MATHICSE Technical Report
Nr. 07.2019
March 2019

Low-rank updates and divide-and-conquer methods for quadratic matrix equations

D. Kressner, P. Kürschner, S. Massei

Low-rank updates and divide-and-conquer methods for quadratic matrix equations

Daniel Kressner* Patrick Kürschner† Stefano Massei‡

Abstract

In this work, we consider two types of large-scale quadratic matrix equations: Continuous-time algebraic Riccati equations, which play a central role in optimal and robust control, and unilateral quadratic matrix equations, which arise from stochastic processes on 2D lattices and vibrating systems. We propose a simple and fast way to update the solution to such matrix equations under low-rank modifications of the coefficients. Based on this procedure, we develop a divide-and-conquer method for quadratic matrix equations with coefficients that feature a specific type of hierarchical low-rank structure, which includes banded matrices. This generalizes earlier work on linear matrix equations. Numerical experiments indicate the advantages of our newly proposed method versus iterative schemes combined with hierarchical low-rank arithmetic.

1 Introduction

This paper is concerned with numerical algorithms for treating two types of quadratic matrix equations with large-scale, data-sparse coefficients.

Type 1: CARE. A *continuous-time algebraic Riccati equation (CARE)* takes the form

$$A^*XE + E^*XA - E^*XFXE + Q = 0, \tag{1}$$

where A, E, F, Q are real $n \times n$ matrices, such that E is invertible and F, Q are symmetric positive semi-definite. Motivated by its central role in robust and optimal control [8, 31, 35, 42, 44], this class of equations has been widely studied in the literature; see, e.g., [6, 11, 15]. A solution X to (1) is called stabilizing if the so called closed-loop matrix $A - FXE$ is stable, that is, all its eigenvalues are contained in the open left half plane. Mild conditions on the coefficients (see, e.g., [15, Sec. 2.2.2]) ensure the existence, uniqueness, and symmetric positive semi-definiteness of such a stabilizing solution X.

We consider the case when n is large and F has low rank, that is, $F = BB^*$ for some matrix $B \in \mathbb{R}^{n \times m}$ with $m \ll n$. This is a common assumption in linear-quadratic optimal control problems, where m corresponds to the number of inputs [10, 11]. However, we do not impose low rank on Q, which allows for having a large number of outputs in control problems, e.g., when observing the

*EPF Lausanne, Switzerland, `daniel.kressner@epfl.ch`
†KU Leuven, Electrical Engineering (ESAT), Kulak Kortrijk Campus, Belgium, `patrick.kurschner@kuleuven.be`
‡EPF Lausanne, Switzerland, `stefano.massei@epfl.ch`

state directly. To simplify the exposition, we will focus the discussion mostly on the case $E = I$; the extension to general invertible E will be explained in Section 3.2.1.

For $F = 0$, the equation (1) becomes linear and is called Lyapunov equation. A low-rank updating procedure for such linear matrix equations has been proposed recently in [30]. In this work, we extend this procedure to CARE. More specifically, assuming that X_0 satisfies a reference CARE

$$A_0^* X_0 + X_0 A_0 - X_0 F_0 X_0 + Q_0 = 0, \tag{2}$$

we aim at computing a correction δX such that $X := X_0 + \delta X$ solves the modified CARE

$$A^* X + XA - XFX + Q = 0, \tag{3}$$

with

$$A := A_0 + \delta A, \quad F := F_0 + \delta F, \quad Q := Q_0 + \delta Q, \quad \text{with } \delta A, \delta F, \, \delta Q \text{ of low rank.}$$

Subtracting (2) from (3) yields

$$(A - FX_0)^* \delta X + \delta X (A - FX_0) - \delta X F \delta X + \widehat{Q} = 0. \tag{4}$$

The modified constant term $\widehat{Q} := \delta Q + \delta A^* X_0 + X_0 \delta A - X_0 \delta F X_0$ satisfies

$$\mathrm{rk}(\widehat{Q}) \leq \mathrm{rk}(\delta Q) + 2\mathrm{rk}(\delta A) + \mathrm{rk}(\delta F),$$

where $\mathrm{rk}(\cdot)$ denotes the rank of a matrix. Hence, independently of the rank of Q_0, the constant term of the CARE (4) is guaranteed to have low rank. Note that most algorithms for large-scale Riccati equations [7, 10, 11, 46, 47] assume the constant term to be of low rank which, in turn, may render them unsuitable for solving (3). In contrast, the formulation (4) is well suited for such methods, returning an approximation of δX in the form of a symmetric low-rank factorization.

Type 2: UQME. A *unilateral quadratic matrix equation (UQME)* takes the form

$$AX^2 + BX + C = 0, \tag{5}$$

with $A, B, C \in \mathbb{R}^{n \times n}$. The spectrum of a solution to (5) corresponds to a subset of the $2n$ eigenvalues of the matrix polynomial

$$\varphi(\lambda) := \lambda^2 A + \lambda B + C. \tag{6}$$

Instances of equation (5) arise in overdamped systems in structural mechanics [24] and are at the core of the matrix analytic method for *quasi-birth–death* (QBD) stochastic processes [13].

A typical situation in applications is that the eigenvalues of $\varphi(\lambda)$ are separated by the unit circle into two subsets of cardinality n:

$$|\lambda_1| \leqslant \dots |\lambda_n| \leqslant 1 \leqslant |\lambda_{n+1}| \leqslant \dots \leqslant |\lambda_{2n}|, \qquad |\lambda_n| < |\lambda_{n+1}|, \tag{7}$$

and it is of interest to compute the *minimal solution* of (5), that is, the solution X associated with $\lambda_1, \dots, \lambda_n$. Note that some of the eigenvalues are allowed to be infinite.

(7) implies that the matrix X is the only power bounded solution of (5); this uniquely identify the *matrix-geometric property* [13] of certain QBD processes. (7) also guarantees the quadratic convergence of the *cyclic reduction* algorithm for computing X [18, Theorem 9]. The minimal solution

can be constructed as $X = V \operatorname{diag}(\lambda_1, \dots, \lambda_n) V^{-1}$ if the matrix V containing the eigenvectors associated with $\lambda_1, \dots, \lambda_n$ is invertible [29]. This property is usually met in practice and in the QBD setting it can be ensured via the probabilistic interpretation of the minimal solution [34, Section 6.2].

We assume that the minimal solution exists and that (7) holds for a reference equation

$$A_0 X_0^2 + B_0 X_0 + C_0 = 0, \tag{8}$$

as well as for the modified equation

$$(A_0 + \delta A)(X_0 + \delta X)^2 + (B_0 + \delta B)(X_0 + \delta X) + (C_0 + \delta C) = 0, \tag{9}$$

where $\delta A, \delta B$ and δC are given low-rank matrices.

Denoting $A := A_0 + \delta A, B := B_0 + \delta B, C := C_0 + \delta C$ and subtracting (8) from (9) yields the following equation for the correction δX:

$$A \delta X^2 + (A X_0 + B) \delta X + A \delta X X_0 + \widehat{C} = 0, \tag{10}$$

where $\widehat{C} := \delta A X_0^2 + \delta B X_0 + \delta C$ has rank bounded by $\operatorname{rk}(\delta A) + \operatorname{rk}(\delta B) + \operatorname{rk}(\delta C)$. Note that equation (10) is not a UQME. Nevertheless, as will be seen in Section 2, there is still a correspondence between the solutions of (10) and an appropriately chosen eigenvalue problem. Similarly as for CARE, the low rank of $\widehat{C}$ will allow us to devise an efficient numerical method for (10).

Quadratic matrix equations with hierarchical low-rank structure. In the second part of the paper we focus on quadratic equations with coefficients that feature hierarchically low-rank structure. More specifically, the coefficients of a CARE (1) or a UQME (5) are assumed to be *hierarchically off-diagonal low-rank (HODLR) matrices* [2, 27]. This framework aligns well with the low-rank updates discussed above, because HODLR matrices are block diagonalized by a low-rank perturbation and, in turn, the corresponding reference equations (2) and (8) decouple into two equations of smaller size. Applying this idea recursively results in a divide-and-conquer method for solving UQMEs with HODLR coefficients and CAREs with a low-rank quadratic term and all other coefficients in the HODLR format.

Existing fast algorithms that address such (and more general) scenarios are based on combining a matrix iteration with fast arithmetic in hierarchical low-rank format. For CAREs, a combination of the sign function iteration with hierarchical matrices has been proposed in [3, 23]. For UQMEs, a combination based on cyclic reduction has been proposed in [16, 17]. As pointed out in [30], a disadvantage of these strategies is that they exploit the structure only indirectly and rely on repeated recompression during the iteration, which may constitute a computational bottleneck.

Outline. The rest of this paper is organized as follows. In Section 2, we study the correction equations (4) and (10), with a particular focus on providing intuition why one can expect their solutions to admit good low-rank approximations. Section 3 is concerned with numerical methods for obtaining such low-rank approximations. While a variety of large-scale solution methods have been recently developed for (4), the equation (10) is non-standard and requires the development of a novel large-scale solver, which may be of independent interest. Section 4 utilizes these solvers to derive divide-and-conquer methods for CARE and UQME featuring HODLR matrix coefficients. Finally, Section 5 highlights several applications of these divide-and-conquer methods and provides numerical evidence of their effectiveness.

2 Analysis of the correction equations

The purpose of this section is to study properties of the correction matrix δX, which satisfies one of the two correction equations, (4) or (10).

2.1 Existence and low-rank approximability

A necessary requirement of most solvers for large-scale matrix equations to perform well is that the solution admits good low-rank approximations. This property can sometimes be verified a priori by showing that the singular values exhibit a strong decay. In the following we first recall such results for linear matrix equations and then use them to shed some insight on the low-rank approximability of δX.

Singular value decay for linear matrix equations Let us consider the so called *Sylvester equation*

$$AX + XB = Q$$

with the coefficients $A, B, Q \in \mathbb{R}^{n\times n}$ such that the spectra of A and $-B$ are disjoint, and Q has rank $k \ll n$. Moreover, let $\mathcal{R}_{h,h}$ denotes the set of rational functions with numerator and denominator degrees at most h.

