Alma Mater Studiorum · Università di Bologna

DOTTORATO DI RICERCA IN MATEMATICA
CICLO XXX

Settore Concorsuale di afferenza: 01/A5

Settore Scientifico disciplinare: MAT/08

NUMERICAL SOLUTION OF LARGE-SCALE LINEAR MATRIX EQUATIONS

Presentata da: Davide Palitta

Coordinatore Dottorato
Chiar.ma Prof.ssa
Giovanna Citti

Relatore
Chiar.ma Prof.ssa
Valeria Simoncini

Esame finale 2018

Al mio papà.

Contents

Introduction

In this thesis, the numerical treatment of the continuous-time Sylvester equation

$$AX + XB + C = 0, \quad A \in \mathbb{R}^{n_A \times n_A}, B \in \mathbb{R}^{n_B \times n_B}, C \in \mathbb{R}^{n_A \times n_B}, \tag{i}$$

is explored. Sylvester equations arise in different settings, such as problems of control [14, 1], discretization of partial differential equations (PDEs) [82, 26], block-diagonalization [45, Chapter 7.1.4], and many others. See, e.g., [99] and the references therein. Very recent applications include optimization techniques for neural networks [75], the analysis of biological systems [117], and the reconstruction of dynamical networks [108].

Due to its important role in control theory, we also address the case of the Lyapunov equation

$$AX + XA^T + C = 0, \quad A, C \in \mathbb{R}^{n \times n}, C = C^T, \tag{ii}$$

that is equation (i) with $B = A^T$ and C symmetric. For instance, consider the continuous-time linear system

$$\Sigma : \begin{cases} \dot{x}(t) = Ax(t) + C_{input}u(t), \\ y(t) = C_{output}^T x(t), \quad x(0) = 0, \end{cases} \tag{iii}$$

where x is the model state, u is the input and y the output; A, C_{input} and C_{output} are time-invariant. Then, assuming A *stable*, that is all its eigenvalues have negative real part, the solution P and Q of the following Lyapunov equations

$$AP + PA^T + C_{input}C_{input}^T = 0, \quad \text{and} \quad A^TQ + QA + C_{output}C_{output}^T = 0,$$

are called the *controllability* and *observability Gramians* respectively, and they provide useful information for measuring, e.g., the energy transfers in the system (iii). See, e.g., [1, Section 4.3.1]. It can also be verified that if A is stable and the right-hand side C in (ii) is positive (semi)definite then X is positive (semi)definite. See, e.g., [101]. This means that both the controllability and observability Gramians are positive semidefinite matrices.

Alternatively, when C_{input} and C_{output} have the same number of columns, we can compute the *cross-Gramian* W by solving the Sylvester equation

$$AW + WA + C_{input}C_{output}^T = 0,$$

which furnishes information about both the controllability and observability of the system. If C_{input} and C_{output} have a single column, or if A is symmetric and C_{input}, C_{output} are such that $C_{output}^T(zI - A)^{-1}C_{input}$ is symmetric, where $(zI - A)^{-1}$ is the resolvent of A, it can be shown that $W^2 = PQ$, so that the eigenvalues of W are the square root of the eigenvalues of PQ. See, e.g., [39]. Usually, the eigenvalues of PQ are called the *Hankel singular values* of Σ and they satisfy important invariance properties. See, e.g., [1, 40] for a detailed discussion.

In what follows, we will focus on the algebraic problems (i) and (ii) with particular attention to the case when the matrices involved are of large dimensions. As pointed out in [99, Section 1], the definition of *large-scale problem* is architecture dependent. However, throughout the thesis, we say that a problem is large if the size of the data is of the order of 10^4 or greater. For smaller problems, various decomposition-based methods have been developed and they can be used on standard laptops with moderate energy consumption. See, e.g., Section 1.3.

A characteristic aspect of the large-scale setting is that although data are sparse, the solution X is in general dense so that storing it may be unfeasible. Therefore, it is necessary that the solution allows for a memory-saving approximation, that is there exists a matrix $\widetilde{X}$, $\widetilde{X} \approx X$ which can be cheaply stored. Moreover, efficient algorithms for the computation of $\widetilde{X}$ must be designed.

An extensive literature treats the case of equations (i)–(ii) with low-rank right-hand side[*] $C = C_1 C_2^T$, $C_1 \in \mathbb{R}^{n_A \times s}$, $C_2 \in \mathbb{R}^{n_B \times s}$, $s \ll n_A, n_B$. This assumption, together with certain hypotheses on the spectral distribution of A and B, is a sufficient condition for proving a fast –in many cases exponential – decay in the singular values of X, see, e.g., the discussion at the beginning of Chapter 2, [86, 95] and the recent work [4] about Lyapunov equations with nonnormal coefficients. This decay justifies the search for a low-rank approximation $\widetilde{X} = Z_1 Z_2^T$ to X so that only the two tall matrices $Z_1, Z_2 \in \mathbb{R}^{n \times t}$, $t \ll n$, are actually computed and stored, remarkably reducing the storage demand. This is the task of the so-called *low-rank methods* and a large amount of work in this direction has been carried out in the past few years. Very different algorithms belong to this family such as projection methods [97, 35], low-rank ADI [15, 13], low-rank sign function methods [6, 7]. See the thorough presentation in [99] and the references therein. In Section 2.1 we will recall the general framework of projection methods for Sylvester and Lyapunov equations with low-rank right-hand side.

The case of large-scale Sylvester and Lyapunov equations with not necessarily low rank right-hand side has not been deeply analyzed so far and efficient numerical methods are still lacking in the literature. In this thesis we aim to significantly contribute to this open problem by introducing solution methods for equations with different structure in their data. In particular, in Chapter 3 we address the case when the coefficient matrices and the right-hand side are generally banded. For this kind of equations an early contribution has been given by Haber and Verhaegen in [51]. We will show that the memory-saving approximation $\widetilde{X}$ to X preserves the banded structure of the data and its bandwidth depends on the conditioning of the coefficient matrices. For ill-conditioned problems it will be illustrated that X can be represented by a pair (X_B, S_m), $X \approx X_B + S_m S_m^T$, where X_B is banded and S_m is low-rank so that moderate allocation requirements are still possible.

In Chapter 4 a more general structure of the data will be considered. In particular, we will suppose A, B and C to be matrices with low quasiseparable-rank. Roughly speaking, a matrix is said to be quasiseparable if its off-diagonal blocks are low-rank matrices, and the quasiseparable rank is defined as the maximum of the ranks of the off-diagonal blocks. Therefore, this structure can be viewed as a generalization of the case of (standard) low-rank and banded matrices.

We further consider equations of the form

$$AX + XB + \sum_{i=1}^{p} N_i X M_i + C_1 C_2^T = 0, \tag{iv}$$

$A, N_i \in \mathbb{R}^{n_A \times n_A}$, $B, M_i \in \mathbb{R}^{n_B \times n_B}$, $C_1 \in \mathbb{R}^{n_A \times s}$, $C_2 \in \mathbb{R}^{n_B \times s}$, $s \ll n$, which are sometimes

[*] In case of the Lyapunov equation, $C_1 = C_2$.

referred to as *generalized Sylvester equations* since they can be viewed as the sum of a Sylvester operator $\mathcal{L}(X) = AX + XB$ and a linear operator $\Pi(X) = \sum_{i=1}^{p} N_i X M_i$.

Many problems arising from the discretization of PDEs can be formulated as generalized Sylvester equations [90, 82]. Moreover, the *generalized Lyapunov equation*, which corresponds to equation (iv) with $B = A^T$, $M_i = N_i$ and $C_1 = C_2$, arises in model order reduction of bilinear and stochastic systems. See e.g. [12, 31, 11] and references therein. More precisely, if we consider the bilinear system

$$\Sigma_{Bil} : \begin{cases} \dot{x}(t) = Ax(t) + \sum_{i=1}^{p} N_i x(t) u_i(t) + C_{input} u(t), \\ y(t) = C_{output}^T x(t), \quad x(0) = 0, \end{cases} \tag{v}$$

where $A, N_i \in \mathbb{R}^{n\times n}$, $C_{input} \in \mathbb{R}^{n\times p}$, $C_{output} \in \mathbb{R}^{n\times s}$, $u(t) = [u_1(t), \ldots, u_p(t)] \in \mathbb{R}^p$ and $y(t) \in \mathbb{R}^s$, the solutions of the generalized Lyapunov equations

$$AP + PA^T + \sum_{i=1}^{p} N_i P N_i^T + C_{input} C_{input}^T = 0, \quad \text{and} \quad A^T Q + QA + \sum_{i=1}^{p} N_i^T Q N_i + C_{output} C_{output}^T = 0,$$

are the *generalized observability* and *reachability Gramian* associated with (v) and they carry important information about the $\mathcal{H}_2$-norm of Σ_{Bil}; see, e.g., [10].

It is not easy to explicitly write down the necessary and sufficient conditions for the well-posedness of equation (iv) in terms of its coefficient matrices. However, as we recall in Section 5.2, the assumption $\rho(\mathcal{L}^{-1}\Pi) < 1$, where $\rho(\cdot)$ denotes the (operator) spectral radius, is a sufficient condition for the existence of a unique solution X. This hypothesis means that $\mathcal{L}$ consists of the dominant part of the overall operator and many numerical algorithms are designed in terms of the regular splitting $\mathcal{L} + \Pi$. See, e.g., [31, 96, 11]. We also rely on the assumption $\rho(\mathcal{L}^{-1}\Pi) < 1$ and, under some further hypotheses on the coefficient matrices, we propose novel approximation spaces for the solution of (iv) by projection. To the best of our knowledge, effective projection methods for generalized Sylvester and Lyapunov equations were lacking in the literature as appropriate spaces have not been identified yet. In Section 6.2 we try to fill this gap for a particular class of problems that commonly arise in the aforementioned applications.

Notation and initial definitions

We now introduce some notation and we recall the definition and some properties of common linear algebra tools used throughout the thesis.

Unless differently stated, all the matrices invoked in this thesis are real and square.

$\boldsymbol{\alpha}$	$s \times s$ matrices	
$(X)_{i,j}$	(i,j)-th entry of the matrix X	
$(x)_k$	k-th component of the vector x	
X^T	transpose of X	
X^*	conjugate transpose of X	
β_T	bandwidth of the symmetric matrix T	$(T)_{i,j} = 0$ for $\lvert i-j\rvert > \beta_T$
$\lambda_{\max}(T)$	the largest (in magnitude) eigenvalue of T	
$\lambda_{\min}(T)$	the smallest (in magnitude) eigenvalue of T	
$\Lambda(T)$	spectrum of T	
$\langle A, B\rangle_F$	matrix inner product	$\langle A, B\rangle_F := \operatorname{trace}(B^T A)$
$\lVert A\rVert_F$	Frobenius norm	$\lVert A\rVert_F^2 := \langle A, A\rangle_F$
$\lVert A\rVert_2$	induced Euclidean norm	$\lVert A\rVert_2 := \sup_{\lVert x\rVert_2=1} \lVert Ax\rVert_2$
$\kappa(A)$	spectral condition number of T	$\kappa(A) := \lVert A\rVert_2 \lVert A^{-1}\rVert_2$
$\lVert A\rVert_{\max}$	max norm	$\lVert A\rVert_{\max} := \max_{i,j} \lvert(A)_{i,j}\rvert$
$\lVert\cdot\rVert$	any submultiplicative matrix norm	
$\mathcal{L}$	generic linear and continuous operator $\mathcal{L} : \mathbb{R}^{n\times n} \to \mathbb{R}^{n\times n}$	
$\lVert\mathcal{L}\rVert$	operator norm induced by the matrix norm $\lVert\cdot\rVert$	$\lVert\mathcal{L}\rVert := \sup_{\lVert A\rVert=1} \lVert\mathcal{L}(A)\rVert$
I_n	identity matrix of order n[†]	
O_n	zero matrix of order n[†]	
e_i	i-th column of I	
$E_i \in \mathbb{R}^{sm\times s}$	i-th block of s columns of I_{sm}	
$[A, B]$	matrix obtained by putting the matrix B next to the matrix A[††]	
$\operatorname{vec}(X)$	n^2-vector obtained by stacking the columns of $X \in \mathbb{R}^{n\times n}$ one below the other	
$\operatorname{Re}(\alpha)$, $\operatorname{Im}(\alpha)$	the real and imaginary part of $\alpha \in \mathbb{C}$	
$\mathbb{C}_-$, $\mathbb{C}_+$	left and right open half complex plane	
$\operatorname{Range}(S)$	vector space generated by the columns of the matrix S	
$\operatorname{span}(\mathcal{S})$	vector space generated by the elements in the set $\mathcal{S}$	
$\mathcal{O}(\cdot)$	big-O notation	

Table 1: Table of symbols and notation.

[†]The subscript is omitted whenever the dimension is clear from the context.

[††]Both having conforming dimensions.

Definition I ([59]). *For given matrices $A \in \mathbb{R}^{n_A \times m_A}$ and $B \in \mathbb{R}^{n_B \times m_B}$ the* Kronecker product *is defined as*

$$A \otimes B = \begin{bmatrix} (A)_{1,1}B & (A)_{1,2}B & \cdots & (A)_{1,m_A}B \\ (A)_{2,1}B & (A)_{2,2}B & \cdots & (A)_{2,m_A}B \\ \vdots & & & \vdots \\ (A)_{n_A,1}B & (A)_{n_A,2}B & \cdots & (A)_{n_A,m_A}B \end{bmatrix} \in \mathbb{R}^{n_A n_B \times m_A m_B}.$$

Lemma II ([59]). *The Kronecker product defined in Definition I has the following properties:*

(1) $vec(AXB) = (B^T \otimes A)vec(X)$;

(2) If $A \in \mathbb{R}^{n_A \times n_A}$, $B \in \mathbb{R}^{n_B \times n_B}$ *and* $\Lambda(A) = \{\lambda_i\}_{i=1}^{n_A}$, $\Lambda(B) = \{\psi_j\}_{j=1}^{n_B}$, *then*

$$\Lambda(A \otimes B) = \{\lambda_i \psi_j \mid i = 1, \dots, n_A,\ j = 1, \dots, n_B\},$$

and

$$\Lambda(I_{n_A} \otimes B + A \otimes I_{n_B}) = \{\lambda_i + \psi_j \mid i = 1, \dots, n_A,\ j = 1, \dots, n_B\}.$$

Any matrix equations can be recast in terms of a large linear system by exploiting the properties of the Kronecker product. Indeed, by using (1) in Lemma II, we can rewrite equation (i) as

$$(B^T \otimes I_{n_A} + I_{n_B} \otimes A)\mathrm{vec}(X) = -\mathrm{vec}(C).$$

This will be used in the following Section to determine the necessary and sufficient conditions for the well-posedness of the Sylvester and Lyapunov equations.

Definition III ([59]). *For a given matrix $A \in \mathbb{R}^{n \times n}$, its* field of values *is defined as the following set of complex numbers*

$$W(A) := \{z \in \mathbb{C} \mid z = x^* A x,\ x \in \mathbb{C}^n,\ x^* x = 1\}. \tag{vi}$$

Lemma IV ([59]). *Some of the relevant properties of the field of values are the following.*

(1) For any $A \in \mathbb{R}^{n \times n}$, $W(A)$ is a convex and compact set of $\mathbb{C}$.

(2) For any $A \in \mathbb{R}^{n \times n}$ and $\alpha \in \mathbb{C}$,

$$W(A + \alpha I) = W(A) + \alpha, \quad \text{and} \quad W(\alpha A) = \alpha W(A).$$

(3) For any $A \in \mathbb{R}^{n \times n}$,

$$\Lambda(A) \subset W(A).$$

(4) For any $A, B \in \mathbb{R}^{n \times n}$,

$$W(A + B) \subset W(A) + W(B).$$

(5) For any $A, Q \in \mathbb{R}^{n \times n}$, Q unitary, that is $QQ^* = I$,

$$W(Q^* A Q) = W(A).$$

Definition V. *We say that a matrix $A \in \mathbb{R}^{n \times n}$ is* positive definite, *$A > 0$, if*

$$Re(x^* A x) > 0, \quad \text{for all } x \in \mathbb{C}^n,\ x^* x = 1,$$

and positive semidefinite, *$A \geq 0$, if*

$$Re(x^* A x) \geq 0, \quad \text{for all } x \in \mathbb{C}^n,\ x^* x = 1.$$

The matrix A is negative (semi)definite, *$A < 0$ ($A \leq 0$), if $-A$ is positive (semi)definite.*

Notice that A is positive (negative) definite if $W(A)$ is strictly contained in $\mathbb{C}_+$ ($\mathbb{C}_-$).

Chapter 1

Preliminaries

1.1 Existence and uniqueness of the solution

The well-posedness of the continuous-time Sylvester equation

$$AX + XB + C = 0, \quad A \in \mathbb{R}^{n_A \times n_A}, B \in \mathbb{R}^{n_B \times n_B}, C \in \mathbb{R}^{n_A \times n_B}, \tag{1.1.1}$$

can be guaranteed in different ways. In the early 50's Roth showed in [92] that equation (1.1.1) admits a solution if and only if the matrices

$$\begin{bmatrix} A & C \\ 0 & -B \end{bmatrix} \quad \text{and} \quad \begin{bmatrix} A & 0 \\ 0 & -B \end{bmatrix},$$

are similar and the similarity transformation is given by

$$\begin{bmatrix} I & X \\ 0 & I \end{bmatrix},$$

where X is the solution to (1.1.1).

Alternatively, as already stated, equation (1.1.1) can be recast in terms of an $n_A n_B \times n_A n_B$ linear system

$$\mathcal{A}\mathrm{vec}(X) = -\mathrm{vec}(C), \quad \mathcal{A} := I_{n_B} \otimes A + B \otimes I_{n_A},$$

which admits a unique solution for every $\mathrm{vec}(C) \neq 0$ if and only if the matrix $\mathcal{A}$ is nonsingular. Thanks to the property (2) in Lemma II, if $\Lambda(A) = \{\lambda_i\}_{i=1}^{n_A}$, $\Lambda(B) = \{\psi_j\}_{j=1}^{n_B}$, the spectrum of $\mathcal{A}$ can be written as $\Lambda(\mathcal{A}) = \{\lambda_i + \psi_j \,|\, i = 1, \ldots, n_A, j = 1, \ldots, n_B\}$, so that the condition $\Lambda(A) \cap \Lambda(-B) = \varnothing$ guarantees the nonsingularity of $\mathcal{A}$. In the following we always assume that the latter condition is satisfied and therefore equation (1.1.1) has a unique solution X.

A similar reasoning leads to the condition $\lambda_i + \lambda_j \neq 0$ for all $i, j = 1, \ldots, n_A$ for the well-posedness of the continuous-time Lyapunov equation in (ii).

$$AX + XA^T + C = 0, \quad C = C^T \tag{1.1.2}$$

Notice that the symmetry of C implies that X is symmetric. As already mentioned, in control theory the matrix A is usually supposed to be stable, that is all its eigenvalues have negative real part, see, e.g., [1], and this is a sufficient condition for the existence and uniqueness of X.

Numerical schemes for large-scale problems may require additional assumptions on the coefficients in order to be employed in the solution process. For instance, projection methods need $W(A) \cap W(-B) = \varnothing$, and $W(A) \subset \mathbb{C}_-$ in the case of the Lyapunov equation. See Sections 2.1–2.1.1.

1.2 Solution in closed-form

The solution X to (1.1.1) admits different closed-form representations. Here we report some of the main ones. See, e.g., [99] and [72] for early contributions.

(i) *Integral of resolvents.*

$$X = \frac{1}{4\pi^2} \int_{\Gamma_1} \int_{\Gamma_2} \frac{(\lambda I_{n_A} - A)^{-1} C (\mu I_{n_B} - B)^{-1}}{\lambda + \mu} d\mu d\lambda,$$

where Γ_1, Γ_2 are two closed curves containing the spectrum of A and B respectively.

(ii) *Integral of exponentials.*

$$X = \int_0^{+\infty} e^{At} C e^{Bt} dt, \tag{1.2.3}$$

where e^D denotes the matrix exponential of $D \in \mathbb{R}^{n \times n}$. Here $\Lambda(A)$, $\Lambda(B)$ are supposed to be separated by a vertical line.

(iii) *Finite power sum.* Let $C = C_A C_B^T$ and a_m of degree m be the minimal polynomial of A with respect to C_A, that is the smallest degree monic polynomial such that $a_m(A)C_A = 0$. Similarly, let b_k of degree k be the minimal polynomial of B with respect to C_B. Then

$$X = -\sum_{i=0}^{m-1} \sum_{j=0}^{k-1} \gamma_{i,j} A^i C B^j = -[C_A, AC_A, \dots, A^{m-1}C_A](\gamma \otimes I)[C_B, B^T C_B, \dots, (B^{k-1})^T C_B]^T,$$

where $\gamma \in \mathbb{C}^{m \times k}$ is the solution of the Sylvester equation defined by the companion matrices of a_m and b_k and right-hand side $[1, 0, \dots, 0]^T [1, 0, \dots, 0]$ of conforming dimension.

(iv) *Similarity transformations.* Let A and B be diagonalizable, namely there exist nonsingular matrices U and V such that $U^{-1}AU = \Lambda$, $\Lambda = \operatorname{diag}(\lambda_1, \dots, \lambda_{n_A}) \in \mathbb{C}^{n_A \times n_A}$, and $V^{-1}BV = \Psi$, $\Psi = \operatorname{diag}(\psi_1, \dots, \psi_{n_B}) \in \mathbb{C}^{n_B \times n_B}$. If $\widetilde{C} := U^{-1}CV$, then

$$X = U \widetilde{X} V^{-1}, \quad \text{where } (\widetilde{X})_{i,j} = -\frac{(\widetilde{C})_{i,j}}{\lambda_i + \psi_j}.$$

These closed-forms have inspired several procedures for the numerical solution of equation (1.1.1). For instance, the representation by integral of exponentials (ii) has been used to design the algorithm presented in Section 3.2, and the closed-form (iv) can be employed in the solution of small-scale equations as illustrated in Section 2.2.1.

All the previous representations can be rewritten with straightforward modifications in case of equation (1.1.2).

1.3 Numerical solution of the small-scale problem

Although this thesis deals with numerical methods for large-scale matrix equations, we now recall the Bartels-Stewart algorithm [5], one of the most commonly used methods for problems of moderate dimensions, namely equation (1.1.1) with $n_A, n_B \leq \mathcal{O}(10^3)$. Indeed, the solution of small-scale equations can be encountered as intermediate step in iterative methods for large-scale problems. See Sections 2.1–2.1.1.

