. *Perturbation Theory for Matrix Equations*. Elsevier, North Holland, 2003.

[28] Daniel Kressner. *Numerical methods for general and structured eigenvalue problems*, volume 46 of *Lecture Notes in Computational Science and Engineering*. Springer-Verlag, Berlin, 2005.

[29] P. Lancaster and L. Rodman. *Algebraic Riccati equations*. Oxford University Press, Oxford, 1995.

[30] P. Lancaster and M. Tismenetsky. *The Theory of Matrices*. Academic Press, Orlando, 2nd edition, 1985.

[31] P. Losse. *The $\mathcal{H}_\infty$ Optimal Control Problem for Descriptor Systems*. Dissertation, Fakultät für Mathematik, Technische Universität Chemnitz, February 2012. Available from `http://nbn-resolving.de/urn:nbn:de:bsz:ch1-qucosa-83628`.

[32] The MathWorks, Inc., Cochituate Place, 24 Prime Park Way, Natick, Mass, 01760. *The MATLAB Control Toolbox, Version 5.0*, 2000.

[33] V. Mehrmann and F. Poloni. Doubling algorithms with permuted Lagrangian graph bases. *SIAM J. Matrix Anal. Appl.*, 33:780–805, 2012. `http://dx.doi.org/10.1137/110850773`.

[34] V. Mehrmann and F. Poloni. Robust control via the computation of permuted graph bases. *Automatica*, 49:1790–1797, 2013. DOI: 10.1016/j.automatica.2013.02.039.

[35] V. Mehrmann, C. Schröder, and V. Simoncini. An implicitly-restarted Krylov subspace method for real symmetric/skew-symmetric eigenproblems. *Linear Algebra Appl.*, 436:4070–4087, 2012. DOI: 10.1016/j.laa.2009.11.009.

[36] V. Mehrmann, C. Schröder, and D. S. Watkins. A new block method for computing the Hamiltonian Schur form. *Linear Algebra Appl.*, 431(3-4):350–368, 2009.

[37] V. Mehrmann and E. Tan. Defect correction methods for the solution of algebraic Riccati equations. *IEEE Trans. Automat. Control*, AC–33:695–698, 1988.

[38] V. Mehrmann and D. Watkins. Structure-preserving methods for computing eigenpairs of large sparse skew-Hamiltonian/Hamiltonian pencils. *SIAM J. Sci. Comput.*, 22:1905–1925, 2001.

[39] V.L. Mehrmann. *The autonomous linear quadratic control problem*, volume 163 of *Lecture Notes in Control and Information Sciences*. Springer-Verlag, Berlin, 1991. Theory and numerical solution.

[40] W. Niethammer and G. Starke. SOR for $AX - XB = C$. *Linear Algebra Appl.*, 154–156:355–375, 1991.

[41] C.C. Paige and C.F. Van Loan. A Schur decomposition for Hamiltonian matrices. *Linear Algebra Appl.*, 14:11–32, 1981.

[42] T. Penzl. A cyclic low-rank Smith method for large sparse Lyapunov equations. *SIAM J. Sci. Comput.*, 21:1401–1418, 1999.

[43] T. Penzl. Eigenvalue decay bounds for solutions of Lyapunov equations: the symmetric case. *Systems Control Letters*, 40:139–144, 2000.

[44] T. Penzl. LYAPACK users' guide: a MATLAB toolbox for large Lyapunov and Riccati equations, model reduction problems, and linear quadratic optimal control problems. Technical report, Technical University Chemnitz, SFB 393, 2000. Version 1.0.

[45] P.H. Petkov, N.D. Christov, and M.M. Konstantinov. *Computational Methods for Linear Control Systems*. Prentice-Hall, Hertfordshire, UK, 1991.

[46] Y. Saad. *Iterative methods for sparse linear systems*. Society of Industrial and Applied Mathematics, Philadelphia, PA, 2nd edition, 2003.

[47] V. Sima. *Algorithms for Linear-Quadratic Optimization*, volume 200 of *Pure and Applied Mathematics*. Marcel Dekker, Inc., New York, NY, 1996.

[48] V. Simoncini, D.B. Szyld, and M. Monsalve. On two numerical methods for the solution of large-scale algebraic Riccati equations. *IMA J. Numer. Anal.*, 2013.

[49] G. W. Stewart and J.-G. Sun. *Matrix Perturbation Theory*. Academic Press, New York, 1990.

[50] G.W. Stewart. Error and perturbation bounds for subspaces associated with certain eigenvalue problems. *SIAM Rev.*, 15:727–764, 1973.

[51] E.L. Wachspress. Iterative solution of the Lyapunov matrix equation. *Appl. Math. Lett.*, 1:87–90, 1988.

[52] D. S. Watkins. On the reduction of a Hamiltonian matrix to Hamiltonian Schur form. *Electron. Trans. Numer. Anal.*, 23:141–157, 2006.

[53] David S. Watkins. Product eigenvalue problems. *SIAM Rev.*, 47(1):3–40, 2005.

[54] K. Zhou, J.C. Doyle, and K. Glover. *Robust and Optimal Control*. Prentice-Hall, Upper Saddle River, NJ, 1995.