In [5], it is shown that for every $r \in \mathcal{R}_{h,h}$ there exists a matrix $\widetilde{X}$ of rank at most kh such that $X - \widetilde{X} = r(A)Xr(-B)^{-1}$, provided that the right-hand side is well defined. Using that the $(kh+1)$th singular value, denoted by $\sigma_{kh+1}(\cdot)$, governs the 2-norm error of the best approximation by a matrix of rank at most kh, one obtains

$$\sigma_{kh+1}(X) \leq \|r(A)\|_2 \|r(-B)^{-1}\|_2 \|X\|_2.$$

Combined with norm estimates for rational matrix functions, this leads to the following theorem.

Theorem 2.1 (Theorem 2.1 in [5])**.** *Consider the Sylvester equation $AX + XB = Q$, with Q of rank k, and let E and F be disjoint compact sets in the complex plane.*

(i) If E, F contain the numerical ranges of A and $-B$, respectively, then

$$\frac{\sigma_{kh+1}(X)}{\|X\|_2} \leqslant K_C \min_{r\in\mathcal{R}_{h,h}} \frac{\max_E |r(z)|}{\min_F |r(z)|},$$

where $K_C = 1$ if A, B are normal matrices and $1 \leq K_C \leq (1+\sqrt{2})^2$ otherwise.

(ii) If A, B are diagonalizable and E, F contain the spectra of A and $-B$, respectively, then

$$\frac{\sigma_{kh+1}(X)}{\|X\|_2} \leqslant \kappa_{\mathsf{eig}}(A)\kappa_{\mathsf{eig}}(B) \min_{r\in\mathcal{R}_{h,h}} \frac{\max_E |r(z)|}{\min_F |r(z)|}$$

where $\kappa_{\mathsf{eig}}(\cdot)$ denotes the 2-norm condition number of the eigenvector matrix.

The quantities $Z_h(E,F) := \min_{r\in\mathcal{R}_{h,h}} \frac{\max_E |r(z)|}{\min_F |r(z)|}$ are known in the literature as *Zolotarev numbers.* When E and F are well separated one can expect that $Z_h(E,F)$ decreases rapidly, as h increases, and quickly reaches the level of machine precision. Explicit bounds showing exponential decay have been established for various configurations of E and F, including disjoint real intervals and circles [5, 48].

CARE. The existence and uniqueness of a stabilizing solution to the correction equation (4) follows immediately from the observation that the closed-loop matrices of CARE (3) and CARE (4) are identical:

$$(A - FX_0) - F\delta X = A - FX.$$

This yields the following lemma.

Lemma 2.2. *Let X_0 be a solution of* (2)*. Then the correction equation* (4) *has a unique stabilizing solution δX if and only if the modified equation* (3) *has a unique stabilizing solution X.*

To study the low-rank approximability of δX, let us first assume that A is stable. By rearranging (4), we get

$$A^*\delta X + \delta X A = -\widehat{Q} + \delta X F(\delta X + X_0) + X_0 F \delta X.$$

Hence, δX satisfies a Lyapunov equation with the rank of the right-hand side bounded by $2\mathrm{rk}(F) + \mathrm{rk}(\widehat{Q})$. If, additionally, the numerical range of A is contained in the open left half plane then the first part of Theorem 2.1 can be applied to yield singular value bounds for δX. Alternatively, the second part can be applied under the milder assumption that A is diagonalizable.

If A is not stable, we rearrange (4) as

$$(A - FX)^*\delta X + \delta X(A - FX) = -\widehat{Q} - \delta X F \delta X.$$

As the closed loop matrix $A - FX$ is stable and the rank of the right-hand side is bounded by $\mathrm{rk}(F) + \mathrm{rk}(\widehat{Q})$, Theorem 2.1 applies under the assumptions stated above with A replaced by $A - FX$. One should note, however, that the obtained bounds are somewhat implicit because they involves the numerical range or the eigenvector conditioning of $A - FX$, quantities that are hard to estimate a priori. If more information is available for the closed loop matrix $A - F\widetilde{X}$ associated with a stabilizing initial guess $\widetilde{X}$, one can instead work with the equation

$$(A - F\widetilde{X})^*\delta X + \delta X(A - F\widetilde{X}) = -\widehat{Q} + \delta X F \delta X - 2\delta\widetilde{X} F \delta\widetilde{X},$$

where $\delta\widetilde{X} := \widetilde{X} - X_0$.

UQME. Solutions of the correction equation (10) are intimately related to the matrix pencil

$$\begin{bmatrix} X_0 & I \\ -\widehat{C} & -(AX_0 + B) \end{bmatrix} - \lambda \begin{bmatrix} I & 0 \\ 0 & A \end{bmatrix}. \qquad (11)$$

In fact, a direct computation shows that δX solves (10) if and only if

$$\begin{bmatrix} X_0 & I \\ -\widehat{C} & -(AX_0 + B) \end{bmatrix} \begin{bmatrix} I \\ \delta X \end{bmatrix} = \begin{bmatrix} I & 0 \\ 0 & A \end{bmatrix} \begin{bmatrix} I \\ \delta X \end{bmatrix} X. \qquad (12)$$

For simplicity, let us assume that A is invertible. Then the eigenvalues of (11) coincide with the eigenvalues of the matrix

$$\begin{bmatrix} X_0 & I \\ -A^{-1}\widehat{C} & -(X_0 + A^{-1}B) \end{bmatrix}.$$

By a similarity transformation,

$$\begin{bmatrix} I & 0 \\ \delta X & I \end{bmatrix}^{-1} \begin{bmatrix} X_0 & I \\ -A^{-1}\widehat{C} & -(X_0 + A^{-1}B) \end{bmatrix} \begin{bmatrix} I & 0 \\ \delta X & I \end{bmatrix} = \begin{bmatrix} X & I \\ 0 & -(X + A^{-1}B) \end{bmatrix}. \tag{13}$$

Because $X = X_0 + \delta X$ is a solution of (5), the quadratic matrix polynomial $\varphi(\lambda)$ defined in (6) admits the factorization

$$\lambda^2 A + \lambda B + C = (\lambda A + AX + B)(\lambda I - X) = A^{-1}(\lambda I + X + A^{-1}B)(\lambda I - X).$$

Together with (13), this shows that the eigenvalues of the pencil (11) coincide with the eigenvalues of $\varphi(\lambda)$. In particular, if X_0 and X are the minimal solutions of (8) and (5), respectively, then the spectra of X_0 and $-(X + A^{-1}B)$ are separated by the unit circle. By rearranging (9), δX can be viewed as the solution of a Sylvester equation with these coefficients and low-rank right hand side:

$$(X + A^{-1}B)\delta X + \delta X X_0 = -A^{-1}\widehat{C}. \tag{14}$$

This indicates good low-rank approximability of δX.

3 Low-rank updates

In Section 1 we already described the basic procedure for updating the solution X_0 of a reference CARE or UQME. This requires solving correction equations of the form (4) or (10), respectively. In the following, we discuss how to solve these correction equations efficiently .

3.1 Projection subspaces

According to the discussion in Section 2.1, one may expect that the solutions of (4) and (10) admit good low-rank approximations. A common strategy for obtaining such approximate solutions is to project these matrix equations to a pair of subspaces. To be more specific, let $U, V \in \mathbb{R}^{n\times t}$ contain orthonormal bases of t-dimensional subspaces $\mathcal{U}, \mathcal{V} \subset \mathbb{R}^n$. Then we consider approximate solutions of the form $\widetilde{X} := UYV^*$, where $Y \in \mathbb{R}^{t\times t}$ is obtained from solving a compressed matrix equation.

The choice of the projection subspaces $\mathcal{U}, \mathcal{V}$ is key to obtaining good approximations. In the context of matrix equations, *rational Krylov subspaces* [41] are a popular and effective choice.

Definition 3.1. *Let $A \in \mathbb{R}^{n\times n}$, $U_0 \in \mathbb{R}^{n\times k}$, $k < n$, and $\xi \in \mathbb{C}^t$. The vector space*

$$\mathcal{RK}_t(A, U_0, \xi) := \operatorname{range}\Big\{ \Big[U_0, (A - \xi_1 I)^{-1}U_0, \dots, \Big(\prod_{j=1}^{t-1} (A - \xi_j I)^{-1} \Big) U_0 \Big] \Big\}$$

is called rational Krylov subspace *with respect to* (A, U_0, ξ).

A good choice of shift parameters ξ_j is crucial and we will discuss our choices for CARE and UQME below.

3.2 Low-rank solution of correction equation for CARE

To describe our approach for approximating the solution of (4), let us define $A_{\mathsf{corr}} := A - FX_0$ and suppose that the low-rank updates of the coefficients are given in factorized, symmetry-preserving form:

$$\delta A = U_A V_A^*, \quad \delta Q = U_Q D_Q U_Q^*, \quad \delta F = U_F D_F U_F^*,$$

with $U_A, V_A \in \mathbb{R}^{n \times \mathrm{rk}(\delta A)}$, $U_Q \in \mathbb{R}^{n \times \mathrm{rk}(\delta Q)}$, $U_F \in \mathbb{R}^{n \times \mathrm{rk}(\delta F)}$, and symmetric matrices $D_Q \in \mathbb{R}^{\mathrm{rk}(\delta Q) \times \mathrm{rk}(\delta Q)}$, $D_F \in \mathbb{R}^{\mathrm{rk}(\delta F) \times \mathrm{rk}(\delta F)}$. This allows us to write the right-hand side of (4) in factorized form $\widehat{Q} = UDU^*$ as well, with

$$U := [U_Q, V_A, X_0 U_A, X_0 U_F], \quad D = \mathrm{diag}\left(D_Q, \begin{bmatrix} 0 & I_{\mathrm{rk}(U_A)} \\ I_{\mathrm{rk}(U_A)} & 0 \end{bmatrix}, D_F\right), \tag{15}$$

where diag denotes a block diagonal matrix with the blocks determined by the arguments. The correction equation (4) now reads as

$$A_{\mathsf{corr}}^* \delta X + \delta X A_{\mathsf{corr}} - \delta X F \delta X + \widehat{Q} = 0. \tag{16}$$

It is recommended to perform an optional preprocessing step that aims at reducing the rank of $\hat{Q}$ further. For this purpose, we compute a thin QR factorization $U = Q_U R_U$ followed by a (reordered) spectral decomposition

$$R_U D R_U^* = \begin{bmatrix} S_1 & S_2 \end{bmatrix} \begin{bmatrix} \Lambda_1 & 0 \\ 0 & \Lambda_2 \end{bmatrix} \begin{bmatrix} S_1 & S_2 \end{bmatrix}^*,$$

such that the diagonal matrix Λ_2 contains all eigenvalues of magnitude smaller than a prescribed tolerance τ_σ. Discarding these eigenvalues results in the reduced-rank approximation $\hat{Q} \approx UDU^*$ with $U \leftarrow Q_U S_1$ and $D \leftarrow \Lambda_1$.