The algorithm consists of three phases. First since A and B are supposed to be real, the real Schur decompositions $A = Q_A R_A Q_A^T$ and $B^T = Q_B R_B Q_B^T$ are computed. Conversely to its complex counterpart, the real Schur decomposition allows us to avoid complex arithmetic but the computed matrices R_A and R_B can be quasi-triangular, that is they have 2×2 blocks on the diagonal in case of complex eigenvalues.

Pre and post-multiplying by Q_A^T and Q_B respectively, equation (1.1.1) is transformed into an equivalent one but with (quasi-)triangular coefficient matrices R_A and R_B^T. Thanks to the structure of its coefficients, the new equation can be solved entry-wised by substitution. The procedure is easy to handle in case of triangular R_A and R_B whereas the diagonal 2×2 blocks can be dealt with by solving the correspondly 2×2 Sylvester equation if quasi-triangular R_A and R_B are computed. See [5] for more details.

At the end, a backward transformation is performed to obtain the solution X to the original problem (1.1.1). The overall procedure is summarized in Algorithm 1.1.

Algorithm 1.1: Bartels-Stewart algorithm for the Sylvester matrix equation [5].

input : $A \in \mathbb{R}^{n_A \times n_A}$, $B \in \mathbb{R}^{n_B \times n_B}$, $C \in \mathbb{R}^{n_A \times n_B}$
output: $X \in \mathbb{R}^{n_A \times n_B}$

1 Compute the real Schur decompositions $A = Q_A R_A Q_A^T$, $B^T = Q_B R_B Q_B^T$ with R_A, R_B quasi-upper triangular
2 Solve $R_A \widetilde{X} + \widetilde{X} R_B^T + Q_A^T C Q_B = 0$
3 Compute $X = Q_A \widetilde{X} Q_B^T$

Line 1 is the most expensive step of Algorithm 1.1. Indeed, explicitly computing the Schur decomposition form costs $10n^3$ flops for a matrix of size n. Moreover, also in case of sparse A and B, the matrices Q_A and Q_B are dense and they have to be stored due to the final transformation in line 3. These are the reasons why the application of Algorithm 1.1 is limited to small-scale problems.

To limit computational costs, several variants of the Bartels-Stewart algorithm have been developed. For instance, if A and B have very different dimensions*, say $n_A \ll n_B$, in the algorithm proposed by Golub, Nash and Van Loan [44] the Schur form of B^T is replaced by the Hessenberg decomposition. The computational cost of the latter factorization is $5/3n_B^3$ flops that has to be compared with the $10n_B^3$ flops required by the Schur decomposition.

Algorithm 1.1 can be employed also in case of the Lyapunov equation (1.1.2) and only one Schur decomposition has to be computed in line 1. However, a specifically designed algorithm was proposed in case of A stable and a positive semidefinite C. In this case X is symmetric and positive semidefinite [101] and it thus admits a Cholesky factorization $X = LL^T$. In [55], Hammarling showed how to determine the Cholesky factor L without computing X first. This method turns out to be more robust and accurate than the Bartels-Stewart counterpart especially in case of an ill-conditioned X.

*Although both moderate.

Chapter 2

Sylvester and Lyapunov equations with low-rank right-hand side

In this Chapter we address the numerical solution of large-scale Sylvester equations (1.1.1) with a particular structure in the right-hand side. Indeed, we suppose C to be low-rank, namely $C = C_1 C_2^T$ where $C_1 \in \mathbb{R}^{n_A \times s}$, $C_2 \in \mathbb{R}^{n_B \times s}$, $s \ll n_A, n_B$. Equation (1.1.1) can thus be written as

$$AX + XB + C_1 C_2^T = 0. \tag{2.0.1}$$

Since the right-hand side is low-rank and under certain hypotheses on the spectral distribution of A and B, it can be shown that the singular values of X have a fast decay, see, e.g., [86, 95], thus justifying the search for a low-rank approximation $\widetilde{X} = Z_1 Z_2^T$ to X so that only these two tall matrices are actually computed and stored avoiding the allocation for the dense matrix X that can be unfeasible for large-scale problems.

One of the most commonly used low-rank approximability result for Sylvester equations has been presented in [46]. If $M \in \mathbb{C}^{n \times n}$ is such that $\Lambda(M) \subset \mathbb{C}_-$, then its inverse can be expressed as $M^{-1} = \int_0^\infty e^{tM} dt$ and, for some fixed k, the integral can be approximated by the quadrature formula

$$M^{-1} = \int_0^\infty e^{tM} dt \approx \sum_{j=-k}^{k} w_j e^{t_j M}, \tag{2.0.2}$$

where the weights w_j and nodes t_j are given in [46, Lemma 5] and derived from [103]. In particular, an explicit formula for the approximation error is given by

$$\left\| \int_0^\infty e^{tM} dt - \sum_{j=-k}^{k} w_j e^{t_j M} \right\| \leq K e^{-\pi\sqrt{k}}, \tag{2.0.3}$$

where K is a constant that only depends on the spectrum of M.

In case of the Sylvester equation (2.0.1), the solution X can be explicitly expressed as $\mathrm{vec}(X) = (I_{n_B} \otimes A + B \otimes I_{n_A})^{-1} \mathrm{vec}(-C_1 C_2^T)$ and $(I_{n_B} \otimes A + B \otimes I_{n_A})^{-1}$ can be approximated

by (2.0.2) if $\Lambda(A) \cup \Lambda(B) \subset \mathbb{C}_-$ so that

$$\begin{aligned}\mathrm{vec}(X) &\approx \sum_{j=-k}^{k} w_j e^{t_j (I_{n_B} \otimes A + B \otimes I_{n_A})} \mathrm{vec}(-C_1 C_2^T) = \sum_{j=-k}^{k} w_j e^{t_j I_{n_B} \otimes A} e^{t_j B \otimes I_{n_A}} \mathrm{vec}(-C_1 C_2^T) \\ &= \sum_{j=-k}^{k} w_j \left(I_{n_B} \otimes e^{t_j A}\right)\left(e^{t_j B} \otimes I_{n_A}\right) \mathrm{vec}(-C_1 C_2^T) = \sum_{j=-k}^{k} w_j \left[e^{t_j B} \otimes e^{t_j A}\right] \mathrm{vec}(-C_1 C_2^T).\end{aligned}$$

By using the property (i) in Lemma II, we can write

$$X \approx -\sum_{j=-k}^{k} w_j e^{t_j A} C_1 C_2^T e^{t_j B^T} =: \overline{X}, \tag{2.0.4}$$

where $\overline{X}$ is such that $\mathrm{rank}(\bar{X}) \le (2k+1)s$ and $\|X - \overline{X}\| \le \|C_1\|\|C_2\| K e^{-\pi\sqrt{k}}$, K is a constant that only depends on the spectrum of $I \otimes A + B \otimes I$.

The computation of $\overline{X}$ by (2.0.4) might be not numerically adequate and it is not used in practice. However, this result legitimizes the employment of low-rank methods in the solution of (2.0.1).

Projection methods have been shown to be among the most competitive low-rank methods and in the following Section we recall their general framework for solving (2.0.1). Moreover, in Section 2.2 we propose some computational enhancements of the classical algorithm leading to a reduction in both the computational efforts and the memory requirements of the procedure in case of equations with symmetric coefficient matrices.

To simplify the presentation, from now on we will focus on the case of the Lyapunov matrix equation, that is $B = A^T$ ($n \equiv n_A = n_B$), A stable and $C_1 = C_2$. The solution X will be thus square, symmetric and positive semidefinite [101]. In later Sections we will describe how to naturally treat the general case with A and B distinct and not necessarily with the same dimensions, and different C_1, C_2.

2.1 Projection methods

For the Lyapunov equation

$$AX + XA^T + C_1 C_1^T = 0, \quad A \in \mathbb{R}^{n\times n},\ C_1 \in \mathbb{R}^{n\times s},\ s \ll n, \tag{2.1.5}$$

projection methods compute the numerical solution $\widetilde{X}$ in a sequence of nested vector subspaces, $\mathcal{K}_m \subseteq \mathcal{K}_{m+1} \subseteq \mathbb{R}^n$, $m \ge 1$. The approximation, usually denoted by X_m, is written as the product of matrices

$$X_m = V_m Y_m V_m^T, \tag{2.1.6}$$

where $\mathcal{K}_m = \mathrm{Range}(V_m)$, $V_m = [\mathcal{V}_1, \ldots, \mathcal{V}_m] \in \mathbb{R}^{n\times sm}$, $sm \ll n$, with Y_m symmetric and positive semidefinite. For the sake of simplicity, we assume the matrix V_m to be full rank* and to have orthonormal columns so that $\dim(\mathcal{K}_m) = sm$.

Before presenting the general framework of projection methods, we would like to stress how the quality and effectiveness of the approximation process deeply rely on how much

*This is not easy to guarantee in practice, especially for large values of s. See, e.g., [49, Section 8] for some deflation strategies to overcome the possible linear dependence of the computed basis vectors.

spectral information is captured by $\mathcal{K}_m$, without the space dimension being too large. The block Krylov subspace

$$\mathcal{K}_m = \mathbf{K}_m^{\square}(A, C_1) := \text{Range}\{[C_1, AC_1, \ldots, A^{m-1}C_1]\} = \left\{ \sum_{i=0}^{m-1} A^i C_1 \varsigma_i, \;\; \varsigma_i \in \mathbb{R}^{s\times s} \right\}, \tag{2.1.7}$$

[93] was the first candidate proposed for solving large-scale Lyapunov matrix equations in [94]. Numerical experiments show that $\mathbf{K}_m^{\square}(A, C_1)$ may need to be quite large before a satisfactory approximate solution is obtained [85, 97]. This large number of iterations causes high computational and memory demands. A similar behavior can be observed when the global Krylov subspace

$$\mathbf{K}_m(A, C_1) = \text{span}\,\{C_1, AC_1, \ldots, A^{m-1}C_1\} = \left\{ \sum_{i=0}^{m-1} A^i C_1 \gamma_i, \;\; \gamma_i \in \mathbb{R} \right\},$$

is employed in the solution process [65] since $\mathbf{K}_m(A, C_1)$ can be viewed as a subspace of $\mathbf{K}_m^{\square}(A, C_1)$. Indeed, $\mathbf{K}_m(A, C_1)$ can be obtained from (2.1.7) choosing $\varsigma_i = \gamma_i I_s$.

In the last decade, more sophisticated spaces leading to nonpolynomial approximations to X have been investigated. In particular, recent alternatives include projection onto the extended Krylov subspace [97]

$$\mathbf{EK}_m^{\square}(A, C_1) := \text{Range}\{[C_1, A^{-1}C_1, \ldots, A^{m-1}C_1, A^{-m}C_1]\}, \tag{2.1.8}$$

and the more general rational Krylov subspace

$$\mathbf{K}_m^{\square}(A, C_1, \mathbf{s}) := \text{Range}\{[C_1, (A - s_2 I)^{-1}C_1, \ldots, \prod_{j=2}^{m}(A - s_j I)^{-1}C_1]\}, \tag{2.1.9}$$

where $\mathbf{s} = [s_2, \ldots, s_m]$ are given shifts [35, 36]. Extended and rational Krylov subspaces contain richer spectral information that allows for a significantly lower subspace dimension at the cost of more expensive computations per iteration since s system solves with the coefficients matrix are required at each iteration.

Once $\mathcal{K}_m$ is chosen, the orthogonalization procedure employed in building the basis V_m determines the sparsity pattern of the matrix $T_m := V_m^T A V_m$, that is the projection of A onto $\mathcal{K}_m$. In particular, if $\mathcal{K}_m$ is the (standard) block Krylov subspace $\mathbf{K}_m^{\square}(A, C_1)$, the block Arnoldi process [93, Section 6.12] produces a block upper Hessenberg matrix T_m with blocks of size s,

$$T_m = \begin{pmatrix} \tau_{11} & \tau_{12} & \cdots & \cdots & \tau_{1m} \\ \tau_{21} & \tau_{22} & \tau_{23} & \cdots & \tau_{2m} \\ & \ddots & \ddots & \ddots & \vdots \\ & & \ddots & \ddots & \tau_{m-1,m} \\ & & & \tau_{m,m-1} & \tau_{m,m} \end{pmatrix} \in \mathbb{R}^{sm\times sm}.$$

In Algorithm 2.1 we summarize the $(m+1)$-th step of the block Arnoldi procedure where the new basis block $\mathcal{V}_{m+1} \in \mathbb{R}^{n\times s}$ and the $s \times s$ blocks $\tau_{i,m}$, $i = 1, \ldots, m+1$ are computed. The orthogonalization is performed by the modified block Gram-Schmidt procedure (MGS), see, e.g., [93]; to ensure local orthogonality in finite precision arithmetic, MGS is executed twice. Notice that the new basis block $\mathcal{V}_{m+1}$ consists of s columns, that is s new basis vectors are added to the current space. Therefore $\dim(\mathbf{K}_{m+1}^{\square}(A, C_1)) = \dim(\text{Range}([V_m, \mathcal{V}_{m+1}])) = (m+1)s$.

By construction, the basis blocks $\mathcal{V}_i$'s are orthogonal blocks, that are also orthogonal to each other, and the following (block) Arnoldi relation holds

$$AV_m = V_m T_m + \mathcal{V}_{m+1}\boldsymbol{\tau}_{m+1,m}E_m^T. \tag{2.1.10}$$

See, e.g., [93, Section 6.12].

We now turn our attention to the numerical procedure for computing the approximation (2.1.6). Once a suitable subspace $\mathcal{K}_m$ is chosen, the matrix Y_m can be determined by imposing additional conditions. One such possibility is the orthogonality (Galerkin) condition on the residual $R_m = AX_m + X_mA + C_1C_1^T$. Other different conditions to determine Y_m can be considered. For instance, the minimization of the residual norm or the orthogonality of the residual with respect to some other space have been explored in the literature. See, e.g., [60, 74, 63].

To implement the Galerkin condition, let us write R_m in vector form, that is

$$\mathrm{vec}(R_m) = (A \otimes I + I \otimes A)\mathrm{vec}(X_m) + \mathrm{vec}(C_1C_1^T).$$

We impose $\mathrm{vec}(R_m)$ to be orthogonal to the space spanned by $V_m \otimes V_m$, that is

$$(V_m \otimes V_m)^T \mathrm{vec}(R_m) = 0 \quad \Leftrightarrow \quad V_m^T R_m V_m = 0. \tag{2.1.11}$$

The same result can be obtained by imposing $R_m \perp \mathcal{K}_m$ with respect to the matrix inner product $\langle \cdot, \cdot \rangle_F$. See, e.g., [94].

Substituting R_m into (2.1.11), we have $V_m^T AX_m V_m + V_m^T X_m AV_m + V_m^T C_1 C_1^T V_m = 0$, and with X_m as in (2.1.6), we obtain

$$\left(V_m^T AV_m\right) Y_m V_m^T V_m + V_m^T V_m Y_m \left(V_m^T AV_m\right) + V_m^T C_1 C_1^T V_m = 0. \tag{2.1.12}$$

We assume $\mathrm{Range}(V_1) = \mathrm{Range}(C_1)$, that is $C_1 = V_1\boldsymbol{\gamma}$ for some nonsingular $\boldsymbol{\gamma} \in \mathbb{R}^{s\times s}$. Since V_m has orthonormal columns, $V_m^T C_1 = E_1\boldsymbol{\gamma}$ where $E_1 \in \mathbb{R}^{sm\times s}$ is the first block of s columns of I_{sm}. Equation (2.1.12) can thus be written as

$$T_m Y_m + Y_m T_m + E_1\boldsymbol{\gamma}\boldsymbol{\gamma}^T E_1^T = 0. \tag{2.1.13}$$

The computation of Y_m thus requires the solution of equation (2.1.13) whose well-posedness is not guaranteed a-priori. To overcome this issue, the matrix A is usually assumed to be negative definite so that T_m is stable. Indeed, for any eigenpair $(\lambda, u) \in \mathbb{C}\times\mathbb{C}^{sm}$, $u^*u = 1$, of T_m we get

$$\mathrm{Re}(\lambda) = \mathrm{Re}\left(u^*T_m u\right) = \mathrm{Re}\left(u^*V_m^T AV_m u\right) = \mathrm{Re}\left(z^*Az\right) < 0, \quad z := V_m u.$$

This further condition on A may be viewed as a shortcoming of projection methods since the original problem (2.1.5) requires weaker conditions to be solvable. See Chapter 1. However, both A negative definite and T_m stable are sufficient conditions for the existence of a unique solution to (2.1.5) and (2.1.13) respectively, and projection methods can work in practice without these hypotheses. See, e.g., [74].

As long as m is of moderate size, decomposition-based methods can be employed to solve equation (2.1.13). See, e.g., Algorithm 1.1 and [5, 55, 44].

As shown in the next Proposition, coming from, e.g., [62], the last s columns (or rows) of the solution matrix Y_m are employed to compute the residual norm.

Algorithm 2.1: One step of block Arnoldi method with block MGS.

input : $A \in \mathbb{R}^{n\times n}$, $V_m = [\mathcal{V}_1, \ldots, \mathcal{V}_m] \in \mathbb{R}^{n\times ms}$
output: $\mathcal{V}_{m+1} \in \mathbb{R}^{n\times s}$, $\boldsymbol{\tau}_{i,m}$, $i = 1, \ldots, m+1$

Set $\boldsymbol{\tau}_{i,m} = 0$ for $i = 1, \ldots, m$
Set $\widehat{\mathcal{V}} = A\mathcal{V}_m$
for $\ell = 1, 2$ **do**
  for $i = 1, \ldots, m$ **do**
    Compute $\boldsymbol{\alpha} = \mathcal{V}_i^T \widehat{\mathcal{V}}$
    Set $\boldsymbol{\tau}_{i,m} = \boldsymbol{\tau}_{i,m} + \boldsymbol{\alpha}$
  end
  Compute $\widehat{\mathcal{V}} = \widehat{\mathcal{V}} - \sum_{i=1}^{m} \mathcal{V}_i \boldsymbol{\tau}_{i,m}$
end
Perform economy-size QR of $\widehat{\mathcal{V}}$, $\widehat{\mathcal{V}} = \mathcal{V}_{m+1}\boldsymbol{\tau}_{m+1,m}$

Proposition 2.1.1 ([62]). *Let R_m be the residual matrix $R_m = AX_m + X_mA + C_1C_1^T$, then*

$$\|R_m\|_F = \sqrt{2}\|Y_m E_m \boldsymbol{\tau}_{m+1,m}^T\|_F. \tag{2.1.14}$$

Proof. We have

$$\|R_m\|_F^2 = \|AX_m + X_mA + C_1C_1^T\|_F^2 = \|AV_mY_mV_m^T + V_mY_mV_m^TA + C_1C_1^T\|_F^2.$$

By applying the Arnoldi relation (2.1.10), it holds

$$\begin{aligned}\|R_m\|_F^2 &= \|V_mT_mY_mV_m^T + \mathcal{V}_{m+1}\boldsymbol{\tau}_{m+1,m}E_m^TY_mV_m^T + V_mY_mT_m^TV_m^T + V_mY_mE_m\boldsymbol{\tau}_{m+1,m}^T\mathcal{V}_{m+1}^T + C_1C_1^T\|_F^2 \\ &= \|V_m\underbrace{\left(T_mY_m + Y_mT_m^T + V_m^TC_1C_1^TV_m\right)}_{=0}V_m^T + \mathcal{V}_{m+1}\boldsymbol{\tau}_{m+1,m}E_m^TY_mV_m^T + V_mY_mE_m\boldsymbol{\tau}_{m+1,m}^T\mathcal{V}_{m+1}^T\|_F^2.\end{aligned}$$

Since $\langle \mathcal{V}_{m+1}\boldsymbol{\tau}_{m+1,m}E_m^TY_mV_m^T, V_mY_mE_m\boldsymbol{\tau}_{m+1,m}^T\mathcal{V}_{m+1}^T\rangle_F = 0$, we can write

$$\begin{aligned}\|R_m\|_F^2 &= \|\mathcal{V}_{m+1}\boldsymbol{\tau}_{m+1,m}E_m^TY_mV_m^T\|_F^2 + \|V_mY_mE_m\boldsymbol{\tau}_{m+1,m}^T\mathcal{V}_{m+1}^T\|_F^2 \\ &= \|\boldsymbol{\tau}_{m+1,m}E_m^TY_m\|_F^2 + \|Y_mE_m\boldsymbol{\tau}_{m+1,m}^T\|_F^2 = 2\|Y_mE_m\boldsymbol{\tau}_{m+1,m}^T\|_F^2,\end{aligned}$$

where in the last step the symmetry of Y_m is exploited. □

The matrix Y_m is determined by solving (2.1.13), and it is again symmetric and positive semidefinite. At convergence, the transformation $X_m = V_mY_mV_m^T$ is never explicitly computed or stored. Instead, we factorize Y_m as

$$Y_m = \widehat{Y}\widehat{Y}^T, \quad \widehat{Y} \in \mathbb{R}^{sm\times sm}, \tag{2.1.15}$$

from which a low-rank factor of X_m is obtained as $Z_m = V_m\widehat{Y} \in \mathbb{R}^{n\times sm}$, $X_m = Z_mZ_m^T$. The matrix Y_m may be numerically rank deficient, and this can be exploited to further decrease the rank of Z_m. We write the eigendecomposition of Y_m, $Y_m = W\Sigma W^T$ (with eigenvalues ordered non-increasingly) and discard the negligible eigenvalues, that is $\Sigma = \mathrm{diag}(\Sigma_1, \Sigma_2)$,

$W = [W_1, W_2]$ with $\|\Sigma_2\|_F \le \epsilon$. Therefore, we define again $Y_m \approx \widehat{Y}\widehat{Y}^T$, with $\widehat{Y} = W_1\Sigma_1^{1/2} \in \mathbb{R}^{sm\times t}$, $t \le sm$; in this way, $\|Y_m - \widehat{Y}\widehat{Y}^T\|_F \le \epsilon$. Hence, we set $Z_m = V_m\widehat{Y} \in \mathbb{R}^{n\times t}$. We notice that a significant rank reduction in Y_m is an indication that all relevant information for generating X_m is actually contained in a subspace that is much smaller than $\mathcal{K}_m$. In other words, if $t \ll sm$, the generated subspace is not efficient in capturing the solution information and a different and much smaller space could have been generated to obtain an approximate solution of comparable accuracy. The overall procedure is summarized in Algorithm 2.2.