For a large-scale CARE of the form (16), with both $\widehat{Q}$ and $F = BB^*$ of low rank, various numerical methods have been proposed [6,7,10,11,46,47]. In the following, we focus on the rational Krylov subspace method (RKSM) [46, 47], but other solvers could be used as well. While these algorithms usually assume $\widehat{Q}$ to be positive semi-definite, their extension to possibly indefinite $\widehat{Q}$ poses no major obstacle; see also the discussion in [33]. RKSM constructs an approximate solution of the form $\delta X_t = V_t Y V_t^*$, where V_t contains an orthonormal basis of a rational Krylov subspace $\mathcal{RK}_t(A_{\mathsf{corr}}^*, U, \xi)$. The small matrix Y is determined via a Galerkin condition, which comes down to solving the compressed CARE

$$\tilde{A}_{\mathsf{corr}}^* Y + Y \tilde{A}_{\mathsf{corr}} - Y \tilde{F} Y + \tilde{U} D \tilde{U}^* = 0, \quad \tilde{A}_{\mathsf{corr}} := V_t^* A_{\mathsf{corr}} V_t, \ \tilde{F} := V_t^* F V_t, \ \tilde{U} := V_t^* U.$$

for a stabilizing solution Y which can be addressed by direct algorithms for small, dense CAREs [9]. Note that the indefinite inhomogeneities $\tilde{U} D \tilde{U}^*$ are not an issue for the existence of such stabilizing solution which will then be also indefinite, see, e.g., [50]. In practice it can happen that the Hamiltonian matrix associated to the compressed CARE has eigenvalue close to the imaginary axis which can result in inaccurate solutions Y. For refining the accuracy of Y we apply a defect correction strategy similar to [37] given by (at most) 2 steps of a Newton's method.

Algorithm 1 gives a basic illustration of this method. We refer to the relevant literature [22, 46, 47] for implementation details and only comment on some critical steps. For selecting the shift

Algorithm 1 RKSM for (4) with $Q = UDU^*$ and $F = BB^*$

```
1: procedure LOW_RANK_CARE($A_{\mathsf{corr}}$, $B$, $U$, $D$)
2:     $V_1 = v_1 = \texttt{orth}(U)$        ▷ Orthonormalize $U$ by thin QR decomposition
3:     for $t = 1, 2, \ldots$ do
4:         $\tilde{A}^*_{\mathsf{corr}} \leftarrow V_t^* A_{\mathsf{corr}} V_t,\ \tilde{B} \leftarrow V_t^* B,\ \tilde{U} \leftarrow V_t^* U$
5:         $Y \leftarrow$DENSE_CARE($\tilde{A}_{\mathsf{corr}},\ \tilde{B},\ \tilde{U}D\tilde{U}^*$)
6:         if converged then return $\delta X_t = V_t Y V_t^*$
7:         Obtain next shift $\xi_t$.
8:         Solve $(A_{\mathsf{corr}} - \xi_t I)^* \tilde{v} = v_t$ for $\tilde{v}$.
9:         $\tilde{v} \leftarrow \tilde{v} - V_t(V_t^* \tilde{v})$, $v_{t+1} = \texttt{orth}(\tilde{v})$, $V_{t+1} = [V_t,\ v_{t+1}]$
10:    end for
11: end procedure
```

parameters in line 7 we employ the adaptive procedure from [22,46,47]. This may result in complex shifts or, more precisely, in complex conjugate pairs of shifts. The increased cost of working in complex arithmetic can be largely reduced by using an appropriate implementation, see, e.g., [40], which also returns a real approximation δX_t. The shifted linear systems in line 7 involve the matrix $(A_{\mathsf{corr}} - \xi_t I)^* = (A - \xi_t I - FX_0)^{-1}$. If A is sparse, such a system can be solved, e.g., by combining a sparse direct solver for $A - \xi_t I$ with the Sherman-Morrison-Woodbury formula to incorporate the low-rank modification FX_0. If A is a HODLR matrix then $A - \xi_t I - FX_0$ is a HODLR matrix as well and solvers for HODLR matrices can be used. Algorithm 1 is terminated once the residual is sufficiently small, that is, $\|R_t\|_2 = \|A^*_{\mathsf{corr}} \delta X_t + \delta X_t A^*_{\mathsf{corr}} - \delta X_t F \delta X_t + UDU^*\|_2 \leqslant \tau_{\mathsf{care}}$ for some prescribed tolerance $\tau_{\mathsf{care}} > 0$. An efficient way of computing the residual norm $\|R_t\|_2$ is described in [46]. After termination, it is recommended to perform an optional post-processing step, which aims at reducing the rank of δX_t, analogous to the rank-reducing procedure for $\hat{Q}$ described above.

3.2.1 Extension to generalized CAREs

In this section, we briefly discuss the extension of our low-rank update procedure to the generalized CARE (GCARE), see (1). The reference and modified equation take the form

$$A_0^* X_0 E_0 + E_0^* X_0 A_0 - E_0^* X_0 F_0 X_0 E_0 + Q_0 = 0, \tag{17}$$

$$\begin{aligned} &(A_0 + \delta A)^*(X_0 + \delta X)(E_0 + \delta E)^* + (E_0 + \delta E)^*(X_0 + \delta X)(A_0 + \delta A) \\ &- (E_0 + \delta E)^*(X_0 + \delta X)(F_0 + \delta F)(X_0 + \delta X)(E_0 + \delta E) + (Q_0 + \delta Q) = 0, \end{aligned} \tag{18}$$

where $\delta A,\ \delta E, \delta F$ and δQ are of low rank, and both $E = E_0 + \delta E$ as well as E_0 are invertible. By subtracting (17) from (18), we find that δX solves

$$(A - EX_0F)^* \delta X E + E^* \delta X (A - EX_0 F) - E^* \delta X F \delta X E + \widehat{Q} = 0 \tag{19}$$

where $\widehat{Q} := \delta Q + \delta A^* X_0 E + E^* X_0 \delta A - E^* X_0 \delta F X_0 E + \delta E^* X_0 \delta F X_0 E_0 + E^* X_0 \delta F X_0 \delta E$ satisfies $\mathrm{rk}\widehat{Q} \leqslant \mathrm{rk}(Q) + 2\mathrm{rk}(\delta A) + 2\mathrm{rk}(\delta F) + \min\{\mathrm{rk}(\delta F), \mathrm{rk}(\delta E)\}$. Similarly as in (15), we can write $\widehat{Q} = UDU^*$ with

$$\begin{aligned} U &:= [U_Q, V_A, E^* X_0 U_A, E_0^* X_0 U_F, V_E(U_E^* X_0) U_F], \\ D &= \hat{D}^* = \mathrm{diag}\left(D_Q, \begin{bmatrix} 0 & I_{\mathrm{rk}(U_A)} \\ I_{\mathrm{rk}(U_A)} & 0 \end{bmatrix}, D_F, \begin{bmatrix} 0 & I_{\mathrm{rk}(U_E)} \\ I_{\mathrm{rk}(U_E)} & I_{\mathrm{rk}(U_E)} \end{bmatrix} \right). \end{aligned}$$

where $\delta E = U_E V_E^*$ with $U_E, V_E \in \mathbb{R}^{n\times \mathrm{rk}(\delta E)}$. Again, an optional rank-reducing step for $\hat{Q}$ is recommended. By implicitly working on the equivalent CARE defined by the coefficients $E^{-1}(A - EX_0F)$, F, $E^{-*}\widehat{Q}E^{-1}$, Algorithm 1 extends with minor modifications to (19). We refer to [11,46,47] for further details.

3.3 Low-rank solution of correction equation for UQME

The correction equation (10) features a constant coefficient that has low rank. However, unlike in the case of CARE, we are not aware of existing large-scale solvers tailored to this situation, neither for UQME nor for the modified form (10). For example, a fast cyclic reduction iteration proposed in [14] requires *both* the quadratic and the constant coefficient (that is, the matrices A and C in (5)) to be of low rank.

In the following, we develop a novel subspace projection method, largely inspired by the existing techniques for CARE described above.

We first discuss the choice of subspaces for our method and consider the Sylvester equation (14) for this purpose. In principle, subspace projection methods for Sylvester equations are well understood, but the coefficients of (14) involve the matrix X, which depends on the unknown δX. Solely for the purpose of choosing the subspaces, we replace X with the reference solution X_0 and consider

$$(X_0 + A^{-1}B)\delta X + \delta X X_0 = -A^{-1}\widehat{C} \tag{20}$$

instead. Again, we assume that the low-rank updates are given in factorized form: $\delta A = U_A V_A^*, \delta B = U_B V_B^*$ and $\delta C = U_C V_C^*$. Then, the right-hand sides of (14) and (20) can be written as $A^{-1}\widehat{C} = UV^*$ with

$$U = [A^{-1}U_A, A^{-1}U_B, A^{-1}U_C], \qquad V = [(X_0^*)^2 V_A, X_0^* V_B, V_C]. \tag{21}$$

As for CARE, it is recommended to apply a preprocessing step aiming at reducing the rank of UV^*. Existing solver for Sylvester equations [46] suggest the use of rational Krylov subspaces with coefficient matrices $X_0 + A^{-1}B$, X_0^* and starting vectors U, V in order to solve (20). Specifically, we choose

$$\mathcal{U}_t := \mathcal{RK}_{2t}(X_0 + A^{-1}B, U, \pm\mathbf{1}_t), \qquad \mathcal{V}_t := \mathcal{RK}_{2t}(X_0^*, V, \pm\mathbf{1}_t), \tag{22}$$

where $\pm\mathbf{1}_t = [1, -1, \ldots, 1, -1]^* \in \mathbb{R}^{2t}$. This particular choice of shift parameters corresponds to the *extended Krylov subspace* [45] for Sylvester equations, adapted to the case in which the spectra of the coefficients are separated by the unit circle instead of the imaginary line. Indeed, we replaced 0 and ∞, the usual choice in the extended Krylov method, with $T(0) = -1$ and $T(\infty) = 1$ where $T(z) := -\frac{1+z}{1-z}$ is the Cayley transform.