Algorithm 2.2: Galerkin projection method for the Lyapunov matrix equation.

input : $A \in \mathbb{R}^{n\times n}$, A negative definite, $C_1 \in \mathbb{R}^{n\times s}$, $\epsilon > 0$
output: $Z_m \in \mathbb{R}^{n\times t}$, $t \le sm$

Set $\beta = \|C_1\|_F$
Perform economy-size QR of C_1, $C_1 = V_1\boldsymbol{\gamma}$. Set $\mathcal{V}_1 \equiv V_1$
for $m = 2, 3, \ldots,$ *till convergence,* **do**
  Compute next basis block $\mathcal{V}_m$ and set $V_m = [V_{m-1}, \mathcal{V}_m]$
  Update $T_m = V_m^T A V_m$
  Solve $T_m Y_m + Y_m T_m + E_1\boldsymbol{\gamma}\boldsymbol{\gamma}^T E_1^T = 0, \quad E_1 \in \mathbb{R}^{ms\times s}$
  Compute $\|R_m\|_F = \sqrt{2}\|Y_m E_m \boldsymbol{\tau}_{m+1,m}^T\|_F$
  if $\|R_m\|_F/\beta^2 < \epsilon$ **then**
    Stop
  end
end
Compute the eigendecomposition of Y_m and retain $\widehat{Y} \in \mathbb{R}^{sm\times t}$, $t \le sm$
Set $Z_m = V_m\widehat{Y}$

2.1.1 The case of the Sylvester equation

The strategy presented for the Lyapunov equation (2.1.5) can be extended to the Sylvester equation (2.0.1). In this case, projection methods seek an approximate solution $X_m \in \mathbb{R}^{n_A\times n_B}$ to (2.0.1) of the form $X_m = V_m Y_m U_m^T$ where the orthonormal columns of V_m and U_m span suitable subspaces $\mathcal{K}_m$ and $\mathcal{C}_m$, respectively[†]. The construction of two approximation spaces is thus requested and, for the sake of simplicity, we limit our discussion to the standard Krylov method, that is $\mathcal{K}_m = \mathbf{K}_m^{\square}(A, C_1)$ and $\mathcal{C}_m = \mathbf{K}_m^{\square}(B^T, C_2)$. As in the Lyapunov case, Y_m is computed by imposing a Galerkin condition on the residual matrix $R_m := AX_m + X_mB + C_1C_2^T$, that is

$$V_m^T R_m U_m = 0. \tag{2.1.16}$$

We assume $C_1 = V_1\boldsymbol{\gamma}_1$, $C_2 = U_1\boldsymbol{\gamma}_2$ for some nonsingular $\boldsymbol{\gamma}_1, \boldsymbol{\gamma}_2 \in \mathbb{R}^{s\times s}$, and a similar discussion to the one presented in the previous Section shows that condition (2.1.16) is equivalent to solving the reduced Sylvester problem

$$T_m Y_m + Y_m J_m^T + E_1\boldsymbol{\gamma}_1\boldsymbol{\gamma}_2^T E_1^T = 0, \tag{2.1.17}$$

where $T_m := V_m^T A V_m$, $J_m := U_m^T B^T U_m = (\boldsymbol{\iota}_{ij})$ are both block upper Hessenberg $sm \times sm$ matrices computed via the block Arnoldi procedure. A sufficient condition for the existence

[†] Although the space dimensions of $\mathcal{K}_m$ and $\mathcal{C}_m$ are not necessarily equal, we limit our discussion to the same dimension for simplicity of exposition.

of a unique solution to (2.1.17) in terms of the coefficient matrices A, B is given by the assumption of disjoint $W(A)$ and $W(-B)$. Indeed, equation (2.1.17) admits a unique solution if and only if $\lambda_i - \upsilon_j \neq 0$ for every $\lambda_i \in \Lambda(T_m)$ and $\upsilon_j \in \Lambda(J_m^T)$. If $(\lambda_i, u_i) \in \mathbb{C} \times \mathbb{C}^m$, $(\upsilon_j, h_j) \in \mathbb{C} \times \mathbb{C}^m$, $u_i^* u_i = h_j^* h_j = 1$ are eigenpairs of T_m and J_m^T respectively, we have

$$\begin{aligned}\mathrm{Re}(\lambda_i) - \mathrm{Re}(\upsilon_j) &= \mathrm{Re}(u_i^* T_m u_i) - \mathrm{Re}(h_j^* J_m^T h_j) = \mathrm{Re}(u_i^* V_m^T A V_m u_i) - \mathrm{Re}(h_j^* U_m^T B U_m h_j) \\ &= \mathrm{Re}(z^* A z) - \mathrm{Re}(y^* B y), \quad z := V_m u_i \in \mathbb{C}^{n_A},\ y := U_m h_j \in \mathbb{C}^{n_B},\ z^* z = y^* y = 1,\end{aligned}$$

and this is nonzero as $w^* A w - q^* B q \neq 0$ for every $w \in \mathbb{C}^{n_A}$, $q \in \mathbb{C}^{n_B}$, $w^* w = q^* q = 1$, assuming $W(A) \cap W(-B) = \varnothing$.

The last s rows and columns of Y_m are employed in the residual norm calculation. Indeed, letting $\underline{T}_m = V_{m+1}^T A V_m$ and $\underline{J}_m = U_{m+1}^T B^T U_m$, an argument similar to the one used for proving Proposition 2.1.1 shows that

$$\|R_m\|_F^2 \quad = \quad \|\boldsymbol{\tau}_{m+1,m} E_m^T Y_m\|_F^2 + \|Y_m E_m \boldsymbol{\iota}_{m+1,m}^T\|_F^2, \tag{2.1.18}$$

where $\boldsymbol{\tau}_{m+1,m} = E_{m+1}^T \underline{T}_m E_m \in \mathbb{R}^{s\times s}$ and $\boldsymbol{\iota}_{m+1,m} = E_{m+1}^T \underline{J}_m E_m \in \mathbb{R}^{s\times s}$. See also, e.g., [99, 26].

At convergence, we factorize the matrix $Y_m = \widehat{Y}_1 \widehat{Y}_2^T$. Also in the Sylvester problem Y_m may be numerically singular. In this case, the factors $\widehat{Y}_1, \widehat{Y}_2 \in \mathbb{R}^{sm\times t}$, $t \leq sm$, such that $\|Y_m - \widehat{Y}_1 \widehat{Y}_2^T\|_F \leq \epsilon$ can be computed via the truncated singular value decomposition of the nonsymmetric matrix Y_m. The low-rank factors Z_1, Z_2 of X_m, $X_m \approx Z_1 Z_2^T$, are then computed as $Z_1 = V_m \widehat{Y}_1$ and $Z_2 = U_m \widehat{Y}_2$. Algorithm 2.3 outlines the complete procedure.

Algorithm 2.3: Galerkin projection method for the Sylvester matrix equation.

input : $A, B \in \mathbb{R}^{n\times n}$, $C_1, C_2 \in \mathbb{R}^{n\times s}$, $\epsilon > 0$
output: $Z_1, Z_2 \in \mathbb{R}^{n\times t}$, $t \leq sm$

Set $\beta_1 = \|C_1\|_F$, $\beta_2 = \|C_2\|_F$
Perform economy-size QR of C_1, $C_1 = V_1 \boldsymbol{\gamma}_1$, and C_2, $C_2 = U_1 \boldsymbol{\gamma}_2$. Set $\mathcal{V}_1 \equiv V_1$, $\mathcal{U}_1 \equiv U_1$
for $m = 2, 3, \ldots,$ *till convergence,* **do**
  Compute next basis blocks $\mathcal{V}_m$, $\mathcal{U}_m$ and set $V_m = [V_{m-1}, \mathcal{V}_m]$, $U_m = [U_{m-1}, \mathcal{U}_m]$
  Update $T_m = V_m^T A V_m$, $J_m = U_m^T B U_m$
  Solve $T_m Y_m + Y_m J_m^T + E_1 \boldsymbol{\gamma}_1 \boldsymbol{\gamma}_2^T E_1^T = 0, \quad E_1 \in \mathbb{R}^{ms\times s}$
  Compute $\|R_m\|_F^2 = \|\boldsymbol{\tau}_{m+1,m} E_m^T Y_m\|_F^2 + \|Y_m E_m \boldsymbol{\iota}_{m+1,m}^T\|_F^2$
  if $\|R_m\|_F / (\beta_1 \beta_2) < \epsilon$ **then**
    Stop
  end
end
Compute the singular value decomposition of Y_m and retain $\widehat{Y}_1, \widehat{Y}_2 \in \mathbb{R}^{sm\times t}$, $t \leq sm$
Set $Z_1 = V_m \widehat{Y}_1$, $Z_2 = U_m \widehat{Y}_2$

2.2 Enhanced projection methods for equations with symmetric coefficients

In the following Sections we propose some computational enhancements of Algorithm 2.2 and 2.3 in case of equations with symmetric coefficient matrices.

We first focus on the Lyapunov equation and, as the approximation space $\mathcal{K}_m$ expands, the principal costs of Algorithm 2.2 are steps 3 and 5. For A symmetric and $\mathcal{K}_m = \mathbf{K}^{\square}_m(A, C_1)$, the computational cost of the basis construction in line 3 can be reduced. The matrix $T_m = V_m^T A V_m$ stemming from the block Arnoldi procedure must be block upper Hessenberg and symmetric, so that it is block tridiagonal

$$T_m = \begin{pmatrix} \boldsymbol{\tau}_{11} & \boldsymbol{\tau}_{12} & & & \\ \boldsymbol{\tau}_{21} & \boldsymbol{\tau}_{22} & \boldsymbol{\tau}_{23} & & \\ & \ddots & \ddots & \ddots & \\ & \ddots & \ddots & \boldsymbol{\tau}_{m-1,m} & \\ & & \boldsymbol{\tau}_{m,m-1} & \boldsymbol{\tau}_{m,m} \end{pmatrix} \in \mathbb{R}^{sm \times sm},$$

with blocks of size s.

Since $\boldsymbol{\tau}_{i,m} = 0$, $i = 1, \dots, m-2$, from the Arnoldi relation (2.1.10) we have

$$A\mathcal{V}_m = \mathcal{V}_m \boldsymbol{\tau}_{m,m} + \mathcal{V}_{m-1} \boldsymbol{\tau}_{m-1,m} + \mathcal{V}_{m+1} \boldsymbol{\tau}_{m+1,m},$$

and therefore, at least in exact arithmetic, it holds

$$\mathcal{V}_{m+1} \boldsymbol{\tau}_{m+1,m} = A\mathcal{V}_m - \mathcal{V}_m \boldsymbol{\tau}_{m,m} - \mathcal{V}_{m-1} \boldsymbol{\tau}_{m-1,m}. \tag{2.2.19}$$

The procedure that takes advantage of (2.2.19) is called block Lanczos method and it can be employed in line 3 of Algorithm 2.2. Algorithm 2.4 describes this process at iteration $m+1$, with $W = A\mathcal{V}_m$ if the block Krylov space $\mathbf{K}^{\square}_m(A, C_1)$ is chosen (beside each command is the leading computational cost of the operation). We recall once again that we assume throughout the full rank of the generated basis. Deflation could be implemented as it is customary in block methods whenever rank deficiency is detected.

Algorithm 2.4: One step of block Lanczos with block MGS.

input : m, W, $\mathcal{V}_{m-1}, \mathcal{V}_m \in \mathbb{R}^{n \times s}$
output: $\mathcal{V}_{m+1} \in \mathbb{R}^{n \times s}$, $\boldsymbol{\tau}_{m-1,m}, \boldsymbol{\tau}_{m,m}, \boldsymbol{\tau}_{m+1,m} \in \mathbb{R}^{s \times s}$

1 Set $\boldsymbol{\tau}_{m-1,m} = \boldsymbol{\tau}_{m,m} = \mathbf{0}$
for $l = 1, 2$ **do**
  for $i = m-1, m$ **do**
2 Compute $\boldsymbol{\alpha} = \mathcal{V}_i^T W \leftarrow (2n-1)s^2$ flops
3 Set $\boldsymbol{\tau}_{i,m} = \boldsymbol{\tau}_{i,m} + \boldsymbol{\alpha} \leftarrow s^2$ flops
4 Compute $W = W - \mathcal{V}_i \boldsymbol{\alpha} \leftarrow 2s^2 n$ flops
  end
end
5 Perform economy-size QR of W, $W = \mathcal{V}_{m+1} \boldsymbol{\tau}_{m+1,m} \leftarrow 3ns^2$ flops

We emphasize that only the last $3s$ terms of the basis must be stored, and the computational cost of Algorithm 2.4 is fixed with respect to m. In particular, at each iteration m, Algorithm 2.4 costs $\mathcal{O}\left((19n + s)s^2\right)$ flops.

To check convergence, projection methods require to compute the matrix Y_m by solving the reduced problem (2.1.13) whose complexity depends on the approximation space dimension. The solution of the reduced problem is meant to account for a low percentage of the overall computational cost. Unfortunately, this cost grows nonlinearly with the space dimension, therefore solving the reduced problem may become very expensive if a large approximation space is needed. In particular, the computation of the whole matrix Y_m requires

full matrix-matrix operations and a Schur decomposition of the coefficient matrix T_m, whose costs are $\mathcal{O}\big((sm)^3\big)$ flops. See Section 1.3. Clearly, step 5 becomes comparable with step 3 in cost for $sm \gg 1$, for instance if convergence is slow, so that $m \gg 1$.

In case of Lyapunov equations with symmetric coefficient matrix A, we devise a strategy that significantly reduces the computational cost of evaluating the residual norm for both $\mathbf{K}_m^{\square}(A, C_1)$ and the extended Krylov subspace $\mathbf{EK}_m^{\square}(A, C_1)$.

Moreover, Step 10 of Algorithm 2.2 shows that at convergence, the whole basis must be saved to return the factor Z_m. This represents a major shortcoming when convergence is slow, since V_m may require large memory allocations. In case of $\mathbf{K}_m^{\square}(A, C_1)$, a "two-pass" strategy is implemented to avoid storing the whole basis V_m; see [69] for earlier use of this device in the same setting, and, e.g., [42] in the matrix function context.

The procedures are then generalized to address equation (2.0.1) with symmetric A, B.

The results of the following Sections are taken from [83].

2.2.1 Computing the residual norm without the whole Y_m for $\mathbf{K}_m^{\square}(A, C_1)$

The solution of the projected problem (2.1.13) requires the Schur decomposition of T_m. For real symmetric matrices, the Schur decomposition amounts to the eigendecomposition $T_m = Q_m \Lambda_m Q_m^T$, $\Lambda_m = \operatorname{diag}(\lambda_1, \dots, \lambda_{sm})$, and the symmetric block tridiagonal structure of T_m can be exploited so as to use only $\mathcal{O}((sm)^2)$ flops; see Section 2.2.2 for further details. Equation (2.1.13) can thus be written as

$$\Lambda_m \widetilde{Y} + \widetilde{Y} \Lambda_m + Q_m^T E_1 \boldsymbol{\gamma}\boldsymbol{\gamma}^T E_1^T Q_m = 0, \qquad \text{where} \quad \widetilde{Y} := Q_m^T Y_m Q_m. \tag{2.2.20}$$

Since Λ_m is diagonal, the entries of $\widetilde{Y}$ can be computed by substitution [99, Section 4], so that

$$Y_m = Q_m \widetilde{Y} Q_m^T = -Q_m \left(\frac{e_i^T Q_m^T E_1 \boldsymbol{\gamma}\boldsymbol{\gamma}^T E_1^T Q_m e_j}{\lambda_i + \lambda_j} \right)_{ij} Q_m^T. \tag{2.2.21}$$

It turns out that only the quantities within parentheses in (2.2.21) are needed for the residual norm computation, thus avoiding the $\mathcal{O}\big((sm)^3\big)$ cost of recovering Y_m.

Proposition 2.2.1. *Let $T_m = Q_m \Lambda_m Q_m^T$ denote the eigendecomposition of T_m. Then*

$$\|R_m\|_F^2 \;=\; 2\left(\|e_1^T S_m D_1^{-1} W_m\|_2^2 + \dots + \|e_{sm}^T S_m D_{sm}^{-1} W_m\|_2^2 \right), \tag{2.2.22}$$

where $S_m = Q_m^T E_1 \boldsymbol{\gamma}\boldsymbol{\gamma}^T E_1^T Q_m \in \mathbb{R}^{sm \times sm}$, $W_m = Q_m^T E_m \boldsymbol{\tau}_{m+1,m}^T \in \mathbb{R}^{sm \times s}$ and $D_j = \lambda_j I_{sm} + \Lambda_m$ for all $j = 1, \dots, sm$.

Proof. Exploiting (2.1.14) and the representation formula (2.2.21) we have

$$\begin{aligned} \|R_m\|_F^2 \;&=\; 2\|Y_m E_m \boldsymbol{\tau}_{m+1,m}^T\|_F^2 = 2 \left\| \left(\frac{e_i^T Q_m^T E_1 \boldsymbol{\gamma}\boldsymbol{\gamma}^T E_1^T Q_m e_j}{\lambda_i + \lambda_j} \right)_{ij} Q_m^T E_m \boldsymbol{\tau}_{m+1,m}^T \right\|_F^2 \\ &=\; 2 \sum_{k=1}^{s} \left\| \left(\frac{e_i^T S_m e_j}{\lambda_i + \lambda_j} \right)_{ij} W_m e_k \right\|_2^2 . \end{aligned} \tag{2.2.23}$$

For all $k = 1, \dots, s$, we can write

$$\begin{aligned} \left\| \left(\frac{e_i^T S_m e_j}{\lambda_i + \lambda_j} \right)_{ij} W_m e_k \right\|_2^2 \;&=\; \left(\sum_{j=1}^{sm} \frac{e_1^T S_m e_j}{\lambda_1 + \lambda_j} e_j^T W_m e_k \right)^2 + \dots + \left(\sum_{j=1}^{sm} \frac{e_{sm}^T S_m e_j}{\lambda_{sm} + \lambda_j} e_j^T W_m e_k \right)^2 \\ &=\; \left(e_1^T S_m D_1^{-1} W_m e_k \right)^2 + \dots + \left(e_{sm}^T S_m D_{sm}^{-1} W_m e_k \right)^2 . \end{aligned} \tag{2.2.24}$$

Plugging (2.2.24) into (2.2.23) we have

$$\begin{aligned}\|R_m\|_F^2 &= 2\sum_{k=1}^{s}\sum_{i=1}^{sm}\left(e_i^T S_m D_i^{-1} W_m e_k\right)^2 = 2\sum_{i=1}^{sm}\sum_{k=1}^{s}\left(e_i^T S_m D_i^{-1} W_m e_k\right)^2 \\ &= 2\sum_{i=1}^{sm}\left\|e_i^T S_m D_i^{-1} W_m\right\|_2^2.\end{aligned}$$

□

Algorithm 2.5: `cTri`.

input : $T_m \in \mathbb{R}^{\ell m\times \ell m}$, $\boldsymbol{\gamma}, \boldsymbol{\tau}_{m+1,m} \in \mathbb{R}^{\ell\times\ell}$ (ℓ is the block size)
output: res ($= \|R\|_F$)

1 Tridiagonalize $P_m^T T_m P_m = F_m$
2 Compute $F_m = G_m \Lambda_m G_m^T$
3 Compute $E_1^T Q_m = \left(E_1^T P_m\right) G_m$, $E_m^T Q_m = \left(E_m^T P_m\right) G_m$
4 Compute $S_m = \left(Q_m^T E_1 \boldsymbol{\gamma}\right)\left(\boldsymbol{\gamma}^T E_1^T Q_m\right) \leftarrow (2\ell-1)\ell^2 m + (2\ell-1)\ell^2 m^2$ *flops*
5 Compute $W_m = \left(Q_m^T E_m\right)\boldsymbol{\tau}_{m+1,m}^T \leftarrow (2\ell-1)\ell^2 m$ *flops*
6 Set $res = 0$
for $i = 1, \ldots, \ell m$ **do**
7 Set $D_i = \lambda_i I_{\ell m} + \Lambda_m$
8 $res = res + \left\|\left(e_i^T S_m\right) D_i^{-1} W_m\right\|_2^2 \leftarrow 2\ell^2 m + \ell m + \ell$ *flops*
end
9 Set $res = \sqrt{2}\sqrt{res}$

2.2.2 The algorithm for the residual norm computation

Algorithm 2.5 summarizes the procedure that takes advantage of Proposition 2.2.1. Computing the residual norm by (2.2.23) has a leading cost of $4s^3m^2$ flops for standard Krylov (with $\ell = s$). This should be compared with the original procedure in steps 5 and 6 of Algorithm 2.2, whose cost is $\mathcal{O}\left(s^3m^3\right)$ flops, with a large constant. Proposition 2.2.1 also shows that only the first and last ℓ components of the eigenvectors of T_m are necessary in the residual norm evaluation and the computation of the complete eigendecomposition $T_m = Q_m \Lambda_m Q_m^T$ may be avoided. To this end, the matrix T_m can be tridiagonalized, $P_m^T T_m P_m = F_m$, explicitly computing only the first and last ℓ rows of the transformation matrix P_m, namely $E_1^T P_m$ and $E_m^T P_m$. The eigendecomposition $F_m = G_m \Lambda_m G_m^T$ is computed exploiting the tridiagonal structure of F_m. The first and last ℓ rows of the eigenvectors matrix Q_m are then computed by $E_1^T Q_m = \left(E_1^T P_m\right) G_m$ and $E_m^T Q_m = \left(E_m^T P_m\right) G_m$, avoiding the expensive matrix-matrix product $Q_m = P_m G_m$.

Once the stopping criterion in step 7 of Algorithm 2.2 is satisfied, the factor Z_m can be finally computed. Once again, this can be performed without explicitly computing Y_m, which requires the expensive computation $Y_m = Q_m \widetilde{Y} Q_m^T$. Indeed, the truncation strategy discussed around (2.1.15) can be applied to $\widetilde{Y}$ by computing the matrix $\breve{Y} \in \mathbb{R}^{sm\times t}$, $t \le sm$ so that $\widetilde{Y} \approx \breve{Y}\breve{Y}^T$. This factorization further reduces the overall computational cost, since

only $(2ms-1)tms$ flops are required to compute $Q_m\widetilde{Y}$, with no loss of information at the prescribed accuracy. The solution factor Z_m is then computed as $Z_m = V_m\left(Q_m\widetilde{Y}\right)$.