Suppose now that U_t, V_t contain orthonormal bases of the subspaces defined in (22). To construct an approximate solution $\delta X_t = U_t Y V_t^*$ of the *original* equation (14), we impose a Galerkin condition with respect to the tensorized space $\mathcal{U}_t \otimes \mathcal{V}_t$. This implies that the small matrix Y satisfies the non-symmetric algebraic Riccati equation

$$\begin{gathered} Y\widetilde{F}Y + \widetilde{A}Y + Y\widetilde{D} = \widetilde{Q}, \\ \widetilde{A} = U_t^*(X_0 + A^{-1}B)V_t, \quad \widetilde{F} = V_t^* U_t, \quad \widetilde{D} = V_t^* X_0^* V_t, \quad \widetilde{Q} = U_t^* U V^* V_t. \end{gathered} \tag{23}$$

This compressed equation is solved by the *structured doubling algorithm (SDA)* [15, 25], see also Algorithm 2. If the projected Hamiltonian

$$\begin{bmatrix} \widetilde{A} & -\widetilde{F} \\ -\widetilde{Q} & -\widetilde{D} \end{bmatrix} \tag{24}$$

has an eigenvalue splitting with respect to the unit disc and both equation (23) and its dual equation (the one obtained interchanging $\widetilde{F}$ and $\widetilde{Q}$) admit a minimal solution, then SDA converges quadratically to the minimal solution of (23) [15, Theorem 5.4].

Algorithm 2 Structured doubling algorithm for (23)

```
1:  procedure SDA_NARE($\widetilde{A}, \widetilde{D}, \widetilde{F}, \widetilde{Q}$)
2:      $\begin{bmatrix} S_{11} & S_{12} \\ S_{21} & S_{22} \end{bmatrix} \leftarrow \begin{bmatrix} I & \widetilde{F} \\ 0 & -\widetilde{A} \end{bmatrix}^{-1} \begin{bmatrix} \widetilde{D} & 0 \\ -\widetilde{Q} & I \end{bmatrix}$
3:      $E \leftarrow S_{11}, \quad G = -S_{12}, \quad P = -S_{21}, \quad F = S_{22}$
4:      for $t = 1, 2, \ldots$ do
5:          if converged then return $P$
6:          $\widetilde{G} \leftarrow I - G \cdot P, \quad \widetilde{P} \leftarrow I - P \cdot G$
7:          $E_1 \leftarrow E^{-1}\widetilde{G}, \quad F_1 \leftarrow F^{-1}\widetilde{P}$
8:          $G \leftarrow G + E_1 \cdot G \cdot F, \quad P \leftarrow P + F_1 \cdot P \cdot E$
9:          $E \leftarrow E_1 \cdot E, \quad F \leftarrow F_1 \cdot F;$
10:     end for
11: end procedure
```

The whole procedure for solving (14) is summarized in Algorithm 3. A few remarks concerning

Algorithm 3 Extended Krylov subspace method for (14)

```
1:  procedure LOW_RANK_UQME_CORR($A, B, X_0, U, V$)
                    ▷ $U, V \in \mathbb{R}^{n \times r}$ defined as in (21) or pre-processed by a rank-reducing step
2:      $\widehat{A} \leftarrow X_0 + A^{-1}B$
3:      $U_1 = \texttt{orth}([(\widehat{A}+I)^{-1}U, (\widehat{A}-I)^{-1}U]), \quad V_1 = \texttt{orth}([V, (X_0^*+I)^{-1}V, (X_0^*-I)^{-1}V])])$
4:      for $t = 1, 2, \ldots$ do
5:          $\widetilde{A} \leftarrow U_t^*\widehat{A}V_t, \quad \widetilde{D} \leftarrow U_t^*X_0^*V^*V_t, \quad \widetilde{F} \leftarrow V_t^*U_t, \quad \widetilde{Q} \leftarrow U_t^*UV^*V_t$
6:          $Y \leftarrow$ SDA_NARE($\widetilde{A}, \widetilde{D}, \widetilde{F}, \widetilde{Q}$)
7:          if converged then return $\delta X_t := U_tYV_t^*$
8:          Partition $U_t = [U^{(0)}, U^{(+)}, U^{(-)}]$ such that $U^{(+)}, U^{(-)} \in \mathbb{R}^{n \times n}$
9:          Partition $V_t = [V^{(0)}, V^{(+)}, V^{(-)}]$ such that $V^{(+)}, V^{(-)} \in \mathbb{R}^{n \times n}$
10:         $\widetilde{U} = [(\widehat{A}+I)^{-1}U^{(+)}, (\widehat{A}-I)^{-1}U^{(-)}], \quad \widetilde{V} = [(X_0^*+I)^{-1}V^{(+)}, (X_0^*-I)^{-1}V^{(-)}]$
11:         $\widetilde{U} \leftarrow \widetilde{U} - U_tU_t^*\widetilde{U}, \quad \tilde{U} = \texttt{orth}(\tilde{U}), \quad U_{t+1} = [U_t, \tilde{U}]$
12:         $\widetilde{V} \leftarrow \widetilde{V} - V_tV_t^*\widetilde{V}, \quad \tilde{V} = \texttt{orth}(\tilde{V}), \quad V_{t+1} = [V_t, \tilde{V}]$
13:     end for
14: end procedure
```

the implementation of Algorithms 2 and 3:

- Algorithm 2 is stopped either when $\min\{\|E\|_1, \|F\|_1\} < 10^{-13}$ or when a maximum of 30 iterations is reached. If the projected Hamiltonian (24) has the desired splitting of eigenvalues with respect to the unit circle then Algorithm 2 converges quadratically and is therefore likely to match the convergence condition within 30 iterations. Otherwise, we move on and consider the next (enlarged) extended Krylov subspaces.
- We rely on the `rktoolbox` [12] for executing the rational block Arnoldi processes that return the orthonormal bases U_t and V_t. We remark that the compressed matrices in line 5 of Algorithm 3 do not need to be computed explicitly; they can be obtained from the rational Krylov decomposition by adding an artificial final step with an infinite shift, see [26, page 74] and [4].
- The number of iterations t in Algorithm 3 is chosen adaptively to ensure that the relation

$$\|\delta X_t^2 + (X_0 + A^{-1}B)\delta X_t + \delta X_t X_0 + A^{-1}\widehat{C}\|_2 \leqslant \tau_{\mathsf{uqme}} \tag{25}$$

  is satisfied for some tolerance τ_{uqme}. The artificial final step with an infinite shift mentioned above allows this relation to be verified efficiently, see [28, 45].
- For applying $(\widehat{A} \pm I)^{-1}$, $(X_0^* \pm I)^{-1}$, LU factorizations of $\widehat{A} \pm I$ and $X_0^* \pm I$ are computed once before starting the rational block Arnoldi process.
- After termination of Algorithm 3, it is – once again – recommended to perform an optional post-processing step that aims at reducing the rank of δX_t.

4 Divide-and-conquer methods

Having an efficient procedure for performing low-rank updates at hand allows us to design divide-and-conquer methods for quadratic matrix equations with rank structured coefficients. For example, suppose that the coefficients of the CARE (3) admit the decompositions

$$A = A_0 + \delta A, \quad F = F_0 + \delta F, \quad Q = Q_0 + \delta Q, \tag{26}$$

where A_0, F_0, Q_0 are block diagonal matrices of the same shape and $\delta A, \delta F, \delta Q$ have low rank. This allows us to split (3) into the correction equation (4), which we solve with Algorithm 1, and the two smaller, decoupled equations associated with the diagonal blocks of A_0, F_0, Q_0. If these diagonal blocks again admit a decomposition of the form (26), we recursively repeat the splitting. The described strategy easily adapts to the UMQE (9).

The storage and manipulation of the low-rank corrections on the various levels of the recursion requires to work with a suitable format, such as the HODLR format.

4.1 HODLR matrices

A HODLR matrix $A \in \mathbb{R}^{n\times n}$ admits block partition

$$A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix}, \tag{27}$$

where A_{12}, A_{21} have low rank and A_{11}, A_{22} are square matrices that again take the form (27). This splitting is continued recursively until the diagonal blocks reach a certain minimal block size n_{min}. Usually, the partitioning is chosen such that A_{11}, A_{22} have nearly equal sizes. Banded matrices are an important special case of HODLR matrices.

We say that A has *HODLR rank* k if k is the smallest integer such that the ranks of A_{21} and A_{12} in (27) are bounded by k at all levels of the recursion. If k remains small then A can be stored efficiently by replacing each off-diagonal block with its low-rank factors. The only dense blocks that need to be stored are the diagonal blocks at the lowest level, see Figure 1. In turn, the storage of a HODLR matrix requires $O(kn\log n)$ memory.

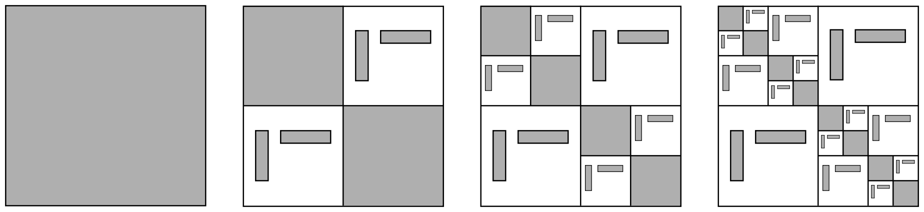

Figure 1: Image taken from [30] describing the HODLR format for different recursion depths.