To make fair comparisons with state-of-the-art algorithms that employ LAPACK and SLICOT subroutines (see Section 2.2.6 for more details), we used a C-compiled mex-code `cTri` to implement Algorithm 2.5, making use of LAPACK and BLAS subroutines. In particular, the eigendecomposition $T_m = Q_m\Lambda_m Q_m^T$ is performed as follows. The block tridiagonal matrix T_m is tridiagonalized, $P_m^T T_m P_m = F_m$, by the LAPACK subroutine `dsbtrd` that exploits its banded structure. The transformation matrix P_m is represented as a product of elementary reflectors and only its first and last ℓ rows, $E_1^T P_m$, $E_m^T P_m$, are actually computed. The LAPACK subroutine `dstevr` is employed to compute the eigendecomposition of the tridiagonal matrix F_m. This routine applies Dhillon's MRRR method [33] whose main advantage is the computation of numerically orthogonal eigenvectors without an explicit orthogonalization procedure. This feature limits to $\mathcal{O}((\ell m)^2)$ flops the computation of $F_m = G_m\Lambda_m G_m^T \in \mathbb{R}^{\ell m\times \ell m}$; see [33, 34] for more details.

2.2.3 A "two-pass" strategy to avoid the storage of the whole basis

While the block Lanczos method requires the storage of only $3s$ basis vectors, the whole $V_m = [\mathcal{V}_1,\dots,\mathcal{V}_m] \in \mathbb{R}^{n\times sm}$ is needed to compute the low-rank factor Z_m at convergence (step 10 of Algorithm 2.2). Since

$$Z_m = V_m(Q_m\widetilde{Y}) = \sum_{i=1}^{m} \mathcal{V}_i E_i^T (Q_m\widetilde{Y}), \tag{2.2.25}$$

we suggest not to store V_m during the iterative process but to perform, at convergence, a second Lanczos pass computing and adding the rank-s term in (2.2.25) at the i-th step, in an incremental fashion. We point out that the orthonormalization coefficients are already available in the matrix T_m, therefore $\mathcal{V}_i$ is simply computed by repeating the three-term recurrence (2.2.19), which costs $\mathcal{O}\left((4n+1)s^2\right)$ flops plus the multiplication by A, making the second Lanczos pass cheaper than the first one.

It is well known that the short-term recurrence of the block Lanczos method may lead to a loss of orthogonality of the columns of V_m which in turn affects the computation of the matrix T_m. We assume this phenomenon to be very mild so that the calculation of T_m does not dramatically suffer from the lack of an explicit reorthogonalization. If this is not the case, a full orthogonalization procedure should be preferred and the two-pass strategy does not provide any gain as the complete basis has to be stored.

2.2.4 Enhanced extended Krylov subspace method

Rational Krylov subspaces have shown to provide dramatic performance improvements over classical polynomial Krylov subspaces, because they build spectral information earlier, thus generating a much smaller space dimension to reach the desired accuracy. The price to pay is that each iteration is more computationally involved, as it requires solves with the coefficient matrices. The overall CPU time performance thus depends on the data sparsity of the given problem; we refer the reader to [99] for a thorough discussion.

In this Section we show that the enhanced procedure for the residual norm computation can be applied to a particular rational Krylov based strategy, the extended Krylov subspace method, since also this algorithm relies on a block tridiagonal reduced matrix when data are symmetric. Different strategies for building the basis $V_m = [\mathcal{V}_1,\dots,\mathcal{V}_m] \in \mathbb{R}^{n\times 2sm}$ of the extended Krylov subspace $\mathbf{EK}_m^{\square}(A, C_1)$ can be found in the literature, see, e.g., [61, 80, 97].

An intuitive key fact is that the subspace expands in the directions of A and A^{-1} at the same time. In the block case, a natural implementation thus generates two new blocks of vectors at the time, one in each of the two directions. Starting with $[V_1, A^{-1}V_1]$, the next iterations generate the blocks $\mathcal{V}_m^{(1)}, \mathcal{V}_m^{(2)} \in \mathbb{R}^{n\times s}$ by multiplication by A and solve with A, respectively, and then setting $\mathcal{V}_m = [\mathcal{V}_m^{(1)}, \mathcal{V}_m^{(2)}] \in \mathbb{R}^{n\times 2s}$. As a consequence, the block Lanczos procedure described in Algorithm 2.4 can be employed with $W = [A\mathcal{V}_m^{(1)}, A^{-1}\mathcal{V}_m^{(2)}]$ (with $2s$ columns). The orthogonalization process determines the coefficients of the symmetric block tridiagonal matrix H_m with blocks of size $2s$,

$$H_m = \begin{pmatrix} \boldsymbol{\vartheta}_{11} & \boldsymbol{\vartheta}_{12} & & & \\ \boldsymbol{\vartheta}_{21} & \boldsymbol{\vartheta}_{22} & \boldsymbol{\vartheta}_{23} & & \\ & \ddots & \ddots & \ddots & \\ & \ddots & \ddots & \boldsymbol{\vartheta}_{m-1,m} & \\ & & \boldsymbol{\vartheta}_{m,m-1} & \boldsymbol{\vartheta}_{m,m} \end{pmatrix} \in \mathbb{R}^{2sm\times 2sm},$$

such that $\mathcal{V}_{m+1}\boldsymbol{\vartheta}_{m+1,m} = [A\mathcal{V}_m^{(1)}, A^{-1}\mathcal{V}_m^{(2)}] - \mathcal{V}_m\boldsymbol{\vartheta}_{m,m} - \mathcal{V}_{m-1}\boldsymbol{\vartheta}_{m-1,m}$. The coefficients $\boldsymbol{\vartheta}$'s correspond to the $\boldsymbol{\tau}$'s in Algorithm 2.4, however as opposed to the standard Lanczos procedure, $H_m \neq T_m = V_m^T A V_m$. Nonetheless, a recurrence can be derived to compute the columns of T_m from those of H_m during the iterations; see [97, Proposition 3.2]. The computed T_m is block tridiagonal, with blocks of size $2s$, and this structure allows us to use the same approach followed for the block standard Krylov method as relation (2.1.14) still holds. Algorithm 2.5 can thus be adopted to compute the residual norm also in the extended Krylov approach with $\ell = 2s$. Moreover, it is shown in [97] that the off-diagonal blocks of T_m have a zero lower $s \times 2s$ block, that is

$$\boldsymbol{\tau}_{i,i-1} = \begin{bmatrix} \overline{\boldsymbol{\tau}}_{i,i-1} \\ 0 \end{bmatrix}, \quad \overline{\boldsymbol{\tau}}_{i,i-1} \in \mathbb{R}^{s\times 2s},\ i = 1, \dots, m.$$

This observation can be exploited in the computation of the residual norm. Indeed, the same argument of Proposition 2.1.1 leads to

$$\|R_m\|_F = \sqrt{2}\|Y_m E_m \overline{\boldsymbol{\tau}}_{m+1,m}^T\|_F,$$

and $\overline{\boldsymbol{\tau}}_{m+1,m}$ can be passed as an input argument to `cTri` instead of the whole $\boldsymbol{\tau}_{m+1,m}$.

The extended Krylov subspace dimension grows faster than the standard one as it is augmented by $2s$ vectors per iteration. In general, this does not create severe storage difficulties as the extended Krylov approach exhibits faster convergence than standard Krylov in terms of number of iterations. However, for hard problems the space may still become too large to be stored, especially for large s. In this case, a two-pass-like strategy may be appealing. To avoid the occurrence of sm new system solves with A, however, it may be wise to still store the second blocks, $\mathcal{V}_i^{(2)}$, $i = 1, \dots, m$, and only save half memory allocations, those corresponding to the matrices $\mathcal{V}_i^{(1)}$, $i = 1, \dots, m$.

Finally, we remark that if we were to use more general rational Krylov subspaces (2.1.9), which use rational functions other than A and A^{-1} to generate the space [99], the projected matrix T_m would lose the convenient block tridiagonal structure, so that the new strategy would not be applicable.

2.2.5 Generalization to Sylvester equations

The strategy presented for Lyapunov equations with symmetric data can be extended to the Sylvester equation (2.0.1) where the coefficient matrices A, B are both large symmetric and

negative definite. Similarly to the Lyapunov case, the projected problem (2.1.17) in line 5 of Algorithm 2.3 has to be solved. Computing the eigendecompositions $T_m = Q_m \Lambda_m Q_m^T$, $\Lambda_m = \operatorname{diag}(\lambda_1, \ldots, \lambda_{sm})$, and $J_m = P_m \Upsilon_m P_m^T$, $\Upsilon_m = \operatorname{diag}(\upsilon_1, \ldots, \upsilon_{sm})$, the solution Y_m to (2.1.17) can be written as

$$Y_m = Q_m \widetilde{Y} P_m^T = -Q_m \left(\frac{e_i^T Q_m^T E_1 \boldsymbol{\gamma}_1 \boldsymbol{\gamma}_2^T E_1^T P_m e_j}{\upsilon_i + \lambda_j} \right)_{ij} P_m^T, \tag{2.2.26}$$

and its last s rows and columns are employed in the residual norm calculation (2.1.18).

The same arguments of Section 2.2.1 can be applied to the factors in (2.1.18) leading to Algorithm 4 for the computation of the residual norm without explicitly assembling the matrix Y_m. The eigendecompositions in step 1 are not fully computed. In particular, only the spectrum and the first and last ℓ components of the eigenvectors of T_m and J_m are explicitly computed following the strategy presented in Section 2.2.2.

Algorithm 2.6: Computing the residual norm for A and B large.

input : T_m, $J_m \in \mathbb{R}^{\ell m \times \ell m}$, $\boldsymbol{\gamma}_1, \boldsymbol{\gamma}_2$, $\boldsymbol{\tau}_{m+1,m}$, $\boldsymbol{\iota}_{m+1,m} \in \mathbb{R}^{\ell \times \ell}$
output: res $(= \|R\|_F)$

1 Compute $T_m = Q_m \Lambda_m Q_m^T$ and $J_m = P_m \Upsilon_m P_m^T$
2 Compute $S_m := \left(Q_m^T E_1 \boldsymbol{\gamma}_1\right)\left(\boldsymbol{\gamma}_2^T E_1^T P_m\right)$
3 Compute $F_m := \left(Q_m^T E_m\right)\boldsymbol{\tau}_{m+1,m}^T$, $G_m := \left(P_m^T E_m\right)\boldsymbol{\iota}_{m+1,m}^T$
4 Set $res = 0$
for $i = 1, \ldots, \ell m$ **do**
5 Set $D_i' := \upsilon_i I_{\ell m} + \Lambda_m$ and $D_i'' := \lambda_i I_{\ell m} + \Upsilon_m$
6 $res = res + \left\| e_i^T S_m D_i'^{-1} G_m \right\|_2^2 + \left\| e_i^T S_m^T D_i''^{-1} F_m \right\|_2^2$
end
7 Set $res = \sqrt{res}$

If equation (2.0.1) is solved by the standard Krylov method, the two-pass strategy presented in Section 2.2.3 can be easily adapted to the Sylvester case. Indeed, denoting the basis matrices of $\mathbf{K}_m^{\square}(A, C_1)$ and $\mathbf{K}_m^{\square}(B, C_2)$ by $V_m = [\mathcal{V}_1, \ldots, \mathcal{V}_m]$ and $U_m = [\mathcal{U}_1, \ldots, \mathcal{U}_m]$ respectively, the low-rank factors Z_1 and Z_2 in line 10 of Algorithm 2.3 can be written as

$$Z_1 = V_m \left(Q_m \widehat{Y}_1\right) = \sum_{i=1}^{m} \mathcal{V}_i E_i^T \left(Q_m \widehat{Y}_1\right), \qquad Z_2 = U_m \left(P_m \widehat{Y}_2\right) = \sum_{i=1}^{m} \mathcal{U}_i E_i^T \left(P_m \widehat{Y}_2\right),$$

where $\widehat{Y}_1$, $\widehat{Y}_2 \in \mathbb{R}^{sm \times t}$, $t \leq sm$, are low-rank factors of $\widetilde{Y}$ in (2.2.26) that is $\|\widetilde{Y} - \widehat{Y}_1 \widehat{Y}_2^T\|_F \leq \epsilon$ for a given ϵ.

As in the Lyapunov case, the factors Z_1, Z_2 can be computed in a second Lanczos pass since the terms $\mathcal{V}_i E_i^T \left(Q_m \widehat{Y}_1\right)$ and $\mathcal{U}_i E_i^T \left(P_m \widehat{Y}_2\right)$ do not require the whole basis to be available. Therefore, for the Sylvester problem (2.0.1), the "two-pass" strategy allows us to store only $6s$ basis vectors, $3s$ vectors for each of the two bases.

In some applications, such as the solution of eigenvalues problems [116] or boundary value problems with separable coefficients [113], the matrices A and B in (2.0.1) could have very different dimensions. In particular, one of them, for instance, B, could be of moderate size, that is $n_B \ll 1000$. In this case, the projection method presented in Section 2.1.1 can be simplified. Indeed, a reduction of the matrix B becomes unnecessary, so that a

numerical solution X_m to (2.0.1) of the form $X_m = V_m Y_m$ is sought, where the columns of V_m span $\mathcal{K}_m = \mathbf{K}_m^{\square}(A, C_1)$, as before. The Galerkin condition on the residual matrix $R_m := AX_m + X_m B + C_1 C_2^T$ thus becomes

$$V_m^T R_m = 0, \tag{2.2.27}$$

see [99, Section 4.3] for more details. The procedure continues as in the previous cases, taking into account that the original problem is only reduced "from the left". Assuming $C_1 = V_1 \boldsymbol{\gamma}_1$, we obtain

$$0 = V_m^T A X_m + V_m^T X_m B + V_m^T C_1 C_2^T = \left(V_m^T A V_m\right) Y_m + \left(V_m^T V_m\right) Y_m B + E_1 \boldsymbol{\gamma}_1 C_2^T,$$

that is

$$T_m Y_m + Y_m B + E_1 \boldsymbol{\gamma}_1 C_2^T = 0. \tag{2.2.28}$$

Computing the eigendecompositions $T_m = Q_m \Lambda_m Q_m^T$, $\Lambda_m = \operatorname{diag}(\lambda_1, \ldots, \lambda_{sm})$ and $B = P \Upsilon P^T$, $\Upsilon = \operatorname{diag}(\upsilon_1, \ldots, \upsilon_{n_B})$, the solution matrix Y_m to (2.2.28) can be written as

$$Y_m = Q_m \widetilde{Y} P^T = -Q_m \left(\frac{Q_m^T E_1 \boldsymbol{\gamma}_1 C_2^T P}{\lambda_i + \upsilon_j} \right)_{ij} P^T. \tag{2.2.29}$$

As before, the block tridiagonal structure of T_m can be exploited in the eigendecomposition computation $T_m = Q_m \Lambda_m Q_m^T$, while the eigendecomposition $B = P \Upsilon P^T$ is computed once for all at the beginning of the whole process.

The expression of the residual norm simplifies as $\|R_m\|_F = \|Y_m^T E_m^T \boldsymbol{\tau}_{m+1,m}^T\|_F$. To compute this norm without assembling the whole matrix Y_m, a slight modification of Algorithm 2.5 can be implemented. The resulting procedure is summarized in Algorithm 2.7 where only selected entries of the eigenvector matrix Q_m in step 1 are computed; see the corresponding strategy in Section 2.2.2.

Algorithm 2.7: Computing the residual norm for A large and B small.

input : $T_m \in \mathbb{R}^{\ell m \times \ell m}$, $\boldsymbol{\tau}_{m+1,m} \in \mathbb{R}^{\ell \times \ell}$, $P^T C_2 \boldsymbol{\gamma}_1^T \in \mathbb{R}^{n_2 \times \ell}$, $\{\upsilon_i\}_{i=1,\ldots,n_B}$
output: res $(= \|R\|_F)$

1 Compute $T_m = Q_m \Lambda_m Q_m^T$
2 Compute $S_m = \left(P^T C_2 \boldsymbol{\gamma}_1^T\right)\left(E_1^T Q_m\right)$
3 Compute $W_m = \left(Q_m^T E_m\right) \boldsymbol{\tau}_{m+1,m}^T$
4 Set $res = 0$
for $i = 1, \ldots, n_2$ **do**
5 Set $D_i = \upsilon_i I_{\ell m} + \Lambda_m$
6 $res = res + \left\| \left(e_i^T S_m\right) D_i^{-1} W_m \right\|_2^2$
end
7 Set $res = \sqrt{res}$

A reduced rank approximation to the solution Y_m obtained by (2.2.29) is given as $\widetilde{Y} \approx \widehat{Y}_1 \widehat{Y}_2^T$, so that the low rank factors Z_1, Z_2 are computed as $Z_1 = V_m \left(Q_m \widehat{Y}_1\right)$ and $Z_2 = P \widehat{Y}_2$. Again, a two-pass strategy can be employed to avoid storing the whole matrix V_m.

2.2.6 Numerical examples

In this Section some numerical examples illustrating the enhanced algorithms are reported. All results were obtained with Matlab R2015a on a Dell machine with two 2GHz processors and 128 GB of RAM.

We compare the standard implementation of projection methods – Algorithm 2.2 and 2.3 for Lyapunov and Sylvester equations respectively – and the proposed enhancements, where lines 5 and 6 of Algorithm 2.2–2.3 are replaced by Algorithm 2.5, 2.6 or 2.7 depending on the considered problem. For the standard implementation, different decomposition based solvers for line 5 in Algorithm 2.2–2.3 are employed: The Bartels-Stewart algorithm (function `lyap`), one of its variants (`lyap2`)[‡], and the Hammarling method (`lyapchol`). All these algorithms make use of SLICOT or LAPACK subroutines.

Examples with a sample of small values of the rank s of $C_1C_2^T$ are reported. In all our experiments the convergence tolerance on the relative residual norm is $\texttt{tol} = 10^{-6}$.

Example 2.2.1. In the first example, the block standard Krylov approach is tested for solving the Lyapunov equation $AX + XA + C_1C_1^T = 0$. We consider $A \in \mathbb{R}^{n\times n}$, $n = 21904$ stemming from the discretization by centered finite differences of the differential operator

$$\mathcal{L}(u) = (e^{-xy}u_x)_x + (e^{xy}u_y)_y,$$

on the unit square with zero Dirichlet boundary conditions, while $C_1 = \texttt{rand}\,(n, s)$, $s = 1, 4, 8$, that is the entries of C_1 are random numbers uniformly distributed in the interval $(0, 1)$. C_1 is then normalized, $C_1 = C_1/\|C_1\|_F$. Table 2.1 (left) reports the CPU time (in seconds) needed for evaluating the residual norm (time res) and for completing the whole procedure (time tot). Convergence is checked at each iteration. For instance, for $s = 1$, using `lyapchol` as inner solver the solution process takes 38.51 secs, 36.51 of which are used for solving the inner problem of step 5. If we instead use `cTri`, the factors of X_m are determined in 7.25 seconds, only 4.42 of which are devoted to evaluating the residual norm. Therefore, 87.9% of the residual computation CPU time is saved, leading to a 81.2% saving for the whole procedure. An explored device to mitigate the residual norm computational cost is to check the residual only periodically. In the right-hand side of Table 2.1 we report the results in case the residual norm is computed every 10 iterations.

Table 2.2 shows that the two-pass strategy of Section 2.2.3 drastically reduces the memory requirements of the solution process, as already observed in [69], at a negligible percentage of the total execution time.

Example 2.2.2. The RAIL benchmark problem[§] solves the generalized Lyapunov equation

$$AXE + EXA + C_1C_1^T = 0, \tag{2.2.30}$$

where $A, E \in \mathbb{R}^{n\times n}$, $n = 79841$, $C_1 \in \mathbb{R}^{n\times s}$, $s = 7$. Following the discussion in [97], equation (2.2.30) can be treated as a standard Lyapunov equation for E symmetric and positive definite. This is a recognized hard problem for the standard Krylov subspace, therefore the extended Krylov subspace method is applied, and convergence is checked at each iteration. Table 2.3 collects the results. In spite of the 52 iterations needed to converge, the space dimension is large, indeed $\dim\left(\mathbf{EK}_m^{\square}(A, C_1)\right) = 728$ and the memory-saving strategy of Section 2.2.4 may be attractive; it was not used for this specific example, but it can be

[‡]The function `lyap2` was slightly modified to exploit the orthogonality of the eigenvectors matrix.

[§]`http://www.simulation.uni-freiburg.de/downloads/benchmark/Steel%20Profiles%20%2838881%29`

	time res (secs)	gain	time tot (secs)	gain	time res (secs)	gain	time tot (secs)	gain
	$s = 1$ (444 its)							
`lyap`	42.36	89.5%	45.18	83.9%	4.78	89.7%	7.87	52.9%
`lyapchol`	36.51	87.9%	38.51	81.2%	4.27	88.5%	7.59	51.25%
`lyap2`	34.27	87.1%	37.07	80.4%	3.85	87.2%	7.14	48.1%
`cTri`	**4.42**	⤾	**7.25**	⤾	**0.49**	⤾	**3.70**	⤾
	$s = 4$ (319 its)							
`lyap`	819.02	96.4%	825.44	95.6%	88.52	96.6%	95.60	91.65%
`lyapchol`	213.87	86.1%	220.51	83.6%	21.38	86.1%	26.83	70.2%
`lyap2`	212.99	86.0%	219.34	83.5%	20.28	85.3%	27.65	71.1%
`cTri`	**29.78**	⤾	**36.21**	⤾	**2.97**	⤾	**7.98**	⤾
	$s = 8$ (250 its)							
`lyap`	2823.31	97.9%	2836.29	97.6%	305.11	98.2%	313.49	95.8%
`lyapchol`	415.42	85.7%	427.21	84.1%	38.94	85.7%	46.96	71.8%
`lyap2`	424.23	86.0%	435.90	84.4%	41.39	86.5%	49.15	73.1%
`cTri`	**59.25**	⤾	**67.89**	⤾	**5.56**	⤾	**13.22**	⤾

Table 2.1: Example 2.2.1. CPU times and gain percentages. Convergence is checked every d iterations. Left: $d = 1$. Right: $d = 10$.

n	s	m	memory whole V_m $s \cdot m$	reduced mem. alloc. $3s$	CPU time (secs)
21904	1	444	444	3	1.44
21904	4	319	1276	12	2.35
21904	8	250	2000	24	3.74

Table 2.2: Example 2.2.1. Memory requirements with and without full storage, and CPU time of the second Lanczos sweep.

easily implemented. The gain in the evaluation of the residual norm is still remarkable, but less impressive from the global point of view. Indeed, the basis construction represents the majority of the computational efforts; in particular, the linear solves $A^{-1}\mathcal{V}_i^{(2)}$, $i = 1, \ldots, 52$, required 17.60 seconds.

	time res (secs)	gain	time tot (secs)	gain
`lyap`	11.25	75.9%	75.53	7.7%
`lyapchol`	6.05	55.2%	70.76	1.5%
`lyap2`	6.68	59.4%	73.01	4.5%
`cTri`	**2.71**	⤾	**69.70**	⤾

Table 2.3: Example 2.2.2. CPU times and gain percentages.