4.2 Divide-and-conquer in the HODLR format

For a CARE (3) with HODLR matrices A, Q and a low-rank matrix $F = BB^*$, a divide-and-conquer method can be derived along the lines of the linear case discussed in [30]. Consider

$$A = \begin{bmatrix} A_{11} & 0 \\ 0 & A_{22} \end{bmatrix} + \begin{bmatrix} 0 & U_1 V_2^* \\ U_1 V_2^* & 0 \end{bmatrix} = \begin{bmatrix} A_{11} & 0 \\ 0 & A_{22} \end{bmatrix} + U_A V_A^*, \quad U_A = \begin{bmatrix} U_1 & \\ & U_2 \end{bmatrix}, \quad V_A = \begin{bmatrix} V_1 & \\ & V_2 \end{bmatrix} \tag{28a}$$

and likewise, by exploiting symmetry and low-rank structure for Q and F, the splittings

$$Q = \begin{bmatrix} Q_{11} & 0 \\ 0 & Q_{22} \end{bmatrix} + \begin{bmatrix} 0 & U_1 U_2^* \\ U_2 U_1^* & 0 \end{bmatrix} = \begin{bmatrix} Q_{11} & 0 \\ 0 & Q_{22} \end{bmatrix} + U_Q D_Q U_Q^*, \quad U_Q = \begin{bmatrix} U_1 & \\ & U_2 \end{bmatrix}, \quad D_Q = \begin{bmatrix} 0 & I \\ I & 0 \end{bmatrix}, \tag{28b}$$

$$F = BB^* = \begin{bmatrix} B_1 \\ B_2 \end{bmatrix} \begin{bmatrix} B_1 \\ B_2 \end{bmatrix}^* = U_F U_F^* + U_F D_F U_F, \quad U_F = \begin{bmatrix} B_1 & \\ & B_2 \end{bmatrix}, \quad D_F = \begin{bmatrix} 0 & I \\ I & 0 \end{bmatrix}. \tag{28c}$$

The diagonal blocks A_{ii}, Q_{ii}, $i = 1, 2$ are again HODLR matrices (with the recursion depth reduced by one). After recursively solving the CAREs associated with the diagonal blocks, a low-rank approximation to the solution of the correction equation (4) is obtained with Algorithm 1. The resulting procedure is summarized in Algorithm 4. As highlighted in the pseudo-code, it is strongly recommended to reduce the ranks of UDU^* in line 9 and of $X_0 + \delta X$ in line 11. Algorithm 4 requires the equations associated with the diagonal blocks to admit a unique stabilizing solution at all levels of the recursion.

The divide-and-conquer method for a UQME with HODLR matrix coefficients is derived in an analogous manner. The only substantial changes are that the equations associated with the diagonal blocks are solved by *cyclic reduction* [18], see Algorithm 5. The resulting procedure is summarized in Algorithm 6. This algorithm requires that the matrix polynomials associated with the diagonal blocks — $\lambda^2 A_{jj} + \lambda B_{jj} + C_{jj}$ for $j = 1, 2$ — maintain the splitting property (7), at all levels of the recursion. Similarly as in Algorithm 4, compression is recommended in lines 9 and 11 of Algorithm 6.

Algorithm 4 Divide-and-conquer method for CARE with HODLR coefficients

```
1:  procedure D&C_CARE($A, B, Q$)                    ▷ Solve $A^*X + XA - XBB^TX + Q = 0$
2:      if $A$ is a dense matrix then
3:          return DENSE_CARE($A, B, Q$)
4:      else
5:          Decompose
                $$A = \begin{bmatrix} A_{11} & 0 \\ 0 & A_{22} \end{bmatrix} + U_A V_A^*,\ F = U_F U_F^* + U_F D_F U_F^*,\ Q = \begin{bmatrix} Q_{11} & 0 \\ 0 & Q_{22} \end{bmatrix} + U_Q D_Q U_Q^*$$
            with $U_A, V_A, U_F, D_F, U_Q, D_Q$ defined as in (28).
6:          $X_{11} \leftarrow$ D&C_CARE($A_{11}, B_1, Q_{11}$)
7:          $X_{22} \leftarrow$ D&C_CARE($A_{22}, B_2, Q_{22}$)
8:          Set $X_0 \leftarrow \begin{bmatrix} X_{11} & 0 \\ 0 & X_{22} \end{bmatrix}$
9:          Set $U = [U_Q, V_A, X_0U_A, X_0U_F]$ and $D$ as in (15).   ▷ Compression is recommended
10:         $\delta X \leftarrow$ LOW_RANK_CARE($A - (X_0B)B^*, B, U, D$)   ▷ Algorithm 1
11:         return $X_0 + \delta X$                        ▷ Compression is recommended
12:     end if
13: end procedure
```

Algorithm 5 Cyclic reduction for UQME

```
1:  procedure DENSE_CR($A, B, C$)                    ▷ Solve $AX^2 + BX + C = 0$
2:      $A^{(0)} \leftarrow A$, $B^{(0)} \leftarrow B$, $\widehat{B}^{(0)} \leftarrow B$, $C^{(0)} \leftarrow C$
3:      for $t = 0, 1, \ldots$ do
4:          if converged then return $-(\widehat{B}^{(t)})^{-1}B$
5:          $A^{(t+1)} \leftarrow -A^{(t)}(B^{(t)})^{-1}A^{(t)}$
6:          $B^{(t+1)} \leftarrow B^{(t)} - C^{(t)}(B^{(t)})^{-1}A^{(t)} - A^{(t)}(B^{(t)})^{-1}C^{(t)}$
7:          $\widehat{B}^{(t+1)} \leftarrow \widehat{B}^{(t)} - C^{(t)}(B^{(t)})^{-1}A^{(t)}$
8:          $C^{(t+1)} \leftarrow -C^{(t)}(B^{(t)})^{-1}C^{(t)}$.
9:      end for
10: end procedure
```

Algorithm 6 Divide-and-conquer method for UQME with HODLR coefficients

```
1: procedure D&C_UQME($A, B, C$)                    ▷ Solve $AX^2 + BX + C = 0$
2:     if $A$ is a dense matrix then
3:         return DENSE_CR($A, B, C$)
4:     else
5:         Decompose
               $A = \begin{bmatrix} A_{11} & 0 \\ 0 & A_{22} \end{bmatrix} + U_A V_A^*,\ B = \begin{bmatrix} B_{11} & 0 \\ 0 & B_{22} \end{bmatrix} + U_B V_B^*,\ C = \begin{bmatrix} C_{11} & 0 \\ 0 & C_{22} \end{bmatrix} + U_C V_C^*.$
6:         $X_{11} \leftarrow$ D&C_UQME($A_{11}, B_{11}, C_{11}$)
7:         $X_{22} \leftarrow$ D&C_UQME($A_{22}, B_{22}, C_{22}$)
8:         Set $X_0 \leftarrow \begin{bmatrix} X_{11} & 0 \\ 0 & X_{22} \end{bmatrix}$
9:         Set $U = [A^{-1}U_A, A^{-1}U_B, A^{-1}U_C]$ and $V = [(X_0^*)^2 V_A, X_0^* V_B, V_C]$
10:        $\delta X \leftarrow$ LOW_RANK_UQME_CORR($A, B, X_0, U, V$)
11:        return $X_0 + \delta X$
12:    end if
13: end procedure
```

4.2.1 Complexity of divide-and-conquer in the HODLR format

The complexity of Algorithms 4 and 6 critically depends on the convergence of the projection methods (Algorithms 1 and 3, respectively) used for solving the correction equations. To a milder extent, it also depends on the numerical methods used for solving the small dense equations associated with the diagonal blocks on the lowest level of the recursion. In order to provide some insights of the computational cost we make the following simplifying assumptions:

(i) Algorithm 1 and Algorithm 3 converge in a constant number of iterations;

(ii) solving the dense (unstructured) equations has complexity $\mathcal{O}(n^3)$;

(iii) the matrix Q in CARE has rank k;

(iv) all involved HODLR matrices have HODLR rank k and have a regular partition, that is, $n = 2^p n_{\mathsf{min}}$ and the splitting (27) always generates equally sized diagonal blocks;

(v) the compressions in Algorithm 4 and Algorithm 6 is *not* performed.

Under the assumptions stated above, the LU decomposition of an $n \times n$ HODLR matrix requires $\mathcal{O}(k^2 n \log^2(n))$ operations, while performing forward or backward substitution with a vector is $\mathcal{O}(kn\log(n))$. A matrix-vector product is $\mathcal{O}(kn\log(n))$ and all involved matrix-matrix operations are at most $\mathcal{O}(k^2 n \log^2(n))$, see, e.g., [27].

CARE. Let $\mathcal{C}_{\mathsf{care}}(n, k)$ denote the complexity of Algorithm 4. Assumption (i) implies that the cost of Algorithm 1, called at Line 10, is $\mathcal{O}(k^2 n \log^2(n))$, because it is dominated by the cost of solving (shifted) linear systems with the matrix A_{corr}. Assumption (i) also implies that X_0, see Line 8, has HODLR rank $\mathcal{O}(k\log(n))$. Because U_A and U_F each have $2k$ columns, the matrix multiplications X_0U_A and X_0U_F at Line 9 require $\mathcal{O}(k^2 n \log^2(n))$ operations. Finally, thanks to assumption (ii) we have

$$\mathcal{C}_{\mathsf{care}}(n,k) = \begin{cases} \mathcal{O}(n_{\mathsf{min}}^3) & \text{if } n = n_{\mathsf{min}}, \\ \mathcal{O}(k^2 n \log^2(n)) + 2\mathcal{C}_{\mathsf{care}}(\frac{n}{2}, k) & \text{otherwise.} \end{cases} \tag{29}$$

Applying the master theorem [20] to (29) yields $\mathcal{C}_{\mathsf{care}}(n,k) = \mathcal{O}(k^2 n \log^3(n))$.

UQME. Let $\mathcal{C}_{\mathsf{uqme}}(n,k)$ denote the complexity of Algorithm 6. Analogously to CARE, Assumption (i) implies that Algorithm 1 requires $\mathcal{O}(k^2 n \log(n)^2)$ operations and that X_0 at Line 8 has HODLR rank $\mathcal{O}(k \log(n))$. Therefore, the complexity of Line 9 is given by the one of solving $\mathcal{O}(k)$ linear systems, that is, $\mathcal{O}(k^2 n \log^2(n))$. In turn, the recurrence relation for $\mathcal{C}_{\mathsf{uqme}}(n,k)$ is identical with (29) and hence $\mathcal{C}_{\mathsf{uqme}}(n,k) = O(k^2 n \log^3(n))$.

5 Numerical results

We now proceed to verify the numerical performance of the divide-and-conquer methods, Algorithms 4 and 6 from Section 4. Our methods are compared with state-of-the-art iterative algorithms for solving quadratic matrix equations:

- structure preserving doubling algorithm (SDA) for CARE [19],
- cyclic reduction (CR) for UQME [18] (Algorithm 5).