Example 2.2.3. In this example, we compare the standard and the extended Krylov approaches again for solving the standard Lyapunov equation. We consider the matrix $A \in \mathbb{R}^{n \times n}$, $n = 39304$, coming from the discretization by isogeometric analysis (IGA) of the

3D Laplace operator on the unit cube $[0,1]^3$ with zero Dirichlet boundary conditions and a uniform mesh. Since high degree B-splines are employed as basis functions (here the degree is 4 but higher values are also common), this discretization method yields denser stiffness and mass matrices than those typically obtained by low degree finite element or finite difference methods; in our experiment, 1.5% of the components of A is nonzero. See, e.g., [30] for more details on IGA.

For the right-hand side we set $C_1 = \texttt{rand}(n, s)$, $s = 3, 8$, $C_1 = C_1/\|C_1\|_F$. In the standard Krylov method the residual norm is computed every 20 iterations. The convergence can be checked every d iterations in the extended approach as well, with d moderate to avoid excessive wasted solves with A at convergence [97]. In our experiments the computation of the residual norm only takes a small percentage of the total execution time and we can afford taking $d = 1$. In both approaches, the residual norm is computed by Algorithm 2.2. Table 2.4 collects the results.

	m	whole V_m mem. alloc.	reduced mem. alloc.	time res (secs)	two-pass (secs)	time tot (secs)
			$s = 3$			
St. Krylov	280	840	9	1.59	20.75	44.56
Ex. Krylov	30	180	180	0.09	-	85.54
			$s = 8$			
St. Krylov	260	2080	24	3.84	45.35	93.49
Ex. Krylov	27	216	216	0.57	-	347.99

Table 2.4: Example 2.2.3. Performance comparison of Standard and Extended Krylov methods.

The standard Krylov method generates a large space to converge for both values of s. Nonetheless, the two-pass strategy allows us to store only 9 basis vectors for $s = 3$ and 24 basis vectors for $s = 8$. This feature may be convenient if storage of the whole solution process needs to be allocated in advance. By checking the residual norm every 20 iterations, the standard Krylov method becomes competitive with respect to the extended procedure, which is in turn penalized by the system solutions with dense coefficient matrices. Indeed, for $s = 3$ the operation $A^{-1}\mathcal{V}_i^{(2)}$ for $i = 1, \ldots, 30$ takes 32.75 secs, that is 38.29% of the overall execution time required by the extended Krylov subspace method. Correspondingly, for $s = 8$ the same operation performed during 27 iterations takes 152.92 secs, that is, 44.94% of the overall execution time. This example emphasizes the potential of the enhanced classical approach when system solves are costly, in which case rational methods pay a higher toll.

Example 2.2.4. In this example, a Sylvester equation (2.0.1) is solved. The coefficient matrices $A, B \in \mathbb{R}^{n\times n}$, $n = 16384$, come from the discretization by centered finite differences of the partial differential operators

$$\mathcal{L}_A(u) = (e^{-xy}u_x)_x + (e^{xy}u_y)_y \quad \text{and} \quad \mathcal{L}_B(u) = (\sin(xy)u_x)_x + (\cos(xy)u_y)_y\,,$$

on $[0,1]^2$ with zero Dirichlet boundary conditions. The right-hand side is a uniformly distributed random matrix where $C_1 = \texttt{rand}(n, s)$, $C_1 = C_1/\|C_1\|_F$ and $C_2 = \texttt{rand}(n, s)$, $C_2 = C_2/\|C_2\|_F$, $s = 3, 8$. Since both A and B are large, equation (2.0.1) is solved by the standard Krylov method presented in Section 2.1.1 and 217 iterations are needed to converge for $s = 3$, and 145 iterations for $s = 8$. The residual norm is checked at each iteration and Table 2.5 collects the results. Two approximation spaces, $\mathbf{K}_m^{\square}(A, C_1) = \text{Range}(V_m)$,

$\mathbf{K}_m^{\square}(B, C_2) = \mathrm{Range}(U_m)$, are generated and a two-pass strategy is employed to cut down the storage demand. See Table 2.6.

	time res (secs)	gain	time tot (secs)	gain
	$s = 3$ (217 its)			
`lyap`	60.19	83.6%	65.32	76.2%
`lyap2`	74.05	86.6%	78.08	80.1%
`cTri`	**9.89**	↵	**15.51**	↵
	$s = 8$ (145 its)			
`lyap`	201.28	88.7%	208.93	81.5%
`lyap2`	140.92	83.8%	149.95	74.2%
`cTri`	**22.74**	↵	**38.65**	↵

Table 2.5: Example 2.2.4. CPU times and gain percentages.

n	s	m	memory whole V_m, U_m $2s \cdot m$	reduced mem. alloc. $6s$	CPU time (secs)
16384	3	217	1032	18	2.62
16384	8	145	2320	48	4.93

Table 2.6: Example 2.2.4. Memory requirements with and without full storage, and CPU time of the second Lanczos sweep.

Example 2.2.5. In this last example, we again consider the Sylvester problem (2.0.1), this time stemming from the 3D partial differential equation

$$(e^{-xy}u_x)_x + (e^{xy}u_y)_y + 10u_{zz} = f \quad \text{on } [0,1]^3, \tag{2.2.31}$$

with zero Dirichlet boundary conditions. Thanks to the regular domain, its discretization by centered finite differences can be represented by the Sylvester equation

$$AX + XB = F, \tag{2.2.32}$$

where $A \in \mathbb{R}^{n^2 \times n^2}$ accounts for the discretization in the x, y variables, while $B \in \mathbb{R}^{n \times n}$ is associated with the z variable. The right-hand side $F \in \mathbb{R}^{n^2 \times n}$ takes into account the source term f in agreement with the space discretization. See Section 5.1 for a detailed discussion.

In our experiment, $n = 148$ (so that $n^2 = 21904$) and equation (2.2.32) falls into the case addressed in the last part of Section 2.2.5. The right-hand side is $F = -C_1 C_2^T$ where C_1,C_2 are two different normalized random matrices, $C_j = \mathtt{rand}(n, s)$, $C_j = C_j / \|C_j\|_F$, $j = 1, 2$, and $s = 3,\ 8$. Convergence is checked at each iteration and Table 2.7 collects the results.

The method requires 190 iterations to converge below 10^{-6} for $s = 3$ and 150 for $s = 8$, and a two-pass strategy allows us to avoid the storage of the whole basis $V_m \in \mathbb{R}^{n^2 \times sm}$. See Table 2.8.

	time res (secs)	gain	time tot (secs)	gain
	$s = 3$ (190 its)			
`lyap`	15.47	75.7%	17.88	63.6%
`lyap2`	25.35	85.2%	27.50	76.3%
`cTri`	**3.76**	⤴	**6.51**	⤴
	$s = 8$ (150 its)			
`lyap`	36.99	68.2%	40.90	60.0%
`lyap2`	77.04	84.7%	80.91	79.8%
`cTri`	**11.77**	⤴	**16.35**	⤴

Table 2.7: Example 6.2.1. CPU times and gain percentages.

			memory whole V_m	reduced mem. alloc.	CPU time (secs)
n^2	s	m	$s \cdot m$	$3s$	
21904	3	190	570	9	0.93
21904	8	150	1200	24	1.31

Table 2.8: Example 2.2.4. Memory requirements with and without full storage, and CPU time of the second Lanczos sweep.

2.3 Closing considerations

In the recent literature, projection methods have been shown to be among the most effective procedures for solving Lyapunov and Sylvester equations with low-rank right-hand side.

In case of symmetric coefficient matrices, we have improved the classical algorithm presenting an expression for the residual norm that significantly reduces the cost of monitoring convergence when projection is based on $\mathbf{K}_m^{\square}$ and $\mathbf{EK}_m^{\square}$.

For the standard Krylov approach, the combination with a two-pass strategy makes this algorithm appealing compared with recently developed methods, both in terms of computational costs and memory requirements, whenever data do not allow for cheap system solves.

The proposed enhancements rely on the symmetric block tridiagonal structure of the projected matrices. In case this pattern does not arise, as in the nonsymmetric setting or when general rational Krylov subspaces are employed, different approaches must be considered.

Chapter 3

Sylvester and Lyapunov equations with banded symmetric data

In this Chapter we deal with a different class of equations. In particular, we no longer consider equations with a low-rank right-hand side but we are interested in the numerical solution of large-scale Lyapunov equations

$$AX + XA - C = 0, \tag{3.0.1}$$

where $A \in \mathbb{R}^{n\times n}$ is symmetric and positive definite, $C \in \mathbb{R}^{n\times n}$ is symmetric, and both are large and banded matrices with bandwidth β_A, β_C, respectively[*]. Numerical methods for (3.0.1) with large, banded, and not necessarily low rank right-hand side have not be given attention so far, in spite of possible occurrence of this setting in practical applications; see, e.g., [51, 82, 66].

If A is well conditioned, the entries of X present a decay in absolute value as they move away from the banded pattern of C. Therefore, a banded approximation $\widehat{X} \approx X$ can be sought. This idea was exploited in [51], where two algorithms for computing $\widehat{X}$ were proposed. We show that if A is well conditioned, a matrix-oriented formulation of the conjugate gradient method (CG) provides a quite satisfactory banded approximation at a competitive computational cost.

For general symmetric banded data, the decay pattern of X fades as the conditioning of A worsens, to the point that for ill-conditioned matrices, no appreciable (exponential) decay can be detected in X. Nevertheless, we show that X can be split into two components, which can be well approximated by a banded matrix and by a low-rank matrix, respectively. This observation leads to a feasible numerical procedure for solving (3.0.1) both in terms of CPU time and memory requirements.

In Section 3.2.6, the procedures are then generalized to handle Sylvester equations with symmetric banded data and positive definite coefficient matrices.

[*]The minus in the given matrix C is just for convenience in the derivation of what follows. Equivalent results can be obtained considering equations of the form $AX + XA + C = 0$ as well.

3.1 The case of well conditioned A

In the case when A is well conditioned, it is possible to fully exploit the banded structure of the data, and to substantially maintain it in a suitably constructed approximate solution. To this end, advantage can be taken of recently developed results on the entry decay of function of matrices; see, e.g., [17, 18, 27, 32]. For instance, bounds for the entries of the inverse of $\mathcal{A} := A \otimes I + I \otimes A$ (viewed as a banded matrix with bandwidth $n\beta_A$) have been employed to estimate the decay in the entries of the solution X to (3.0.1).

Theorem 3.1.1 ([51]). *Consider equation (3.0.1) and let*

$$\tau := \frac{1}{2|\lambda_{\max}(A)|} \max\left\{1, \frac{\left(1+\sqrt{\kappa(A)}\right)^2}{2\kappa(A)}\right\}, \quad \text{and} \quad \rho := \left(\frac{\sqrt{\kappa(A)}-1}{\sqrt{\kappa(A)}+1}\right)^{\frac{1}{n\beta_A}},$$

then

$$|(X)_{i,j}| \le \tau \sum_{k=1}^{n}\sum_{\ell=1}^{n} |(C)_{k,\ell}|\, \rho^{|(\ell-j)n+k-i|}. \tag{3.1.2}$$

By exploiting the Kronecker structure of $\mathcal{A}$, sharper bounds for $\left(\mathcal{A}^{-1}\right)_{i,j}$ can be derived, see, e.g., [27], leading to different, and possibly more accurate, estimates for $|(X)_{i,j}|$.

Theorem 3.1.2. *Consider equation (3.0.1) and define $\lambda_1 = \lambda_1(\omega) := \lambda_{\min}(A) + i\omega$, $\lambda_2 = \lambda_2(\omega) := \lambda_{\max}(A) + i\omega$, and $R := \alpha + \sqrt{\alpha^2 - 1}$ where $\alpha := \left(|\lambda_1| + |\lambda_2|\right)/|\lambda_2 - \lambda_1|$. Then*

$$|(X)_{i,j}| \le \sum_{k=1}^{n}\sum_{\ell=1}^{n} \theta_{k,\ell} |(C)_{k,\ell}|, \tag{3.1.3}$$

where

- *If $k \neq i$ and $\ell \neq j$, then*

$$\theta_{k,l} = \frac{64}{2\pi|\lambda_{\max}(A) - \lambda_{\min}(A)|^2} \int_{-\infty}^{\infty} \left(\frac{R^2}{(R^2-1)^2}\right)^2 \left(\frac{1}{R}\right)^{\frac{|k-i|}{\beta_A}+\frac{|\ell-j|}{\beta_A}-2} d\omega.$$

- *If either $k = i$ or $\ell = j$, then*

$$\theta_{k,l} = \frac{8}{2\pi|\lambda_{\max}(A) - \lambda_{\min}(A)|} \int_{-\infty}^{\infty} \frac{1}{\sqrt{\lambda_{\min}(A)^2 + \omega^2}} \frac{R^2}{(R^2-1)^2} \left(\frac{1}{R}\right)^{\frac{|k-i|}{\beta_A}+\frac{|\ell-j|}{\beta_A}-1} d\omega.$$

- *If both $k = i$ and $\ell = j$, then*

$$\theta_{k,\ell} = \frac{1}{2\lambda_{\min}(A)}.$$

Proof. The statement directly comes from [98, Theorem 3.3] summing up on the entries of C. □

We emphasize that since C is banded, only few $(C)_{k,\ell}$ are nonzero, so that only few terms in the summation (3.1.3) are actually computed.

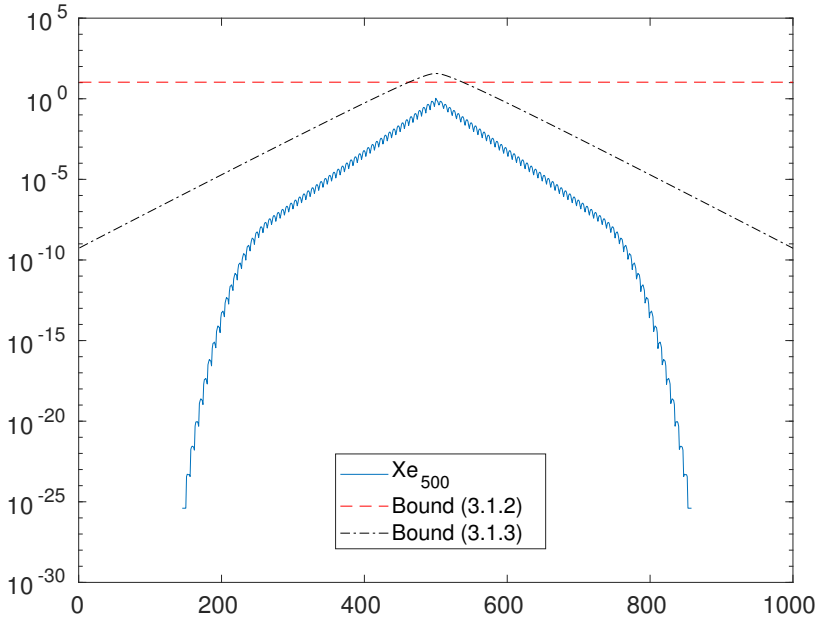

Figure 3.1: Magnitude of Xe_{500} and its estimates (3.1.2) and (3.1.3).

Example 3.1.1. To illustrate the quality of the new bound compared with that in Theorem 3.1.1 we consider the data generated in Example 3.1.2 later in this section. For $6n = 1020$ in Figure 3.1 we report the entry magnitude in logarithmic scale of the 500-th column of the solution X, Xe_{500} (solid line), together with the corresponding computed bounds in (3.1.2) (dashed line) and in (3.1.3) (dashed and dotted line). The new bound correctly captures the decay of the entries, while (3.1.2) predicts a misleading almost flat slope.

Since A is symmetric and positive definite (SPD), the matrix-oriented CG method can be employed in the numerical solution of (3.0.1), in agreement with similar matrix-oriented strategies in the literature; see, e.g., [58] for an early presentation.

An implementation of the procedure is illustrated in Algorithm 3.1.

Several properties of Algorithm 3.1 can be observed. For instance, since C is symmetric, it is easy to show that all the iterates, W_k, X_k, P_k, R_k, are symmetric for all k if a symmetric X_0 is chosen. This implies that only one matrix-matrix multiplication by A in line 2 is needed. Indeed, if $S_k := AP_{k-1}$, then $W_k = AP_{k-1} + P_{k-1}A = AP_{k-1} + (AP_{k-1})^T = S_k + S_k^T$. Furthermore, only the lower – or upper – triangular part of the iterates need be stored, leading to some gain in terms of both memory requirements and number of flops. Some easy tricks have to be adopted to perform the matrix inner products and the Frobenius norms in line 3, 6, 8 as well as for the matrix-matrix products in line 2.

We next show that all the matrices involved in Algorithm 3.1 are banded matrices, with bandwidth linearly depending on k, the number of iterations performed so far. This matrix-oriented procedure is effective in maintaining the banded structure as long as k is moderate, and this is related to the conditioning of the coefficient matrix.

Algorithm 3.1: CG for the Lyapunov matrix equation.

input : $A \in \mathbb{R}^{n\times n}$, A SPD, $C, X_0 \in \mathbb{R}^{n\times n}$ with banded storage, $\epsilon_{res} > 0$, $m_{\max}$
output: $X_k \in \mathbb{R}^{n\times n}$

1 Set $R_0 = C - AX_0 - X_0A$, $P_0 = R_0$
for $k = 1, 2, \ldots, m_{\max}$ **do**
2 $\quad W_k = AP_{k-1} + P_{k-1}A$
3 $\quad \alpha_k = \frac{\|R_{k-1}\|_F^2}{\langle P_{k-1}, W_k\rangle_F}$
4 $\quad X_k = X_{k-1} + P_{k-1}\alpha_k$
5 $\quad R_k = R_{k-1} - W_k\alpha_k$
6 $\quad$ **if** $\|R_k\|_F / \|R_0\|_F < \epsilon_{res}$ **then**
7 $\quad\quad$ **Stop**
$\quad$ **end**
8 $\quad \beta_k = \frac{\|R_k\|_F^2}{\|R_{k-1}\|_F^2}$
9 $\quad P_k = R_k + P_{k-1}\beta_k$
end

Proposition 3.1.3. *If $X_0 = 0$, all the iterates generated by Algorithm 3.1 are banded matrices and, in particular,*

$$\beta_{W_k} \le k\beta_A + \beta_C, \quad \beta_{X_k} \le (k-1)\beta_A + \beta_C, \quad \beta_{R_k} \le k\beta_A + \beta_C, \quad \beta_{P_k} \le k\beta_A + \beta_C.$$

Proof. We first focus on the effects of Algorithm 3.1 on the bandwidth of the current iterates. We recall that if $G, H \in \mathbb{R}^{n\times n}$ are banded matrices with bandwidth β_G, β_H respectively, the matrix GH has bandwidth at most $\beta_G+\beta_H$. The multiplication by A in line 2 of Algorithm 3.1 is the only step that increases the iterate bandwidth at iteration k, therefore we have $\beta_{W_k} \le \beta_A+\beta_{P_{k-1}}$, $\beta_{X_k} \le \max\{\beta_{X_{k-1}}, \beta_{P_{k-1}}\}$, $\beta_{R_k} \le \max\{\beta_{R_{k-1}}, \beta_{W_k}\}$ and $\beta_{P_k} \le \max\{\beta_{R_k}, \beta_{P_{k-1}}\}$. We now demonstrate the statement by induction on k. Since $X_0 = 0$, $R_0 = C$ and $\beta_{R_0} = \beta_{P_0} = \beta_C$. Moreover, for $k = 1$,

$$\beta_{W_1} \le \beta_A + \beta_C, \quad \beta_{R_1} \le \max\{\beta_{R_0}, \beta_{W_1}\} \le \beta_A + \beta_C,$$

$$\beta_{X_1} = \beta_C, \qquad \beta_{P_1} \le \max\{\beta_{R_1}, \beta_{P_0}\} \le \beta_A + \beta_C.$$

Supposing that the statement holds for $k = j - 1 > 1$, we prove it for $k = j$.

$$\beta_{W_j} \le \beta_A + \beta_{P_{j-1}} \le \beta_A + (j-1)\beta_A + \beta_C = j\beta_A + \beta_C,$$

$$\beta_{X_j} \le \max(\beta_{X_{j-1}}, \beta_{P_{j-1}}) \le \beta_{P_{j-1}} \le (j-1)\beta_A + \beta_C,$$

$$\beta_{R_j} \le \max(\beta_{R_{j-1}}, \beta_{W_j}) \le \beta_{W_j} \le j\beta_A + \beta_C,$$

$$\beta_{P_j} \le \max(\beta_{R_j}, \beta_{P_{j-1}}) \le \beta_{R_j} \le j\beta_A + \beta_C.$$

□

A similar result can be shown if X_0 is a banded matrix. Theorem 3.1.3 implies that after k iterations all iterates are banded matrices with bandwidth at most $k\beta_A + \beta_C$. Moreover,

only their lower (or upper) triangular parts are stored so that the number of nonzero entries of each iterate is at most

$$n + \sum_{i=1}^{k\beta_A+\beta_C} (n-i) = n + (k\beta_A+\beta_C)n - \frac{1}{2}(k\beta_A+\beta_C)(k\beta_A+\beta_C-1) = \mathcal{O}(n).$$

Exploiting Theorem 3.1.3, it can be shown that the computational cost of Algorithm 3.1 linearly scales with the problem size n. This is a major saving of the matrix-oriented version of the algorithm, compared with its standard vector-oriented counterpart with $\mathcal{A}$, which would require $\mathcal{O}(n^2)$ operations per iteration.