Both algorithms are well suited for coefficients with hierarchical low-rank structures; we have implemented in HODLR arithmetic using the `hm-toolbox` [36]. As indicated in the description of the algorithms, we apply recompression with the threshold $\tau_\sigma = 10^{-12}$ in order to keep the ranks under control. Unless stated otherwise, we set the minimal block-size to $n_{\mathsf{min}} = 256$ for the representation in the HODLR format. The parameters $\tau_{\mathsf{care}}, \tau_{\mathsf{uqme}}$ used in Algorithm 4 and Algorithm 6, respectively, for stopping the low-rank iterative solver have been set to 10^{-8}.

All experiments have been performed on a Laptop with a dual-core Intel Core i7-7500U 2.70 GHz CPU, 256KB of level 2 cache, and 16 GB of RAM. The algorithms are implemented in MATLAB and tested under MATLAB2017a, with MKL BLAS version 11.2.3 utilizing both cores.

5.1 Results for CARE

We will use the following three examples to test the performance of Algorithm 4 for CARE.

Example 5.1. This is an academic example of arbitrary size n: $A = \text{tridiag}(1, -2, 1)$, that is, A is a tridiagonal matrix with -2 on the diagonal and 1 on the sub- and supdiagonal. The matrix $B \in \mathbb{R}^{n\times 2}$ is random with normally distributed entries, $Q = Q_0 + (0.1 - \theta)I$, where Q_0 is a random symmetric tridiagonal matrix also with normally distributed entries, and $\theta \in \mathbb{R}$ is the smallest eigenvalue of Q_0.

Example 5.2. This example is taken from [30, Example 3.2 and Section 5.6]:

$$A = \left[\begin{smallmatrix} 0 & M \\ I & -I \end{smallmatrix}\right], \quad M = \tfrac{1}{4}\,\text{tridiag}(1, -2, 1) - \tfrac{1}{2}(e_1 e_1^T + e_n e_n^T), \quad B = \tfrac{1}{4}[e_{n+1}, -e_{2n}], \quad Q = I_n,$$

	Algorithm 4			SDA		
n	Time	Res	HODLR rank	Time	Res	HODLR rank
1,024	2.06	$4.41 \cdot 10^{-11}$	55	6.11	$1.17 \cdot 10^{-10}$	55
2,048	4.02	$1.00 \cdot 10^{-10}$	58	21.36	$1.82 \cdot 10^{-9}$	59
4,096	8.33	$5.85 \cdot 10^{-10}$	67	65.42	$1.42 \cdot 10^{-9}$	67
8,192	18.27	$5.62 \cdot 10^{-9}$	67	226	$1.33 \cdot 10^{-8}$	67
16,384	39.71	$1.02 \cdot 10^{-8}$	74	670.17	$7.44 \cdot 10^{-8}$	76
32,768	84.1	$5.56 \cdot 10^{-8}$	73	2,023.8	$2.33 \cdot 10^{-6}$	62

Table 1: Execution times (in seconds) and residuals for the divide-and-conquer method and SDA applied to the CARE from Example 5.1.

where e_j denotes the jth unit vector of appropriate length. Since A is unstable, we use an initial stabilizing solution $X_0 := Z_0 Z_0^*$, $Z_0 = 8 \left[\begin{smallmatrix} -e_n & e_1 \\ -e_n & e_1 \end{smallmatrix}\right]$ and consider the stabilized CARE given by $\tilde{A} := A - X_0^* BB^*$ and $\tilde{Q} := Q - X_0^* BB^* X_0^* + A^* X_0 + X_0 A$. Because of the structure of B and Z_0, $\tilde{A}$ is still sparse. All matrices are scaled by $\|A\|_2$ and, to acquire a banded structure, reordered by a perfect shuffle permutation.

Example 5.3. This example is `carex18` from the CARE benchmark collection [1] with tridiagonal A and E, $B \in \mathbb{R}^n$, but we set $Q := I_n$.

All matrices have been converted into the HODLR format using the `hmtoolbox`. However, for the fast solution of the linear systems in Algorithm 1 we invoke the original sparse matrices A and E and call a sparse direct solver via MATLAB's "backslash". The results — compared to those of SDA — are summarized in the Tables 1–3, where Res $= \|A^*X + XA - XBB^*X + Q\|_2 / \|X\|_2$. The reported computing times for both methods clearly reveal that the divide-and-conquer method requires substantially less time than SDA while achieving a similar or even better level of accuracy at the same time. Most of the time in SDA was spent in the numerous HODLR matrix-matrix multiplications and the associated recompression steps after each multiplication.

For the largest matrices from Example 5.1, $n = 32\,768$, we have profiled the computing time spent at the different stages of the divide-and-conquer method. Solving dense CAREs for the diagonal blocks at the lowest level of recursion consumed about 30% of the total time, while about 50% was spent on solving the correction equation, CARE (4), by RKSM (Algorithm 1). About 15% of the time was spent on performing the update $X_0 + \delta X$ (line 11 in Algorithm 4). The work spent on rank compressions was negligible; it consumed less than 1% of the total time. Within RKSM, orthonormalization within the Arnoldi method and the solution of the compressed CAREs were the most time consuming steps (totaling approximately 40% of the time spent on RKSM), followed by the procedure for shift generation (15%). Due to the sparse, banded structure of A, the linear system solves consumed only a very small fraction (about 3%). We note that the time for solving the correction equation (4) could potentially be reduced by employing a different low-rank solver for CAREs. A good candidate for such a solver is the recently proposed RADI method [7][1], which does not rely on Galerkin projections and, hence, does not require solving compressed CAREs. We also investigated the effect of reducing the block size n_{min} from 256 to 128. As expected, this decreased the fraction of computing time spent on diagonal blocks from 30% to 10% but, due to the higher

[1]Preliminary results suggest that replacing Algorithm 1 with RADI reduces the time by 10% on average over all used sizes n.

	Algorithm 4			SDA		
n	Time	Res	HODLR rank	Time	Res	HODLR rank
1,024	1.44	$2.34 \cdot 10^{-10}$	10	3.15	$9.54 \cdot 10^{-14}$	5
2,048	2.74	$1.23 \cdot 10^{-11}$	9	7.76	$2.54 \cdot 10^{-13}$	5
4,096	5.5	$8.03 \cdot 10^{-12}$	10	20.43	$2.80 \cdot 10^{-13}$	4
8,192	11.2	$7.51 \cdot 10^{-13}$	10	52.07	$2.88 \cdot 10^{-13}$	4
16,384	22.21	$8.82 \cdot 10^{-13}$	10	131.18	$2.89 \cdot 10^{-13}$	3
32,768	44.47	$1.56 \cdot 10^{-13}$	9	312.57	$2.89 \cdot 10^{-13}$	3

Table 2: Execution times (in seconds) and residuals for the divide-and-conquer method and SDA applied to the CARE from Example 5.2.

	Algorithm 4		
n	Time	Res	HODLR rank
1,024	7.54	$1.20 \cdot 10^{-7}$	23
2,048	15.01	$3.02 \cdot 10^{-8}$	27
4,096	29.74	$7.58 \cdot 10^{-9}$	28
8,192	62.01	$1.90 \cdot 10^{-9}$	31
16,384	128.99	$4.74 \cdot 10^{-10}$	28
32,768	263.61	$1.19 \cdot 10^{-10}$	27

Table 3: Execution times (in seconds) and residuals for the divide-and-conquer method and SDA applied to the CARE from Example 5.3.

number of occurrences, increased part spent on solving the correction CAREs to about 67%. This resulted in a negligible change in the overall time for Algorithm 4 compared to $n_{\mathsf{min}} = 256$.

5.2 Results for UQMEs from QBD processes

QBD processes are discrete-time stochastic processes with a two-dimensional discrete state space. The variables of the state space are called *level* and *phase*; the transition — at each time step — with respect to the level coordinate has at most unit length. We consider models whose state space is isomorphic to $\mathbb{N} \times \{0, \dots, n-1\}$, that is, we have infinite levels and a finite number of possible phases. Moreover, we assume that the process is *level independent*, i.e. the transition probability depends on the variation of the level but not on its current value.

Computing the stationary distribution of a level independent QBD process amounts to solving a UQME with coefficients corresponding to (possibly shifted) sub-blocks of its transition probability matrix [13]. More specifically, the coefficients of the UQME $AX^2 + BX + C = 0$ have the properties that $A, B+I, C \in \mathbb{R}^{n\times n}$ are non-negative and $A+B+C+I$ is stochastic, that is, each row sums to one. As the following lemma shows, these properties imply — under some mild additional conditions — the eigenvalue splitting property (7) on every level of recursion in the divide-and-conquer method.

Lemma 5.4. *Suppose that A, B, C have the properties stated above and that $\varphi(\lambda)$ has only one eigenvalue on the unit circle, the simple eigenvalue 1. For some index set $J \subseteq \{1, \dots, n\}$, let A_J, B_J, C_J denote the corresponding principal submatrices of A, B, C. Assume that B_J is invertible and $B_J^{-1}(A_J + C_J)$ is irreducible. Then $\varphi_J(\lambda) := \lambda^2 A_J + \lambda B_J + C_J$ has the splitting property* (7).

Proof. For the moment, let us assume that $A_J + B_J + C_J + I$ is substochastic, that is, $(A_J + B_J + C_J + I)\mathbf{e} \lneq \mathbf{e}$, where $\mathbf{e}$ denotes the vector of all ones and the inequality is understood componentwise. We aim at utilizing the following consequence of Rouché's theorem for matrix-valued functions [38, Theorem 3.2]: if

$$\|(\lambda B_J)^{-1}(\lambda^2 A_J + C_J)\| < 1, \qquad \forall|\lambda| = 1, \tag{30}$$

holds for an induced norm $\|\cdot\|$ then $\varphi_J(z)$ has exactly k eigenvalues (counting multiplicities) in the open unit disc and k eigenvalues with modulus greater than 1, where k denotes the cardinality of J. This implies the result of the lemma.