Corollary 3.1.4. *For small values of k, the computational cost of the k-th iteration of Algorithm 3.1 amounts to $\mathcal{O}(n)$ flops.*

Proof. We first notice that if $G, H \in \mathbb{R}^{n\times n}$ are banded matrices with bandwidth β_G, β_H respectively, the matrix-matrix product GH costs $\mathcal{O}\left(n(2\beta_G+1)(2\beta_H+1)\right)$ flops. Therefore, the number of operations required by line 2 of Algorithm 3.1 is

$$\mathcal{O}\left(2n(2\beta_A+1)(2\beta_{P_{k+1}}+1)\right) = \mathcal{O}\left(2n(2\beta_A+1)(2(k\beta_A+\beta_C)+1)\right) = \mathcal{O}\left(8k\beta_A^2 n\right).$$

Similarly, matrix-matrix products with banded matrices determine the matrix inner products $\langle\cdot,\cdot\rangle_F$, and thus the Frobenius norms $\|\cdot\|_F$, in lines 3 and 8. Finally, again the summations in lines 4,5 and 9 require a number of operations of the order of the number of nonzero entries of the matrices involved, that is $\mathcal{O}(n)$. □

When A is well conditioned, the simple matrix-oriented CG typically outperforms more sophisticated methods proposed in the very recent literature. A typical situation is reported in the next example.

Example 3.1.2. We consider an example from [51], where $A = M \otimes I_6 + I_n \otimes L \in \mathbb{R}^{6n\times 6n}$, $M = \text{tridiag}(e, \underline{e}, e) \in \mathbb{R}^{n\times n}$, $L = \text{tridiag}(e, \underline{a-e}, e) \in \mathbb{R}^{6\times 6}$, $e = -0.34$, $a = 1.36$. The right-hand side is $C = Q \otimes \mathbf{1}\mathbf{1}^T + 0.8 I_{6n}$ where $\mathbf{1} \in \mathbb{R}^6$ is the vector of all ones and $Q = \text{tridiag}(0.1, \underline{0.2}, 0.1) \in \mathbb{R}^{n\times n}$; note the change of sign in A and D compared with [51]. Both matrices A and C are block tridiagonal with blocks of size 6 and $\beta_A = 6$, $\beta_C = 11$. Furthermore, thanks to the Kronecker structure of A, it easy to provide an estimate of its condition number which turns out to be independent of n as $\lambda_{\max}(A) = \lambda_{\max}(M) + \lambda_{\max}(L)$ and $\lambda_{\min}(A) = \lambda_{\min}(M) + \lambda_{\min}(L)$. Since M and L are tridiagonal Toeplitz matrices, we can explicitly compute their spectrum and it holds $\lambda_{\max}(L) = a - e + 2|e|\cos(\frac{\pi}{7})$, $\lambda_{\min}(L) = a - e + 2|e|\cos(\frac{6}{7}\pi)$, $\lambda_{\max}(M) = e + 2|e|\cos(\frac{\pi}{n+1})$ and $\lambda_{\min}(M) = e + 2|e|\cos(\frac{n}{n+1}\pi)$. See, e.g., [100]. Therefore,

$$\begin{aligned}
\kappa(A) &= \frac{\lambda_{\max}(A)}{\lambda_{\min}(A)} = \frac{a + 2|e|\left(\cos(\frac{\pi}{7}) + \cos(\frac{\pi}{n+1})\right)}{a + 2|e|\left(\cos(\frac{6}{7}\pi) + \cos(\frac{n}{n+1}\pi)\right)} = \frac{a + 2|e|\left(\cos(\frac{\pi}{7}) + \cos(\frac{\pi}{n+1})\right)}{a - 2|e|\left(\cos(\frac{\pi}{7}) + \cos(\frac{\pi}{n+1})\right)} \\
&\leq \frac{a + 2|e|\left(\cos(\frac{\pi}{7}) + 1\right)}{a - 2|e|\left(\cos(\frac{\pi}{7}) + 1\right)} \leq 40, \quad \text{for all } n.
\end{aligned}$$

The matrix A is thus well-conditioned and Algorithm 3.1 can be employed in the solution process. By using classical CG convergence results (see, e.g., [2, Section 13.2.1]), it follows that $\bar{k} = 44$ iterations will be sufficient to obtain a relative error (in energy norm) less than 10^{-6} for all n[†]. The solution $X_{\bar{k}}$ will be a banded matrix with bandwidth $\beta_{X_{\bar{k}}} \leq 43\beta_A + \beta_C = 269$.

[†]In the experiment we actually need 45 iterations to converge as the stopping criterion is based on the relative residual norm and not on the relative error (in energy norm).

We next apply Algorithm 3.1 for different values of n and relative residual tolerance 10^{-6}, and we compare the method performance with that of the second procedure described in [51]. This method consists in a gradient projection method applied to $\min_X \|C-AX-XA\|_F^2$ where the initial guess is chosen as a coarse approximation to the integral in (3.2.4). We employ the same setting suggested by the authors; see [51] for details. The results are collected in Table 6.2 where the CPU time is expressed in seconds. In the first instance, Algorithm 3.1 is stopped as soon as the relative residual norm satisfies the stopping criterion. In the second instance, a fixed number of iterations for Algorithm 3.1 is used, so as to obtain the same final approximate solution bandwidth as that of the procedure in [51]. With this second instance, we are able to directly compare the accuracy and efficiency of CG and of the method in [51].

$6n$	CG (Algorithm 3.1)				CG (Algorithm 3.1)				Algorithm [51]		
	Its.	β_X	Time	**Res.**	Its.	$\boldsymbol{\beta_X}$	Time	Res.	β_X	Time	Res.
10200	45	275	17.1	8.4e-7	8	53	0.7	1.2e-1	53	123.1	5.5e-1
102000	45	275	170.8	8.4e-7	8	53	4.6	1.2e-1	53	1880.2	5.5e-1
1020000	45	275	1677.2	8.4e-7	8	53	56.9	1.2e-1	53	23822.9	5.5e-1

Table 3.1: Algorithm 3.1 and the second procedure presented in [51] applied to Example 3.1.2. Results for different values of $6n$. For CG, in bold is the quantity used in the stopping criterion.

Because the condition number is bounded independently of n, the number of CG iterations is also bounded by a constant independent of n; this is clearly shown in the table. Therefore the total CPU time to satisfy a fixed convergence criterion scales linearly with n. The results illustrated in Table 6.2 show that Algorithm 3.1 is very effective, in terms of CPU time, while it always reaches the desired residual norm, when this is used as stopping criterion. This is not the case for the algorithm in [51], which would probably require a finer parameter tuning to be able to meet all stopping criteria.

If the final bandwidth is the stopping criterion, the obtained accuracy is comparable with the results of algorithm [51], however CG is many orders of magnitude faster.

The situation changes significantly if A is ill conditioned, since a larger number of iterations will be required to determine a sufficiently good approximation. This difficulty is not a peculiarity of the method, but rather it reflects the fact that the exact solution X cannot be well represented by a banded matrix. Therefore, any acceleration strategy to reduce the CG iteration count will necessarily end up constructing a denser approximation. In this case, a different strategy needs to be devised, and this is discussed in the next section.

3.2 A new method for ill-conditioned A

If A is ill-conditioned, the entries of the solution X to (3.0.1) do not have, in general, a fast decay away from the diagonal, so that a banded approximation is usually not sufficiently accurate. By using the closed-form (ii) presented in Section 1.2 for the matrix X that can be written as

$$X = \int_0^{+\infty} e^{-tA} C e^{-tA} dt, \tag{3.2.4}$$

where the minus in the matrix exponential is due to the positive definiteness of A, we next derive a splitting of the matrix X that leads to a memory saving approximation.

Theorem 3.2.1. *Let $X(\tau) = \int_0^\tau e^{-tA}Ce^{-tA}dt$, for $\tau > 0$, so that $X \equiv X(+\infty)$. For $\tau > 0$ the solution matrix X in (3.2.4) can be written as*

$$X = X(\tau) + e^{-\tau A}Xe^{-\tau A}. \tag{3.2.5}$$

Proof. We can split X as $X = \int_0^\tau e^{-tA}Ce^{-tA}dt + \int_\tau^{+\infty} e^{-tA}Ce^{-tA}dt$, where the first term is $X(\tau)$. Performing the change of variable $t = s + \tau$ it holds

$$\begin{aligned}\int_\tau^{+\infty} e^{-tA}Ce^{-tA}dt &= \int_0^{+\infty} e^{-(s+\tau)A}Ce^{-(s+\tau)A}ds \\ &= e^{-\tau A}\int_0^{+\infty} e^{-sA}Ce^{-sA}ds e^{-\tau A} = e^{-\tau A}Xe^{-\tau A}.\end{aligned}$$

□

The splitting in (3.2.5) emphasizes two components in the solution matrix X. If τ is sufficiently large, the second term is clearly numerically low rank, since $e^{-\tau A}$ is numerically low rank. Depending on the magnitude of τA, the following Theorem 3.2.2 proved in [18] ensures that the first term is banded. As a result, Theorem 3.2.1 provides a splitting of X between its banded and numerically low rank parts. Our new method aims at approximating these two terms separately, so as to limit memory consumptions.

Theorem 3.2.2 ([18])**.** *Let M be Hermitian positive semidefinite with eigenvalues in the interval $[0, 4\rho]$. Assume in addition that M is β_M-banded. For $k \neq \ell$, let $\xi = \lceil |k-\ell|/\beta_M \rceil$, then*

(i) For $\rho t \geq 1$ and $\sqrt{4\rho t} \leq \xi \leq 2\rho t$, $|(e^{-tM})_{k,\ell}| \leq 10\, e^{-\frac{\xi^2}{5\rho t}}$;

(ii) For $\xi \geq 2\rho t$, $|(e^{-tM})_{k,\ell}| \leq 10\frac{e^{-\rho t}}{\rho t}\left(\frac{e\rho t}{\xi}\right)^\xi$.

In our setting, Theorem 3.2.2 can be applied to $e^{-t(A-\lambda_{min}I)}$ by appropriately scaling the original matrix e^{-tA}. For small t, Theorem 3.2.2 ensures that e^{-tA} has small components away from the diagonal so that it can be well approximated by a banded matrix, $\widehat{e^{-tA}} \approx e^{-tA}$; the product $\widehat{e^{-tA}}C\widehat{e^{-tA}}$ is still banded.

With these considerations in mind, we are going to approximate X by estimating the two quantities $X(\tau)$, $e^{-\tau A}Xe^{-\tau A}$ in (3.2.5), for a suitable $\tau > 0$, that is

$$X = X(\tau) + e^{-\tau A}Xe^{-\tau A} \approx X_B + X_L,$$

where the banded matrix X_B approximates the fast decaying portion $X(\tau)$, while X_L approximates the numerically low rank part $e^{-\tau A}Xe^{-\tau A}$.

3.2.1 Approximating $X(\tau)$ by a banded matrix

The approximation of the first term by a banded matrix is obtained with the following steps:

i) We first replace the integral in $X(\tau)$ by an adaptive quadrature formula;

ii) We approximate the two exponential matrix functions by rational counterparts, using a partial fraction expansion;

iii) We truncate the elementary terms in the partial fraction expansion to banded form.

The a-priori accuracy of the first two steps can be estimated by using well established results in the literature applied to the eigendecomposition of A. In the third step, terms of the type $(t_iA - \xi_jI)^{-1}$ are dense, however recent theoretical results ensure that they can be approximated with banded matrices by truncation.

We start with step (i), that is

$$X(\tau) = \int_0^\tau e^{-tA}Ce^{-tA}dt \approx \frac{\tau}{2}\sum_{i=1}^{\ell}\omega_i e^{-t_iA}Ce^{-t_iA}, \tag{3.2.6}$$

where $t_i = \frac{\tau}{2}x_i + \frac{\tau}{2}$, while x_i, ω_i are respectively the nodes and weights of the formula; in our experiments we considered a matrix-oriented version of the adaptive Gauss-Lobatto quadrature in [43, Section 4.5] with given tolerance ϵ_{quad}.

As for step (ii), rational functions provide very accurate approximations to the matrix exponential $e^A \approx \mathcal{R}_\nu(A)$. See, e.g., [3, 28, 106]. In particular, we have investigated Padé and Chebyshev rational approximations.

Padé: The matrix exponential e^A can be well approximated by a diagonal Padé approximant of degree (d, d) if $\|A\|$ is small enough[‡]. We thus satisfy this condition by using the relation $e^A = (e^{2^{-k}A})^{2^k}$, a technique typically called "scaling and squaring". The Padé approximant is known explicitly for all d. See, e.g., [57, Chapter 10]. In this case the evaluation of the matrix exponential requires $2d + 3 + k$ matrix multiplication and one inversion where $k = \lceil \log_2 \|A\| \rceil$. This strategy is also implemented in the MATLAB function `expm`.

Chebyshev: Since A is supposed to be positive definite, the matrix exponential e^{-tA} can be approximated by a rational Chebyshev function that is uniformly accurate for every positive value of t. See, e.g., [87]. The rational function is of the form

$$e^x \approx \frac{\theta_1}{x - \xi_1} + \ldots + \frac{\theta_d}{x - \xi_d}.$$

Given the poles and the weights in the above expansion, this strategy requires d inversions and additions. See Section 3.2.2 for a numerical procedure to compute the poles and weights ξ_i, θ_i.

Remark 3.2.3. In general, evaluating the matrix exponential e^{-t_iA} by means of the Padé approximant performs better when t_iA has a moderate norm. When $t_i \|A\|_2$ is large the squaring phase becomes the bottleneck of the computation. In this case we should rely on the rational Chebyshev expansion, which has a cost independent of $\|A\|_2$.

In our setting rational Chebyshev functions in $\mathbb{R}^+$ appear to be appropriate. They admit the following partial fraction expansion

$$\mathcal{R}_\nu(A) = \sum_{j=1}^{\nu}\theta_j(A - \xi_jI)^{-1}, \tag{3.2.7}$$

where $\theta_j, \xi_j \in \mathbb{C}$ are its weights and (distinct) poles, respectively. For A real, the poles ξ_j are complex conjugate, yielding the simplified form

$$\mathcal{R}_\nu(A) = \sum_{\substack{j=1,\\ j\ odd}}^{\nu-1} 2\mathrm{Re}\left(\theta_j\left(t_iA - \xi_jI\right)^{-1}\right) + \theta_\nu\left(t_iA - \xi_\nu I\right)^{-1}, \tag{3.2.8}$$

[‡]The exact choice of the ball where Padé is accurate enough depends on the desired accuracy and the value of d.

where ξ_ν is the real pole of $\mathcal{R}_\nu$ if ν is odd. The formula is well defined. Indeed, since A is symmetric, the matrix $t_iA-\xi_jI$ is invertible if ξ_j has nonzero imaginary part. In case of a real ξ_ν, a direct computation shows that $\xi_\nu<0$ for $\nu\in\{1,\dots,13\}$, ν odd, so that $t_iA-\xi_\nu I$ is nonsingular as well. We refer the reader to Section 3.2.2 for details on the computation of the weights and poles of the rational Chebyshev function (3.2.7). The number ν of terms in (3.2.7) is closely related to the accuracy of the computed approximation. Indeed, it holds (see, e.g., [28])

$$\sup_{\lambda\geq 0}|e^{-\lambda}-\mathcal{R}_\nu(\lambda)|\approx 10^{-\nu};$$

a similar estimate holds for $\|e^{-A}-\mathcal{R}_\nu(A)\|$ for A symmetric. Few terms are needed to obtain a quite accurate approximation, for our purposes.

The rational function approximation (3.2.8) requires the computation of several inverses of the form $(t_iA-\xi_jI)^{-1}$ for all $i=1,\dots,\ell$, $j=1,\dots,\nu$, which are, in general, dense. This leads to the third approximation step above, that is a banded approximation $\widehat{(t_iA-\xi_jI)^{-1}}\approx(t_iA-\xi_jI)^{-1}$ with bandwidth much smaller than n. The quality of this approximation is ensured by the following result, which takes great advantage of the complexity of the shifts ξ_j's.

Proposition 3.2.4 ([41]). *Let $M=\upsilon_1I+\upsilon_2M_0$ be β_M-banded with M_0 Hermitian and $\upsilon_1,\upsilon_2\in\mathbb{C}$. Define $a:=(\lambda_{\max}(M)+\lambda_{\min}(M))/(\lambda_{\max}(M)-\lambda_{\min}(M))$ and $R:=\alpha+\sqrt{\alpha^2-1}$ with $\alpha=(|\lambda_{\max}(M)|+|\lambda_{\min}(M)|)/|\lambda_{\max}(M)-\lambda_{\min}(M)|$. Then,*

$$\left|\left(M^{-1}\right)_{p,q}\right|\leq\frac{2R}{|\lambda_{\max}(M)-\lambda_{\min}(M)|}B(a)\left(\frac{1}{R}\right)^{\frac{|p-q|}{\beta_M}},\quad p\neq q,\tag{3.2.9}$$

where, writing $a=\zeta_R\cos(\psi)+i\eta_R\sin(\psi)$,

$$B(a):=\frac{R}{\eta_R\sqrt{\zeta_R^2-\cos^2(\psi)}(\zeta_R+\sqrt{\zeta_R^2-\cos^2(\psi)})},$$

with $\zeta_R=(R+1/R)/2$ and $\eta_R=(R-1/R)/2$.

If spectral estimates are available, the entry decay of $(t_iA-\xi_jI)^{-1}$ can be cheaply predicted by means of (3.2.9), so that the sparsity pattern of the banded approximation $\widehat{(t_iA-\xi_jI)^{-1}}$ to $(t_iA-\xi_jI)^{-1}$ can be estimated a-priori, during its computation. The actual procedure to determine $\widehat{(t_iA-\xi_jI)^{-1}}$ is discussed in Section 3.2.2.

The matrix exponential e^{-t_iA} in (3.2.6) is thus approximated by

$$\widehat{\mathcal{R}}_\nu(t_iA):=\sum_{j=1}^{\nu-1}2\mathrm{Re}\left(\theta_j\widehat{(t_iA-\xi_jI)^{-1}}\right)+\theta_\nu\widehat{(t_iA-\xi_\nu I)^{-1}}\approx\mathcal{R}_\nu(t_iA),\quad i=1,\dots,\ell.$$

We notice that the entries of the most external diagonals of $\widehat{R}_\nu(t_iA)$ might be small in magnitude. To further reduce the bandwidth of $\widehat{R}_\nu(t_iA)$, we thus suggest to set to zero those components of $\widehat{R}_\nu(t_iA)$ that are smaller than ϵ_{quad}, that is, we replace the matrix $\widehat{R}_\nu(t_iA)$ with the matrix $\widetilde{\mathcal{R}}_\nu(t_iA)$ defined as follows

$$\widetilde{\mathcal{R}}_\nu(t_iA):=\widehat{R}_\nu(t_iA)-\mathcal{E}_i,\quad(\mathcal{E}_i)_{k,j}:=\begin{cases}\left(\widehat{R}_\nu(t_iA)\right)_{k,j},\ \text{if }\left|\left(\widehat{R}_\nu(t_iA)\right)_{k,j}\right|<\epsilon_{quad},\\ 0,\ \text{otherwise.}\end{cases}\tag{3.2.10}$$

Collecting all these observations, we have

$$X(\tau) \approx \frac{\tau}{2} \sum_{i=1}^{\ell} \omega_i \widetilde{\mathcal{R}}_\nu(t_i A) C \widetilde{\mathcal{R}}_\nu(t_i A) =: X_B, \tag{3.2.11}$$

and the bandwidth β_{X_B} of X_B is such that $\beta_{X_B} \le 2\max_i\{\beta_{\widetilde{\mathcal{R}}_\nu(t_i A)}\} + \beta_D$. The overall procedure for computing X_B is illustrated in Algorithm 3.2.

Algorithm 3.2: Numerical approximation of $X(\tau)$.

input : $A \in \mathbb{R}^{n\times n}$, A SPD, $C \in \mathbb{R}^{n\times n}$, $\nu \in \mathbb{N}$, $\epsilon_B, \epsilon_{quad}, \tau > 0$
output: $X_B \in \mathbb{R}^{n\times n}$, $X_B \approx X(\tau)$

Compute t_i, ω_i, $i = 1,\dots,\ell$, for the Gauss-Lobatto formula (3.2.6)
Compute ξ_j, θ_j, $j = 1,\dots,\nu$, for the rational Chebyshev approximation (3.2.7)
Set $X_B = 0$
for $i = 1,\dots,\ell$ **do**
    For $j = 1,\dots,\nu$ compute $\widehat{(t_i A - \xi_j I)^{-1}}$
    Set $\widehat{R}_\nu(t_i A) := \sum_{j=1}^{\nu-1} 2\mathrm{Re}\left(\theta_j \widehat{(t_i A - \xi_j I)^{-1}}\right) + \theta_\nu \widehat{(t_i A - \xi_\nu I)^{-1}}$
    Compute $\widetilde{\mathcal{R}}_\nu(t_i A)$ as in (3.2.10)
    Set $X_B = X_B + \omega_i \widetilde{\mathcal{R}}_\nu(t_i A) C \widetilde{\mathcal{R}}_\nu(t_i A)$
end
Set $X_B = \frac{\tau}{2} X_B$

3.2.2 Implementation details for computing X_B

In this section we illustrate some details to efficiently implement Algorithm 3.2.

For given coefficients of the numerator and denominator polynomials (see, e.g., [29]), the weights and poles of the rational Chebyshev function (3.2.7) can be computed by the residue theorem, implemented in Matlab via the function `residue`. In our experiments we always used $\nu = 7$, thus providing a maximum attainable accuracy of the order of 10^{-7}.

The approximation of $(t_i A - \xi_j I)^{-1}$ for all considered i's and j's is the most time consuming part of the process to obtain X_B. This is performed by using a sparse approximate inverse approach, which has been extensively studied in the context of preconditioning techniques for solving large scale linear systems; see, e.g., [19, 16, 20]. Furthermore, many packages such as SPAI[§] and FSAIPACK[¶] are available on-line for its computation. Unfortunately, open software seldom handles complex arithmetic, as it occurs here whenever the poles have nonzero imaginary part.