Setting $\psi(\lambda) := -B_J^{-1}(\lambda A_J + \lambda^{-1}C_J)$, the condition (30) clearly holds if we can show that the spectral radius $\rho(\psi(\lambda))$ is less than 1 for every λ on the unit circle. Note that $|\lambda A_J + \lambda^{-1}C_J| \leq A_J + C_J$ because A_J, C_J are non-negative. Combined with the fact that $-B_J$ is an M-matrix, which implies $-B_J^{-1} \geqslant 0$, and the monotonicity of the spectral radius, we obtain

$$\rho(\psi(\lambda)) \leq \rho(|\psi(\lambda)|) = \rho(|-B_J^{-1}(\lambda A_J + \lambda^{-1}C_J)|) \leq \rho(-B_J^{-1}(A_J + C_J)) = \rho(\psi(1)).$$

Using $-B_J^{-1} \geqslant 0$ we also have

$$(A_J + B_J + C_J + I)\mathbf{e} \lneq \mathbf{e} \quad \Longrightarrow \quad (A_J + C_J)\mathbf{e} \lneq -B_J\mathbf{e} \quad \Longrightarrow \quad \psi(1)\mathbf{e} \lneq \mathbf{e}.$$

In particular, the matrix $\psi(1)$ is irreducible and substochastic, and by the Perron Frobenius theorem [43, Theorem 1.5] it has spectral radius strictly less than 1.

It remains to consider the case when $A_J + B_J + C_J + I$ is stochastic. Note that, under this assumption also the matrix $\psi(1)$ is stochastic. Obviously, the statement of the lemma holds when $A_J = 0$ and $C_J = 0$, so we assume $A_J + C_J \neq 0$ from now on. Assuming $J = \{1, \ldots, k\}$ after a suitable reordering, we can partition

$$\varphi(\lambda) = \begin{bmatrix} \varphi_J(\lambda) & 0 \\ \star & \star \end{bmatrix}$$

and hence an eigenvalue of $\varphi_J(\lambda)$ is also an eigenvalue of $\varphi(\lambda)$.

Let us consider the perturbed matrix polynomial $\varphi_{J,\epsilon}(\lambda) := \lambda^2(A_J - \epsilon E_A) + \lambda B_J + (C_J - \epsilon E_C)$ for $\epsilon > 0$ and Boolean matrices E_A, E_C with the sparsity pattern of A and C, respectively. Because of $A_J + C_J \neq 0$, the matrix $A_J - \epsilon E_A + B_J + C_J - \epsilon E_C + I$ is substochastic for ϵ sufficiently close to 0. Using again Rouché's theorem, this ensures that $\varphi_{J,\epsilon}(\lambda)$ has the property (7). By continuity, the eigenvalue functions of $\varphi_{J,\epsilon}(\lambda)$ do not cross the unit circle as $\epsilon \to 0$ and, in turn, $\varphi_J(\lambda) = \lim_{\epsilon\to 0} \varphi_{J,\epsilon}(\lambda)$ has n eigenvalues inside or on the unit circle and n eigenvalues outside or on the unit circle. Because the simple eigenvalue 1 is the only eigenvalue of $\varphi(\lambda)$ on the unit circle and the same property holds for $\varphi_J(\lambda)$, this completes the proof. $\square$

We remark that the eigenvalue assumption on $\varphi(\lambda)$ in Lemma 5.4 can be relaxed to the assumption that 1 is a simple eigenvalue (admitting possibly other eigenvalues on the unit circle), provided that $B_J^{-1}(A_J + C_J)$ is primitive and $A_J \circ C_J \neq 0$, where $\circ$ indicates the componentwise product.

Often, the probabilistic model requires bounded transitions in the phase coordinate as well. This translates into a band structure in the matrices A, B and C. For example, in the case of *double QBD processes* (DQBD) [39] the coefficients are all tridiagonal, see also Figure 2. We test Algorithm 6 on instances of DQBD with increasing size n. In particular, we choose the entries of the

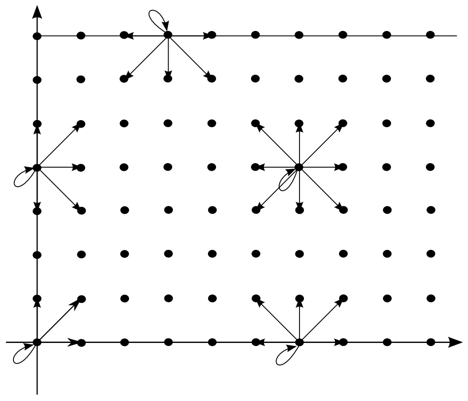

Figure 2: Transitions of a double quasi-birth-death process on $\mathbb{N} \times \{0, \dots n-1\}$.

	Algorithm 6			CR		
n	Time	Res	HODLR rank	Time	Res	HODLR rank
1,024	1.84	$6.45 \cdot 10^{-9}$	15	3	$7.04 \cdot 10^{-9}$	20
2,048	3.2	$3.83 \cdot 10^{-9}$	16	10.37	$4.31 \cdot 10^{-9}$	18
4,096	8.68	$5.08 \cdot 10^{-9}$	17	23.55	$6.82 \cdot 10^{-9}$	21
8,192	22.27	$5.18 \cdot 10^{-9}$	18	68.16	$3.87 \cdot 10^{-9}$	20
16,384	55.54	$6.10 \cdot 10^{-9}$	18	160.28	$5.54 \cdot 10^{-9}$	22
32,768	137.09	$6.67 \cdot 10^{-9}$	18	429.2	$8.61 \cdot 10^{-9}$	22

Table 4: Execution times (in seconds) and residuals for the divide-and conquer-method and cyclic reduction applied to the example from Section 5.2.

3 central diagonals of A, B and C randomly from a uniform distribution on $[0, 1]$. We divide each row of the three matrices by the corresponding entry in $(A + B + C)\mathbf{e}$, in order to make $A + B + C$ row stochastic. Finally, we subtract the identity matrix from B:

$$A = \begin{bmatrix} a_1^{(0)} & a_1^{(1)} & & \\ a_1^{(-1)} & \ddots & \ddots & \\ & \ddots & \ddots & a_{n-1}^{(1)} \\ & & a_{n-1}^{(-1)} & a_n^{(0)} \end{bmatrix}, \qquad B = \begin{bmatrix} b_1^{(0)} & b_1^{(1)} & & \\ b_1^{(-1)} & \ddots & \ddots & \\ & \ddots & \ddots & b_{n-1}^{(1)} \\ & & b_{n-1}^{(-1)} & b_n^{(0)} \end{bmatrix} - I,$$

$$C = \begin{bmatrix} c_1^{(0)} & c_1^{(1)} & & \\ c_1^{(-1)} & \ddots & \ddots & \\ & \ddots & \ddots & c_{n-1}^{(1)} \\ & & c_{n-1}^{(-1)} & c_n^{(0)} \end{bmatrix}.$$

In Table 4 we compare the performance of Algorithm 6 with the method in [16] that combines cyclic reduction — Algorithm 5 — with HODLR arithmetic. Both methods can handle large values

	Algorithm 6			CR		
n	Time	Res	HODLR rank	Time	Res	HODLR rank
1,024	0.48	$7.76 \cdot 10^{-9}$	2	0.61	$2.38 \cdot 10^{-8}$	4
2,048	1.2	$7.76 \cdot 10^{-9}$	2	1.28	$2.38 \cdot 10^{-8}$	4
4,096	2.95	$7.76 \cdot 10^{-9}$	2	2.97	$2.38 \cdot 10^{-8}$	4
8,192	7.18	$7.76 \cdot 10^{-9}$	2	6.95	$2.38 \cdot 10^{-8}$	4
16,384	18.05	$7.76 \cdot 10^{-9}$	2	14.75	$2.38 \cdot 10^{-8}$	4
32,768	41.14	$7.76 \cdot 10^{-9}$	2	34.57	$2.38 \cdot 10^{-8}$	4

Table 5: Execution times (in seconds) and residuals for the divide-and-conquer method and cyclic reduction applied to the example from Section 5.3.

for n and return solutions of comparable accuracy, measured in terms of Res $:= \|AX^2 + BX + C\|_2$. However, the divide-and-conquer method provides a significant speed up; it is about 3 times faster than the competitor for $n \geqslant 4096$.

5.3 Results for UQMEs from damped mass-spring system

Another application of UQME is the solution of the quadratic eigenvalue problem $(\lambda^2 A + \lambda B + C)v = 0$, arising in the analysis of damped structural systems and vibration problems [21, 29, 32]. After having determined the solution X of (5), the quadratic eigenvalue problem reduces to two linear eigenvalue problems: the one associated with X and the generalized eigenproblem $(AX + B)v = -\lambda Av$.

We repeat the experiments from Section 5.2 for the UQME associated with a quadratic eigenvalue problem from a damped mass spring system considered in [49, Example 2]. The $n \times n$ coefficients of the UQME are given by

$$A = I, \qquad B = \begin{bmatrix} 20 & -10 & & & \\ -10 & 30 & -10 & & \\ & \ddots & \ddots & \ddots & \\ & & -10 & 30 & -10 \\ & & & -10 & 20 \end{bmatrix}, \qquad C = \begin{bmatrix} 15 & -5 & & & \\ -5 & 15 & -5 & & \\ & \ddots & \ddots & \ddots & \\ & & -5 & 15 & -5 \\ & & & -5 & 15 \end{bmatrix}.$$

The results reported in Table 5 confirm the good scalability and accuracy of both methods. The solution exhibits a very low HODLR rank and cyclic reduction needs only 2–3 iterations to converge. As a consequence, cyclic reduction is faster than Algorithm 6 on larger instances of this example.

6 Conclusions

We have proposed novel Krylov subspace methods for updating the solution of continuous-time algebraic Riccati equations and unilateral quadratic matrix equations whose coefficients are subject to low-rank modifications. We have provided theoretical insights into the low-rank and stability properties of the solutions to the involved correction equations. This has led us to design novel divide-and-conquer methods for quadratic equations with large-scale coefficients featuring hierarchical low-rank structures. Our methods have linear polylogarithmic complexity and often outperform existing techniques, sometimes significantly. The applications highlighted in this work include quasi-birth–death processes and damped mass-spring systems.

Acknowledgements. During the larger part of the work on this article, the second author PK was affiliated with the Max Planck Institute in Magdeburg.

References

[1] J. Abels and P. Benner. CAREX - A Collection of Benchmark Examples for Continuous-Time Algebraic Riccati Equations (Version 2.0). SLICOT working note 1999-14, 1999.

[2] S. Ambikasaran and E. Darve. An $\mathcal{O}(N \log N)$ fast direct solver for partial hierarchically semi-separable matrices: with application to radial basis function interpolation. *J. Sci. Comput.*, 57(3):477–501, 2013.

[3] U. Baur and P. Benner. Factorized solution of Lyapunov equations based on hierarchical matrix arithmetic. *Computing*, 78(3):211–234, 2006.

[4] B. Beckermann and L. Reichel. Error estimates and evaluation of matrix functions via the Faber transform. *SIAM J. Numer. Anal.*, 47(5):3849–3883, 2009.

[5] B. Beckermann and A. Townsend. On the singular values of matrices with displacement structure. *SIAM J. Matrix Anal. Appl.*, 38(4):1227–1248, 2017.

[6] P. Benner, Z. Bujanović, P. Kürschner, and J. Saak. A numerical comparison of solvers for large-scale, continuous-time algebraic Riccati equations. Technical Report 1811.00850, arXiv, 2018.

[7] P. Benner, Z. Bujanović, P. Kürschner, and J. Saak. RADI: A low-rank ADI-type algorithm for large scale algebraic Riccati equations. *Numer. Math.*, 138(2):301–330, 2018.

[8] P. Benner, R. Byers, V. Mehrmann, and H. Xu. Robust numerical methods for robust control. Technical Report 06-2004, Institut für Mathematik, TU Berlin, 2004.

[9] Peter Benner, Matthias Bollhöfer, Daniel Kressner, Christian Mehl, and Tatjana Stykel. *Numerical Algebra, Matrix Theory, Differential-Algebraic Equations and Control Theory*. Springer International Publishing, 2015.

[10] Peter Benner, Jing-Rebecca Li, and Thilo 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.

[11] Peter Benner and Jens Saak. Numerical solution of large and sparse continuous time algebraic matrix Riccati and Lyapunov equations: a state of the art survey. *GAMM-Mitt.*, 36(1):32–52, 2013.

[12] Mario Berljafa, Steven Elsworth, and Stefan Güttel. A rational Krylov toolbox for MATLAB. MIMS EPrint 2014.56, Manchester Institute for Mathematical Sciences, The University of Manchester, UK, 2014.

[13] D. A. Bini, G. Latouche, and B. Meini. *Numerical methods for structured Markov chains*. Numerical Mathematics and Scientific Computation. Oxford University Press, New York, 2005. Oxford Science Publications.

[14] Dario A. Bini, Paola Favati, and Beatrice Meini. A compressed cyclic reduction for QBD processes with low-rank upper and lower transitions. In *Matrix-analytic methods in stochastic models*, volume 27 of *Springer Proc. Math. Stat.*, pages 25–40. Springer, New York, 2013.

[15] Dario A. Bini, Bruno Iannazzo, and Beatrice Meini. *Numerical solution of algebraic Riccati equations*, volume 9 of *Fundamentals of Algorithms*. SIAM Publications, Philadelphia, PA, 2012.

[16] Dario A. Bini, Stefano Massei, and Leonardo Robol. Efficient cyclic reduction for quasi-birth-death problems with rank structured blocks. *Appl. Numer. Math.*, 116:37–46, 2017.

[17] Dario A. Bini, Stefano Massei, and Leonardo Robol. On the decay of the off-diagonal singular values in cyclic reduction. *Linear Algebra Appl.*, 519:27–53, 2017.

[18] Dario A. Bini and Beatrice Meini. The cyclic reduction algorithm: from Poisson equation to stochastic processes and beyond. In memoriam of Gene H. Golub. *Numer. Algorithms*, 51(1):23–60, 2009.

[19] E. K.-W. Chu, H.-Y. Fan, and W.-W. Lin. A structure-preserving doubling algorithm for continuous-time algebraic Riccati equations. *Linear Algebra Appl.*, 396:55–80, 2005.

[20] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to algorithms*. MIT Press, Cambridge, MA, third edition, 2009.

[21] Biswa Nath Datta. *Numerical linear algebra and applications*. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, second edition, 2010.

[22] V. Druskin and V. Simoncini. Adaptive rational Krylov subspaces for large-scale dynamical systems. *Syst. Cont. Lett.*, 60(8):546–560, 2011.

[23] L. Grasedyck, W. Hackbusch, and B. N. Khoromskij. Solution of large scale algebraic matrix Riccati equations by use of hierarchical matrices. *Computing*, 70(2):121–165, 2003.

[24] Chun-Hua Guo, Nicholas J. Higham, and Françoise Tisseur. Detecting and solving hyperbolic quadratic eigenvalue problems. *SIAM J. Matrix Anal. Appl.*, 30(4):1593–1613, 2008/09.

[25] Xiao-Xia Guo, Wen-Wei Lin, and Shu-Fang Xu. A structure-preserving doubling algorithm for nonsymmetric algebraic Riccati equation. *Numer. Math.*, 103(3):393–412, 2006.

[26] Stefan Güttel. *Rational Krylov methods for operator functions*. PhD thesis, TU Freiberg, 2010.

[27] Wolfgang Hackbusch. *Hierarchical matrices: algorithms and analysis*, volume 49 of *Springer Series in Computational Mathematics*. Springer, Heidelberg, 2015.

[28] M. Heyouni. Extended Arnoldi methods for large low-rank Sylvester matrix equations. *Appl. Numer. Math.*, 60(11):1171–1182, 2010.

[29] Nicholas J. Higham and Hyun-Min Kim. Numerical analysis of a quadratic matrix equation. *IMA J. Numer. Anal.*, 20(4):499–519, 2000.

[30] Daniel Kressner, Stefano Massei, and Leonardo Robol. Low-rank updates and a divide-and-conquer method for linear matrix equations. *arXiv preprint arXiv:1712.04349*, 2017.

[31] P. Lancaster and L. Rodman. *The Algebraic Riccati Equation.* Oxford University Press, Oxford, 1995.

[32] Peter Lancaster. *Lambda-matrices and vibrating systems.* Dover Publications, Inc., Mineola, NY, 2002. Reprint of the 1966 original [Pergamon Press, New York; MR0210345 (35 #1238)].

[33] N. Lang, H. Mena, and J. Saak. On the benefits of the LDL^T factorization for large-scale differential matrix equation solvers. *Linear Algebra Appl.*, 480:44–71, September 2015.

[34] G. Latouche and V. Ramaswami. *Introduction to matrix analytic methods in stochastic modeling.* ASA-SIAM Series on Statistics and Applied Probability. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA; American Statistical Association, Alexandria, VA, 1999.

[35] A. Locatelli. *Optimal Control: An Introduction.* Basel, Switzerland, 2001.

[36] S. Massei and L. Robol. MATLAB toolbox for HODLR and HSS matrices: `hm-toolbox`, 2017. Available at https://github.com/numpi/hm-toolbox.

[37] V. Mehrmann and E. Tan. Defect correction methods for the solution of algebraic Riccati equations. *IEEE Trans. Automat. Control*, 33(7):695–698, 1988.

[38] A. Melman. Generalization and variations of Pellet's theorem for matrix polynomials. *Linear Algebra Appl.*, 439(5):1550–1567, 2013.

[39] Masakiyo Miyazawa. Tail decay rates in double QBD processes and related reflected random walks. *Math. Oper. Res.*, 34(3):547–575, 2009.

[40] Axel Ruhe. The rational Krylov algorithm for nonsymmetric eigenvalue problems. III: Complex shifts for real matrices. *BIT Numerical Mathematics*, 34(1):165–176, 1994.

[41] Axel Ruhe. Rational Krylov: A practical algorithm for large sparse nonsymmetric matrix pencils. *SIAM J. Sci. Comput.*, 19(5):1535–1551, 1998.

[42] D. L. Russell. *Mathematics of Finite-Dimensional Control Systems*, volume 43 of *Lect. Notes Pure Appl. Math.* Marcel Dekker Inc., New York, 1979.

[43] E. Seneta. *Non-negative matrices and Markov chains.* Springer Series in Statistics. Springer, New York, 2006. Revised reprint of the second (1981) edition [Springer-Verlag, New York; MR0719544].

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

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

[46] V. Simoncini. Analysis of the rational Krylov subspace projection method for large-scale algebraic Riccati equations. *SIAM J. Matrix Anal. Appl.*, 37(4):1655–1674, 2016.

[47] Valeria Simoncini, Daniel B Szyld, and Marlliny Monsalve. On two numerical methods for the solution of large-scale algebraic Riccati equations. *IMA Journal of Numerical Analysis*, 34(3):904–920, 2013.

[48] Gerhard Starke. Near-circularity for the rational Zolotarev problem in the complex plane. *J. Approx. Theory*, 70(1):115–130, 1992.

[49] F. Tisseur. Backward error and condition of polynomial eigenvalue problems. *Linear Algebra Appl.*, 309(1-3):339–361, 2000.

[50] J. C. Willems. Least Squares Stationary Optimal Control and the Algebraic Riccati Equation. *IEEE Trans. Autom. Control*, 16:621–634, 1971.

Recent publications:

INSTITUTE of MATHEMATICS
MATHICSE Group

Ecole Polytechnique Fédérale (EPFL)

CH-1015 Lausanne

2019

01.2019 ASSYR ABDULLE, DOGHONAY ARJMAND, EDOARDO PAGANONI
Exponential decay of the resonance error in numerical homogenization via parabolic and elliptic cell problems

02.2019 CESARE BRACCO, CARLOTTA GIANNELLI, MARIO KAPL, RAFAEL VÁZQUEZ
Isogeometric analysis with C1 hierarchical functions on planar two-patch geometries

03.2019 SOPHIE HAUTPHENNE, STEFANO MASSEI
A low-rank technique for computing the quasi-stationary distribution of subcritical Galton-Watson processes

04.2019 ALESSIA PATTONA, JOHN-ERIC DUFOUR, PABLO ANTOLIN, ALESSANDRO REALI,
Fast and accurate elastic analysis of laminated composite plates via isogeometric collocation and an equilibrium-based stress recovery approach

05.2019 ALICE CORTINOVIS, DANIEL KRESSNER, STEFANO MASSEI
On maximum volume submatrices and cross approximation for symmetric semidefinite and diagonally dominant matrices

06.2019 ANNALISA BUFFA, RICCARDO PUPPI, RAFAEL VAZQUEZ
A minimal stabilization procedure for isogeometric methods on trimmed geometries

07.2019 DANIEL KRESSNER, PATRICK KÜRSCHNER, STEFANO MASSEI
Low-rank updates and divide-and-conquer methods for quadratic matrix equations

***