With the notation in Proposition 3.2.4, we have

$$\left|\left((t_i A - \xi_j I)^{-1}\right)_{p,q}\right| \le \frac{2R}{|\lambda_2 - \lambda_1|} B(a) \left(\frac{1}{R}\right)^{\frac{|p-q|}{\beta_A}}, \quad p > 1,$$

and this allows us to explicitly compute only those entries that are above a given tolerance, taking symmetry into account.

[§] `https://cccs.unibas.ch/lehre/software-packages/`
[¶] `http://hdl.handle.net/11577/3132741`

For every column $q = 1, \dots, n$, we compute $\bar{p}_q(t_i, \xi_j)$ such that

$$\bar{p}_q(t_i, \xi_j) = \operatorname{argmin}\left\{ p > 1, \text{ s.t. } \frac{2R}{|\lambda_2 - \lambda_1|} B(a) \left(\frac{1}{R}\right)^{\frac{|p-q|}{\beta_A}} < \epsilon_B \right\},$$

where ϵ_B is a given threshold. Defining $\widehat{p}_q(t_i, \xi_j) := \min\{n, q + \bar{p}_q(t_i, \xi_j)\}$, we calculate $\left((t_i A - \xi_j I)^{-1}\right)_{p,q}$, $q = 1, \dots, n$, $p = q, \dots, \widehat{p}_q(t_i, \xi_j)$. To this end, we perform an LDLt factorization of $t_i A - \xi_j I$, that is $t_i A - \xi_j I = L(t_i, \xi_j) D(t_i, \xi_j) L(t_i, \xi_j)^T$, and solve

$$L(t_i, \xi_j) D(t_i, \xi_j) L(t_i, \xi_j)^T s_q = e_q, \quad q = 1, \dots, n. \tag{3.2.12}$$

Since $t_i A + \xi_j I$ is a β_A-banded matrix, the computation of $L(t_i, \xi_j)$ and $D(t_i, \xi_j)$ requires $\mathcal{O}(n\beta_A)$ flops. We do not compute all the entries of s_q but only the ones in position r, $r = q, \dots, \widehat{p}_q(t_i, \xi_j)$, suitably performing the forward and backward substitution with $L(t_i, \xi_j)$ and $L(t_i, \xi_j)^T$ respectively. Notice that $L(t_i, \xi_j)$ maintains the bandwidth of A. Therefore, the computation of the $\widehat{p}_q(t_i, \xi_j) - q + 1$ entries of s_q costs $\mathcal{O}(2\beta_A(\widehat{p}_q(t_i, \xi_j) - q + 1))$ flops. The computed s_q approximates the q-th column of $(t_i A + \xi_j I)^{-1}$, in particular, $(s_q)_r = ((t_i A + \xi_j I)^{-1} e_q)_r$ for $r = q, \dots, \widehat{p}_q(t_i, \xi_j)$. The overall computational cost of (3.2.12), for all q, amounts to $\mathcal{O}(2\beta_A \sum_{q=1}^{n} (\widehat{p}_q(t_i, \xi_j) - q + 1))$ flops.

If $\mathfrak{S} = [s_1, \dots, s_n]$ and $\mathfrak{s}$ denotes its diagonal, we define $\widehat{(t_i A + \xi_j I)^{-1}} := \mathfrak{S} + \mathfrak{S}^T - \operatorname{diag}(\mathfrak{s})$, and it holds $\|\widehat{(t_i A + \xi_j I)^{-1}} - (t_i A + \xi_j I)^{-1}\|_{\max} < \epsilon_B$. The matrix $\widehat{(t_i A + \xi_j I)^{-1}}$ has to be computed for all $i = 1, \dots, \ell$, $j = 1, \dots, \nu$, leading to a computational cost that amounts to $\mathcal{O}(2\beta_A \sum_{i,j,q} (\widehat{p}_q(t_i, \xi_j) - q + 1))$ flops. In all our numerical experiments we set $\nu = 7$ and, thanks to the observation in (3.2.8), we can compute $\widehat{(t_i A + \xi_j I)^{-1}}$, for $i = 1, \dots, \ell$, and only four terms in j. Fixing $i \in \{1, \dots, \ell\}$, the four matrices $\widehat{(t_i A + \xi_j I)^{-1}}$, $j = 1, 3, 5, 7$, are computed in parallel decreasing the cost of the overall procedure to $\mathcal{O}(2\beta_A \sum_{i,q} \max_j \{\widehat{p}_q(t_i, \xi_j) - q + 1\})$ flops.

3.2.3 Approximating $e^{-\tau A} X e^{-\tau A}$ by a low-rank matrix

We next turn our attention to the second component in (3.2.5), $e^{-\tau A} X e^{-\tau A}$. We show that for large τ this matrix can be well approximated by a low-rank matrix.

Proposition 3.2.5. *Consider the matrix $e^{-\tau A} X e^{-\tau A}$. Then, rank$(e^{-\tau A} X e^{-\tau A}) \searrow 0$ as $\tau \to +\infty$, and there exists a matrix $X_L \in \mathbb{R}^{n \times n}$, rank$(X_L) = \bar{\ell} \ll n$, such that*

$$\|e^{-\tau A} X e^{-\tau A} - X_L\|_2^2 \le \frac{3}{4\lambda_n^2} e^{-2\tau(\lambda_n + \lambda_{n-\bar{\ell}})} \|C\|_F^2, \tag{3.2.13}$$

where $\lambda_1 \ge \dots \ge \lambda_n > 0$ denote the eigenvalues of A.

Proof. Let $A = Q \Lambda Q^T$, $\Lambda = \operatorname{diag}(\lambda_1, \dots, \lambda_n)$, $\lambda_1 \ge \dots \ge \lambda_n > 0$, be the eigendecomposition of A. Then, we can write $e^{-\tau A} X e^{-\tau A} = Q e^{-\tau \Lambda} (Q^T X Q) e^{-\tau \Lambda} Q^T = Q e^{-\tau \Lambda} Y e^{-\tau \Lambda} Q^T$, where $Y \in \mathbb{R}^{n \times n}$ is such that $\Lambda Y + Y \Lambda - Q^T C Q = 0$. We notice that $e^{-\tau \lambda_i} \le e^{-\tau \lambda_j}$ for all $j \le i$ and $e^{-\tau \lambda_i} \to 0$, $\tau \to +\infty$, for all $i = 1, \dots, n$. Hence, $e^{-\tau A} X e^{-\tau A} = Q e^{-\tau \Lambda} Y e^{-\tau \Lambda} Q^T$ is numerically low-rank as $\tau \to +\infty$ since $\operatorname{rank}(e^{-\tau \Lambda}) = \operatorname{rank}\left(\operatorname{diag}(e^{-\tau \lambda_1}, \dots, e^{-\tau \lambda_n})\right) \searrow 0$ as $\tau \to +\infty$.

For a fixed $\bar{\ell}$, we consider the partition $Q = [Q_1, Q_2]$, $Q_1 \in \mathbb{R}^{n \times (n-\bar{\ell})}, Q_2 \in \mathbb{R}^{n \times \bar{\ell}}$, $e^{-\tau \Lambda} = \operatorname{blkdiag}(e^{-\tau \Lambda_1}, e^{-\tau \Lambda_2})$, $\Lambda_1 = \operatorname{diag}(\lambda_1, \dots, \lambda_{n-\bar{\ell}}), \Lambda_2 = \operatorname{diag}(\lambda_{n-\bar{\ell}+1}, \dots, \lambda_n)$, and $Y = [Y_{11}, Y_{12}; Y_{21}, Y_{22}]$

with blocks Y_{st}, $s,t = 1,2$, of conforming dimensions, that is Y_{st} is the solution of the Sylvester equation $\Lambda_s Y_{st} + Y_{st}\Lambda_t - Q_s^T C Q_t = 0$, $s,t = 1,2$. Then,

$$e^{-\tau A} X e^{-\tau A} = Q e^{-\tau \Lambda} Y e^{-\tau \Lambda} Q^T = [Q_1, Q_2] \begin{bmatrix} e^{-\tau \Lambda_1} & \\ & e^{-\tau \Lambda_2} \end{bmatrix} \begin{bmatrix} Y_{11} & Y_{12} \\ Y_{21} & Y_{22} \end{bmatrix} \begin{bmatrix} e^{-\tau \Lambda_1} & \\ & e^{-\tau \Lambda_2} \end{bmatrix} \begin{bmatrix} Q_1^T \\ Q_2^T \end{bmatrix}.$$

Defining $X_L := Q_2 e^{-\tau \Lambda_2} Y_{22} e^{-\tau \Lambda_2} Q_2^T$, $\operatorname{rank}(X_L) = \bar{\ell}$, we have

$$\begin{aligned}
\|e^{-\tau A} X e^{-\tau A} - X_L\|_2^2 &= \left\| [Q_1, Q_2] \begin{bmatrix} e^{-\tau \Lambda_1} & \\ & e^{-\tau \Lambda_2} \end{bmatrix} \begin{bmatrix} Y_{11} & Y_{12} \\ Y_{21} & 0 \end{bmatrix} \begin{bmatrix} e^{-\tau \Lambda_1} & \\ & e^{-\tau \Lambda_2} \end{bmatrix} \begin{bmatrix} Q_1^T \\ Q_2^T \end{bmatrix} \right\|_2^2 \\
&= \left\| \begin{bmatrix} e^{-\tau \Lambda_1} & \\ & e^{-\tau \Lambda_2} \end{bmatrix} \begin{bmatrix} Y_{11} & Y_{12} \\ Y_{21} & 0 \end{bmatrix} \begin{bmatrix} e^{-\tau \Lambda_1} & \\ & e^{-\tau \Lambda_2} \end{bmatrix} \right\|_2^2 \\
&\le \left(\|e^{-\tau \Lambda_1} Y_{11} e^{-\tau \Lambda_1}\|_2 + \|e^{-\tau \Lambda_2} Y_{21} e^{-\tau \Lambda_1}\|_2 + \|e^{-\tau \Lambda_1} Y_{12} e^{-\tau \Lambda_2}\|_2 \right)^2 \\
&\le \left(e^{-2\tau \lambda_{n-\bar{\ell}}} \|Y_{11}\|_2 + e^{-\tau(\lambda_n + \lambda_{n-\bar{\ell}})} \|Y_{21}\|_2 + e^{-\tau(\lambda_n + \lambda_{n-\bar{\ell}})} \|Y_{12}\|_2 \right)^2 \\
&\le \left(e^{-2\tau \lambda_{n-\bar{\ell}}} \|Y_{11}\|_F + e^{-\tau(\lambda_n + \lambda_{n-\bar{\ell}})} \|Y_{21}\|_F + e^{-\tau(\lambda_n + \lambda_{n-\bar{\ell}})} \|Y_{12}\|_F \right)^2 \\
&\le \left(e^{-2\tau \lambda_{n-\bar{\ell}}} + 2 e^{-\tau(\lambda_n + \lambda_{n-\bar{\ell}})} \right)^2 \|Y\|_F^2 \\
&\le \left(e^{-\tau \lambda_{n-\bar{\ell}}} + 2 e^{-\tau \lambda_n} \right)^2 e^{-2\tau \lambda_{n-\bar{\ell}}} \|Y\|_F^2 \le 3 e^{-2\tau(\lambda_n + \lambda_{n-\bar{\ell}})} \|Y\|_F^2.
\end{aligned}$$

Since Y is such that $\Lambda Y + Y\Lambda - Q^T C Q = 0$, it holds $\|Y\|_F^2 \le \frac{\|C\|_F^2}{4\lambda_n^2}$, and we can write

$$\|e^{-\tau A} X e^{-\tau A} - X_L\|_2^2 \le \frac{3}{4\lambda_n^2} e^{-2\tau(\lambda_n + \lambda_{n-\bar{\ell}})} \|C\|_F^2.$$

□

The proof is constructive, since it provides an explicit form for X_L, that is $X_L = Q_2 e^{-\tau \Lambda_2} Y_{22} e^{-\tau \Lambda_2} Q_2^T$, where Λ_2 contains the $\bar{\ell}$ eigenvalues closest to the origin, and the columns of Q_2 constitute the associated invariant subspace basis; Y_{22} is the solution of a reduced Lyapunov equation.

Depending on the eigenvalue distribution, Proposition 3.2.5 shows that a good approximation may be obtained by using only few of the eigenvectors of A, where however $\bar{\ell}$ is not known a priori. Moreover, the computation of $\bar{\ell}$ eigenpairs of a large matrix, though SPD and banded, may be too expensive. We thus propose to employ a Krylov subspace type procedure to capture information on the relevant portion of the eigendecomposition of A. More precisely, let $\mathbf{K}_m(A^{-1}, v) := \operatorname{Range}([v, A^{-1}v, \dots, A^{-m+1}v])$ where $v \in \mathbb{R}^n$ is a random vector with unit norm, let the columns of $V_m = [v_1, \dots, v_m] \in \mathbb{R}^{n \times m}$, $m \ll n$, be an orthonormal basis of $\mathbf{K}_m(A^{-1}, v)$ and $K_m = V_m^T A V_m$. If V_m is such that $e^{-\tau A} \approx V_m e^{-\tau K_m} V_m^T$, then we approximate

$$e^{-\tau A} X e^{-\tau A} \approx V_m \left(e^{-\tau K_m} \left(V_m^T X V_m \right) e^{-\tau K_m} \right) V_m^T. \tag{3.2.14}$$

The use of A^{-1} in the definition of the Krylov subspace $\mathbf{K}_m(A^{-1}, v)$ is geared towards a fast approximation of the smallest eigenvalues of A and the associated eigenvectors, particularly suitable for the approximation of the exponential [107]. We observe that $e^{-\tau A} X e^{-\tau A}$ solves the Lyapunov equation

$$A e^{-\tau A} X e^{-\tau A} + e^{-\tau A} X e^{-\tau A} A - e^{-\tau A} C e^{-\tau A} = 0.$$

Substituting the approximation in (3.2.14) we can define the following residual matrix

$$\begin{aligned}\mathcal{R}_m &= AV_m e^{-\tau K_m}(V_m^T X V_m)e^{-\tau K_m}V_m^T + V_m e^{-\tau K_m}(V_m^T X V_m)e^{-\tau K_m}V_m^T A \\ &\quad -V_m e^{-\tau K_m}(V_m^T C V_m)e^{-\tau K_m}V_m^T.\end{aligned}$$

To complete the approximation, we need to replace $V_m^T X V_m$ with some easily computable quantity $Z_m \approx V_m^T X V_m$, so that the final approximation will be

$$e^{-\tau A}Xe^{-\tau A} \approx V_m\left(e^{-\tau K_m}Z_m e^{-\tau K_m}\right)V_m^T.$$

To this end, we get inspired by the projection methods procedure presented in Section 2.1 and we impose the standard matrix Galerkin condition on the residual matrix $\mathcal{R}_m$, that is $V_m^T\mathcal{R}_m V_m = 0$. Explicitly writing all terms in this matrix equation leads to the solution of the following $m \times m$ Lyapunov equation

$$K_m Z_m + Z_m K_m - C_m = 0, \tag{3.2.15}$$

where $C_m = V_m^T C V_m$. Note that the matrix exponential terms $e^{-\tau K_m}$ simplify. For $m \ll n$ equation (3.2.15) could be solved by decomposition-based methods such as the Bartels-Stewart method [5], or its symmetric version, the Hammarling method [55]. We opt for the explicit computation, since the eigendecomposition is also used to get the final matrix S_m. Following the discussion in Section 2.2.1, if $K_m = \Pi_m \Psi_m \Pi_m^T$, $\Psi_m = \mathrm{diag}(\psi_1, \ldots, \psi_m)$, denotes the eigendecomposition of K_m, equation (3.2.15) is equivalent to

$$\Psi_m \widehat{Z}_m + \widehat{Z}_m \Psi_m - \Pi_m^T C_m \Pi_m = 0, \tag{3.2.16}$$

where $\widehat{Z}_m = \Pi_m^T Z_m \Pi_m$. Since Ψ_m is diagonal, we can write $\left(\widehat{Z}_m\right)_{i,j} = \frac{(\Pi_m^T C_m \Pi_m)_{i,j}}{\psi_i + \psi_j}$. With $\widehat{Z}_m$ at hand, and with its eigendecomposition being $\widehat{Z}_m = W\Theta W^T$, we can set

$$S_m \coloneqq V_m\left(\Pi_m e^{-\tau\Psi_m}W\Theta^{1/2}\right), \qquad \text{so that} \quad e^{-\tau A}Xe^{-\tau A} \approx S_m S_m^T. \tag{3.2.17}$$

A rank reduction of S_m can be performed if some of the diagonal elements of $\Theta^{1/2}$ fall below a certain tolerance, so that the corresponding columns can be dropped. This post-processing gives rise to a thinner matrix S_m, with fewer than m columns.

Assume that the matrix X_B in (3.2.11) has been already computed. Then the space $\mathbf{K}_m(A^{-1}, v)$ is expanded until the residual norm of the original problem

$$\|R\|_F \coloneqq \|A(X_B + S_m S_m^T) + (X_B + S_m S_m^T)A - C\|_F,$$

is sufficiently small. Exploiting the sparsity of X_B and the low-rank property of $S_m S_m^T$, the quantity $\|R\|_F$ can be computed in $\mathcal{O}(sn)$ flops, where $s = \mathrm{rank}(S_m)$, without the construction of the large and dense matrix R. See Section 3.2.4 for more details. The overall procedure is summarized in Algorithm 3.3.

The two-step procedure for the approximation of X provides a threshold for the final attainable accuracy, and in particular for $\|R\|_F$. Indeed, assume that $X_B \neq X(\tau)$. Then the final residual cannot go below the discrepancy $X(\tau) - X_B$ even if the low rank portion of the solution is more accurate. Indeed,

$$\begin{aligned}R &= A(X_B + S_m S_m^T) + (X_B + S_m S_m^T)A - C \\ &= \underbrace{A(X_B - X(\tau)) + (X_B - X(\tau))A}_{} + \underbrace{A(X(\tau) + S_m S_m^T) + (X(\tau) + S_m S_m^T)A - C}_{}.\end{aligned}$$

Algorithm 3.3: Iterative approximation of $e^{-\tau A} X e^{-\tau A}$.

input : $A \in \mathbb{R}^{n\times n}$, A SPD., $C, X_B \in \mathbb{R}^{n\times n}$, $v \in \mathbb{R}^n$, $\tau, \epsilon_{res}, \epsilon_{it} > 0$, $m_{\max} \in \mathbb{N}$
output: $S_m, \in \mathbb{R}^{n\times s}$, $s \ll n$, such that $S_m S_m^T \approx e^{-\tau A} X e^{-\tau A}$

1 Set $\mu = \|C\|_F$
2 Compute the Cholesky factorization $A = LL^T$
3 Set $V_1 = v/\|v\|$
for $m = 1, 2, \ldots$ *until convergence* **do**
4 Expand $K_m = V_m^T A V_m$, $C_m = V_m^T C V_m$
5 Compute the eigendecomposition $K_m = \Pi_m \Psi_m \Pi_m^T$
6 Solve $\Psi_m \widehat{Z}_m + \widehat{Z}_m \Psi_m - \Pi_m^T C_m \Pi_m = 0$
7 Compute the eigendecomposition $\widehat{Z}_m = W \Theta W^T$
8 Set $S_m := V_m \left(\Pi_m e^{-\tau \Psi_m} W \Theta^{1/2}\right)$ and reduce columns if desired
9 Compute $\|R\|_F / \|C\|_F$
10 **if** $\|R\|_F/\|C\|_F < \epsilon_{res}$ ***or*** $|\|R\|_F - \mu|/\|R\|_F < \epsilon_{it}$ ***or*** $m > m_{\max}$ **then**
11 **Stop**
end
12 $\widehat{v} = L^{-T} L^{-1} v_m$
13 $\widetilde{v} \leftarrow$ Orthogonalize $\widehat{v}$ w.r.t. V_m
14 Set $v_{m+1} = \widetilde{v}/\|\widetilde{v}\|$ and $V_{m+1} = [V_m, v_{m+1}]$
15 Set $\mu = \|R\|_F$
end

The quantity $R_{ideal} = A(X(\tau) + S_m S_m^T) + (X(\tau) + S_m S_m^T)A - C$ is the ideal (non-computable) residual one would obtain if the banded part were computed exactly. Therefore, R differs from this ideal residual by the quantity $A(X_B - X(\tau)) + (X_B - X(\tau))A$. In particular,

$$\|R - R_{ideal}\|_F = \|A(X_B - X(\tau)) + (X_B - X(\tau))A\|_F \le 2\,\|A\|_F\,\|X_B - X(\tau)\|_F$$

and

$$|\|R\|_F - \|R_{ideal}\|_F| \le 2\,\|A\|_F\,\|X_B - X(\tau)\|_F.$$

Therefore, even if $S_m S_m^T$ is accurate, $\|R\|_F$ may stagnate at the level of $\|X_B - X(\tau)\|_F$. To limit this stagnation effect, we include a stopping criterion that avoids iterating when the residual stops decreasing significantly, and in all our numerical experiments we set $\epsilon_{it} = \epsilon_{quad}$, where ϵ_{quad} is related to the accuracy of X_B.

3.2.4 Implementation details for computing the low rank part of the solution

We first notice that the update of the matrices $K_m = V_m^T A V_m$, $C_m = V_m^T C V_m$ in line 4 of Algorithm 3.3 only requires the addition of one extra column and row at each iteration. Moreover, for the sake of robustness we perform a full basis orthogonalization at step 13, though in exact arithmetic this would be ensured by the symmetry of A. Alternative computationally convenient strategies would include a selective orthogonalization [84].

The computational core of Algorithm 3.3 is the residual norm calculation in line 9. The sparsity of X_B and the low rank of S_m allow for a cheap evaluation of $\|R\|_F$ without the explicit computation of the dense and large R. To this end, we first write down a quite standard Arnoldi-type relation for A holding for the space $\mathbf{K}_m(A^{-1}, v)$.

Lemma 3.2.6. *For $v \in \mathbb{R}^n$, $v \neq 0$, let the columns of V_m be an orthonormal basis of $\mathbf{K}_m(A^{-1}, v)$ generated by the Arnoldi method, so that $A^{-1}V_m = V_m H_m + v_{m+1}h_{m+1,m}e_m^T$. Let $\eta = \|(I - V_m V_m^T)Av_{m+1}\|$ and $\widehat{v} = (I - V_m V_m^T)Av_{m+1}/\eta$. Then*

$$AV_m = [V_m, \widehat{v}]\, G_m, \quad \text{with} \quad G_m = \begin{bmatrix} I_m & V_m^T A v_{m+1} \\ 0 & \eta \end{bmatrix} \begin{bmatrix} H_m^{-1} \\ -h_{m+1,m} e_m^T H_m^{-1} \end{bmatrix} \in \mathbb{R}^{(m+1)\times m}.$$

Proof. Consider the Arnoldi relation $A^{-1}V_m = V_{m+1}\underline{H}_m = V_m H_m + v_{m+1}h_{m+1,m}e_m^T$, where $\underline{H}_m \in \mathbb{R}^{(m+1)\times m}$, $(\underline{H}_m)_{i,j} = h_{i,j}$, collects the orthogonalization coefficients stemming from the Arnoldi procedure in lines 12–14 in Algorithm 3.3. Premultiplying by A and postmultiplying by H_m^{-1} we get

$$AV_m = V_m H_m^{-1} - Av_{m+1}h_{m+1,m}e_m^T H_m^{-1} = [V_m, Av_{m+1}] \begin{bmatrix} H_m^{-1} \\ -h_{m+1,m} e_m^T H_m^{-1} \end{bmatrix}.$$

Let $\eta\widehat{v} := Av_{m+1} - V_m V_m^T Av_{m+1}$ where $\eta = \|Av_{m+1} - V_m V_m^T Av_{m+1}\|_2$. Then

$$Av_{m+1} = \eta\widehat{v} + V_m V_m^T Av_{m+1} = [V_m, \widehat{v}] \begin{bmatrix} V_m^T Av_{m+1} \\ \eta \end{bmatrix},$$

so that

$$\begin{aligned} AV_m &= [V_m, Av_{m+1}] \begin{bmatrix} H_m^{-1} \\ -h_{m+1,m} e_m^T H_m^{-1} \end{bmatrix} \\ &= [V_m, \widehat{v}] \begin{bmatrix} I_m & V_m^T Av_{m+1} \\ 0 & \eta \end{bmatrix} \begin{bmatrix} H_m^{-1} \\ -h_{m+1,m} e_m^T H_m^{-1} \end{bmatrix} = [V_m, \widehat{v}]\, G_m, \end{aligned}$$

where $G_m \in \mathbb{R}^{(m+1)\times(m+1)}$ and $W_m := [V_m, \widehat{v}]$ has orthonormal columns by construction. □

Proposition 3.2.7. *With the notation of Lemma 3.2.6, let $W_m = [V_m, \widehat{v}]$ and $S_m = V_m\left(\Pi_m e^{-\tau\Psi_m} W \Theta^{1/2}\right) =: V_m\Delta_m$. Moreover, let $R_B = AX_B + X_B A - C$ and $\gamma = \|R_B\|_F$. Then*

$$\|R\|^2 = \gamma^2 + \|J_m\|_F^2 + 2\,\mathit{trace}\left(J_m\left(W_m^T R_B W_m\right)\right),$$

where $J_m = \left[\begin{array}{c|c} I_m \\ 0 \end{array} G_m\right] \begin{bmatrix} 0 & \Delta_m\Delta_m^T \\ \Delta_m\Delta_m^T & 0 \end{bmatrix} \left[\begin{array}{c|c} I_m \\ 0 \end{array} G_m\right]^T \in \mathbb{R}^{(m+1)\times(m+1)}.$

Proof. Recalling that $\|G + H\|_F^2 = \|G\|_F^2 + \|H\|_F^2 + 2\langle G, H\rangle_F$, it holds

$$\begin{aligned} \|R\|_F^2 &= \|A(X_B + S_m S_m^T) + (X_B + S_m S_m^T)A - C\|_F^2 \\ &= \|AS_m S_m^T + S_m S_m^T A\|_F^2 + \|AX_B + X_B A - C\|_F^2 \\ &\quad + 2\langle AS_m S_m^T + S_m S_m^T A, AX_B + X_B A - C\rangle_F. \end{aligned}$$

The banded matrix $R_B = AX_B + X_B A - C$ and its Frobenius norm can be computed once for all at the beginning of Algorithm 3.3. The computation of the additional two terms can be cheaply carried out in $\mathcal{O}(sn)$ flops. We first focus on the matrix $AS_m S_m^T + S_m S_m^T A$. Denoting $\Delta_m := \Pi_m e^{-\tau\Psi_m} W \Theta^{1/2}$, we have

$$AS_m S_m^T + S_m S_m^T A = [V_m, AV_m] \begin{bmatrix} 0 & \Delta_m\Delta_m^T \\ \Delta_m\Delta_m^T & 0 \end{bmatrix} \begin{bmatrix} V_m^T \\ V_m^T A \end{bmatrix}. \tag{3.2.18}$$

Using Lemma 3.2.6 we have

$$AS_mS_m^T + S_mS_m^TA = W_m \underbrace{\left[\begin{array}{c|c} I_m & G_m \\ 0 & \end{array}\right]\begin{bmatrix} 0 & \Delta_m\Delta_m^T \\ \Delta_m\Delta_m^T & 0 \end{bmatrix}\left[\begin{array}{c|c} I_m & G_m \\ 0 & \end{array}\right]^T}_{=:J_m} W_m^T,$$

so that

$$\|AS_mS_m^T + S_mS_m^TA\|_F^2 = \|J_m\|_F^2,$$

and only matrices of order (at most) $m+1$ are involved in the computation of this norm. Concerning the computation of $\langle AS_mS_m^T + S_mS_m^TA, AX_B + X_BA - C\rangle_F$ we have

$$\langle AS_mS_m^T + S_mS_m^TA, R_B\rangle_F = \operatorname{trace}(W_mJ_mW_m^TR_B) = \operatorname{trace}(J_mW_m^TR_BW_m),$$

and, similarly to K_m and C_m, the matrix $W_m^TR_BW_m \in \mathbb{R}^{(m+1)\times(m+1)}$ requires only the two matrix-vector products $W_m^TR_B[v_m, \widehat{v}]$ to be updated at each iteration. □

Although the computation of the residual norm costs $\mathcal{O}(sn)$ flops at each iteration, lines 9–11 still remain among the most expensive steps of the overall procedure for solving (3.0.1) and they are thus performed periodically, say every d iterations.

3.2.5 Complete numerical procedure and the choice of τ

The algorithm we propose, hereafter called LYAP_BANDED, approximates the solution X to (3.0.1) as $X \approx X_B + S_mS_m^T$ where X_B is banded and S_m is low rank. It is important to realize that unless $\tau \to +\infty$, the entries of $S_mS_m^T$ contribute in a significant way towards the solution, and in particular to the nonzero entries of the leading banded part of X. Indeed, even assuming that X_B is exact, that is $X_B = X(\tau)$, we obtain

$$e^{-2\tau\lambda_{\max}(A)} \le \frac{\|X - X_B\|}{\|X\|} \le e^{-2\tau\lambda_{\min}(A)}, \tag{3.2.19}$$

since $\|X - X_B\| = \|e^{-\tau A}Xe^{-\tau A}\| \le \|e^{-\tau A}\|^2\|X\| = e^{-2\tau\lambda_{\min}(A)}\|X\|$, and $\|e^{-\tau A}Xe^{-\tau A}\| \ge \frac{\|X\|}{\|e^{\tau A}\|^2} = e^{-2\tau\lambda_{\max}(A)}\|X\|$.

The performance of LYAP_BANDED crucially depends on the choice of τ. Indeed, a large τ corresponds to a wider bandwidth of $X(\tau)$ and thus to a possibly too wide β_{X_B}. On the other hand, Corollary 3.2.5 says that $e^{-\tau A}Xe^{-\tau A}$ is numerically low rank if $\tau \to +\infty$. Therefore, if the selected value of τ is too small then the numerical rank of $e^{-\tau A}Xe^{-\tau A}$ may be so large that an accurate low rank approximation is hard to determine; see Table 3.5 in Section 3.3. A trade-off between the bandwidth of X_B and the rank of S_m has to be sought. To make the action of $e^{-\tau A}$ scaling-independent, and without loss of generality, equation (3.0.1) can be scaled by $1/\lambda_{\min}(A)$, and this is done in all our experiments. This seemed to also speed-up the computation of the adaptive quadrature formula.

To automatically compute a suitable value of τ we proceed as follows. Intuitively, we fix a maximum value for β_{X_B} and compute the corresponding τ by using the decay estimate of Theorem 3.2.2 applied to $X(\tau)$. If $X(\tau)$ is approximated by the Gauss-Lobatto quadrature formula (3.2.6), the decay in its off-diagonal entries can be estimated by that of $e^{-\tau A}Ce^{-\tau A}$ (for $i = \ell$, $x_i = 1$ and $t_i = \tau$ in (3.2.6)). Note that according to Theorem 3.2.2, the entries of $e^{-\tau A}$ contribute the most to the bandwidth of e^{-tA}, $t \in [0, \tau]$ away from the main diagonal, and thus to the right-hand side of (3.2.6). In addition, following the discussion at the

beginning of Section 3.2, the multiplication by C does not seem to dramatically influence the final bandwidth of $e^{-\tau A}Ce^{-\tau A}$. Let us thus focus on the first column of $e^{-\tau A}$. To apply Theorem 3.2.2 to $e^{-\tau A}$ we fix a value $\beta_{\max} \in \mathbb{N}$ and define $\bar{\xi} := \lceil |\beta_{\max} - 1|/\beta_A \rceil$. For[‖] $\rho = (\lambda_{\max}(A) - \lambda_{\min}(A))/4$ and $\sqrt{4\rho\tau} \le \bar{\xi} \le 2\rho\tau$, we have

$$|(e^{-\tau A})_{\beta_{\max},1}| \le e^{-\tau\lambda_{\min}(A)}|(e^{-\tau(A-\lambda_{\min}(A)I)})_{\beta_{\max},1}| \le 10\, e^{-\frac{\bar{\xi}^2}{5\rho\tau}} e^{-\tau\lambda_{\min}(A)}. \tag{3.2.20}$$

Similarly, for $\bar{\xi} \ge 2\rho\tau$,

$$|(e^{-\tau A})_{\beta_{\max},1}| \le 10\frac{e^{-\rho\tau}}{\rho\tau}\left(\frac{e\rho\tau}{\bar{\xi}}\right)^{\bar{\xi}} e^{-\tau\lambda_{\min}(A)}. \tag{3.2.21}$$

Our aim is to estimate for which τ the quantity $|(e^{-\tau A})_{\beta_{\max},1}|$ is not negligible while the components from $\beta_{\max}+1$ up to n in the same column can be considered as tiny. Since we would like to have a reasonably large value of τ while maintaining $\beta_{\max}$ moderate, we only consider the bound (3.2.20) in our strategy. Indeed, (3.2.21) requires $\bar{\xi} \ge 2\rho\tau$, that is a very large $\beta_{\max}$, to obtain a sizable value of τ. Fixing a threshold ϵ_τ, we can compute τ as

$$\tau_{opt} = \operatorname{argmin}\{t \ge 0 \text{ s.t. } |(e^{-tA})_{\beta_{\max},1}| \ge \epsilon_\tau\}. \tag{3.2.22}$$

In [18] it has been shown that the bounds in Theorem 3.2.2 are rather sharp, leading to correspondingly sharp bounds (3.2.20)–(3.2.21). This allows us to save computational costs by replacing (3.2.22) with

$$\tau := \operatorname{argmin}\{t \ge 0 \text{ s.t. } 10\, e^{-\frac{\bar{\xi}^2}{5\rho t}} e^{-t\lambda_{\min}(A)} \ge \epsilon_\tau\} \approx \tau_{opt},$$

and a direct computation shows that

$$\tau = \frac{1}{10\rho\lambda_{\min}(A)}\left(-5\rho\log(\epsilon_\tau/10) - \sqrt{25\rho^2\log^2(\epsilon_\tau/10) - 20\rho\lambda_{\min}(A)\bar{\xi}^2}\right). \tag{3.2.23}$$

To clarify the discussion, let us consider the vector-valued function $f : \mathbb{R} \to \mathbb{R}^n$, $f_i(t) := 10\, e^{-\frac{\xi_i^2}{5\rho t}} e^{-t\lambda_{\min}(A)}$, $\xi_i = \lceil |i-1|/\beta_A \rceil$, $i = 1, \ldots, n$. Choosing τ as in (3.2.23) ensures that $f_{\bar{\xi}+1}(\tau) \ge \epsilon_\tau$ whereas $f_{\bar{\xi}+1+k}(\tau) < \epsilon_\tau$, $k > 0$, so that also $|(e^{-\tau A})_{\bar{\xi}+1+k,1}| < \epsilon_\tau$. A graphical description is provided in the following Example 3.2.1.

	$t = t_1$	$t = \tau$
$f_{\bar{\xi}}(t)$	$1.27 \cdot 10^{-4}$	$1.74 \cdot 10^{-5}$
$f_{\bar{\xi}+1}(t)$	$7.95 \cdot 10^{-5}$	$1 \cdot 10^{-5}$
$f_{\bar{\xi}+2}(t)$	$4.90 \cdot 10^{-5}$	$5.66 \cdot 10^{-6}$

Table 3.2: Example 3.2.1. Values of $f_{\bar{\xi}+k}(t)$, $k = 0, 1, 2$, $t = t_1, \tau$.

Example 3.2.1. Consider $A = L/\lambda_{\min}(L)$ where $L = \text{tridiag}(-1, \underline{2}, -1) \in \mathbb{R}^{n\times n}$, $n = 200$. Figure 3.2 displays the function f for different values of t and for τ computed by (3.2.23) where $\epsilon_\tau = 10^{-5}$ and $\beta_{\max} = 50$. The range of the y-axis is restricted to $[10^{-15}, 10^2]$ so as to better appreciate the trend of the largest entries of $f(t)$. Since $\beta_{max} = 50$ and $\beta_A = 1$,

[‖] We recall that for the scaled problem, $\lambda_{\min}(A) = 1$, however for the sake of generality we prefer not to substitute its value.

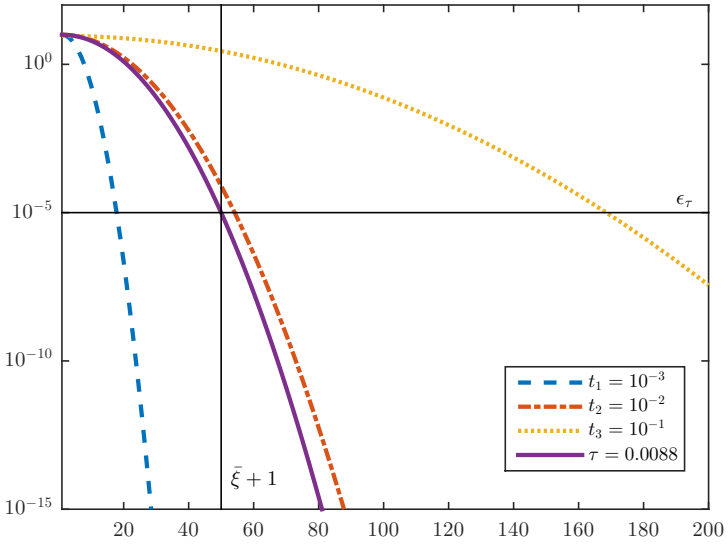

Figure 3.2: $f(t)$ for different values of t and $n = 200$.

it holds that $\bar{\xi} = 49$. For $t = t_1$, $f_{\bar{\xi}+1}(t_1) = 1.11 \cdot 10^{-50} < \epsilon_\tau$ so that t_1 is not a useful value for our purpose. On the other hand, for $t = t_3$, $f_{\bar{\xi}+1}(t_3) = 2.79 \geq \epsilon_\tau$ but also many of the subsequent values satisfy $f_{\bar{\xi}+1+k}(t_3) \geq \epsilon_\tau$. This may lead to an undesired large bandwidth when the rational approximation to $e^{-t_3 A}$ is actually computed. We obtain a similar behavior for $f(t)$ when $t = t_2, \tau$, but only for $t = \tau$ we have that $f_{\bar{\xi}+1}(\tau) \geq \epsilon_\tau$, whereas it holds that $f_{\bar{\xi}+1+k}(\tau) < \epsilon_\tau$, as illustrated in Table 3.2.

The overall procedure for solving (3.0.1) is summarized in the following algorithm.

LYAP_BANDED: Numerical approximation $X \approx X_B + S_m S_m^T$.

Input : $A \in \mathbb{R}^{n\times n}$, A SPD, $C \in \mathbb{R}^{n\times n}$, $\beta_{\max}, \nu, m_{\max} \in \mathbb{N}$, ϵ_τ, ϵ_B, ϵ_{quad}, ϵ_{Res}
Output: $X_B \in \mathbb{R}^{n\times n}$, $S_m \in \mathbb{R}^{n\times s}$, $s \ll n$

1 Compute τ by (3.2.23)
2 Compute X_B by Algorithm 3.2
3 Compute S_m by Algorithm 3.3

Notice that approximations to the extreme eigenvalues of A are necessary to be able to compute τ via (3.2.23). In all our numerical examples, approximations to $\lambda_{\min}(A)$ and $\lambda_{\max}(A)$ were obtained by means of the Matlab function `eigs`.

3.2.6 Numerical solution of the Sylvester equation

The procedure proposed in the previous Sections can be extended to the case of the Sylvester equation,

$$AX + XB - C = 0, \tag{3.2.24}$$

with $A \in \mathbb{R}^{n_A \times n_A}$, $B \in \mathbb{R}^{n_B \times n_B}$ banded and SPD, and $C \in \mathbb{R}^{n_A \times n_B}$ banded. For ease of presentation we consider the case $n = n_A = n_B$, while different n_A, n_B could be considered as well. Once again, the selection of which numerical procedure should be used between those discussed in the previous Sections depends on $\kappa(\mathcal{A})$, where here $\mathcal{A} = B \otimes I + I \otimes A$. In this case, $\kappa(\mathcal{A}) = (\lambda_{\max}(A) + \lambda_{\max}(B))/(\lambda_{\min}(A) + \lambda_{\min}(B))$, therefore the magnitude of $\kappa(\mathcal{A})$ depends on the relative size of the extreme eigenvalues of A and B.

If $\mathcal{A}$ is well-conditioned, Algorithm 3.1 can be applied with straightforward modifications in lines 1 and 2. Notice that, even if C is symmetric, none of the CG iterates is symmetric so that the memory-saving strategies and computational tricks discussed in Section 3.1 cannot be applied. Nevertheless, the bandwidth of the iterates still grows linearly with the number of iterations.

Proposition 3.2.8. *If $X_0 = 0$, all the iterates generated by* CG *applied to equation* (3.2.24) *are banded matrices and, in particular,*

$$\beta_{W_k} \le k \max(\beta_A, \beta_B) + \beta_C, \quad \beta_{X_k} \le (k-1) \max(\beta_A, \beta_B) + \beta_C,$$

$$\beta_{R_k} \le k \max(\beta_A, \beta_B) + \beta_C, \quad \beta_{P_k} \le k \max(\beta_A, \beta_B) + \beta_C.$$

Proof. The same arguments of the proof of Theorem 3.1.3 can be applied noticing that the bandwidth of the matrix $W_k = AP_k + P_kB$ is such that $\beta_{W_k} \le \max(\beta_A, \beta_B) + \beta_{P_k}$. □

If $\mathcal{A}$ is ill conditioned, Algorithm LYAP_BANDED can be generalized to handle the new setting. The solution X can be written as (see, e.g., Chapter 1)

$$X = \int_0^{+\infty} e^{-tA} C e^{-tB} dt = \int_0^{\tau} e^{-tA} C e^{-tB} dt + \int_{\tau}^{+\infty} e^{-tA} C e^{-tB} dt. \tag{3.2.25}$$

A procedure similar to Algorithm 3.2 can be applied to approximate the first integral. Clearly, the presence of two different matrix exponentials increases the computational cost of the method as two approximations $\widehat{R}_\nu(t_i A)$, $\widehat{R}_\nu(t_i B)$ have to computed at each node.

To approximate the second integral addend in (3.2.25) we can generalize Algorithm 3.3. Taking into account the presence of two coefficient matrices, a left and a right space need to be constructed, namely $\mathbf{K}_m(A^{-1}, v)$, $\mathbf{K}_m(B^{-1}, w)$, as it is customary in projection methods for Sylvester equations. See Section 2.1.1.

The choice of τ may be less straightforward in case of (3.2.24). If A and B have similar condition numbers, we suggest to still compute τ by (3.2.23) but replacing $\lambda_{\min}(A)$ by $\lambda_{\min}(D)$, where D is the matrix with the widest bandwidth[**] between A and B.

3.3 Numerical examples

In this section we present numerical experiments illustrating the effectiveness of the method LYAP_BANDED. All results were obtained with Matlab R2015a on a Dell machine with two 2GHz processors and 128 GB of RAM. All reported experiments use the parameter settings in Table 3.3.

[**] Also the computation of ρ in (3.2.23) will change accordingly.

$\epsilon_{res} = 10^{-3}$	relative residual stopping tol (CG, LYAP_BANDED)
$m_{\max} = 2000$	max number of iterations (CG, LYAP_BANDED)
$(\epsilon_\tau, \beta_{\max}) = (10^{-5}, 500)$	setting for the computation of τ in LYAP_BANDED
$(\nu, \epsilon_B, \epsilon_{quad}) = (7, 10^{-7}, 10^{-5})$	truncation and approximation parameters for X_B

Table 3.3: Parameter settings.

Example 3.3.1. We consider the symmetric tridiagonal matrix $A \in \mathbb{R}^{n\times n}$ (thus $\beta_A = 1$) stemming from the discretization by centered finite differences of the 1D differential operator

$$\mathcal{L}u = -\frac{1}{\gamma}\left(e^x u_x\right)_x + \gamma u,$$

on $[0,1]$ with zero Dirichlet boundary conditions. The matrix A is asymptotically ill-conditioned due to the second order term of the operator, and $\kappa(A)$ grows with n. The parameter $\gamma \in \mathbb{R}$ is used to vary the condition number of A. The right-hand side C of (3.0.1) is a diagonal matrix (thus $\beta_C = 0$) with random diagonal entries. We run LYAP_BANDED for different values of n and $\kappa(A)$ and compare its performance with that of Algorithm 3.1. In LYAP_BANDED the parameter τ is computed with the parameters set in Table 3.3. The relative residual norm $\|R\|_F/\|C\|_F$ is computed every $d = 10$ iterations. Table 3.4 collects the results as n and γ vary.

n	γ	$\kappa(A)$	CG (Algorithm 3.1)				LYAP_BANDED				
			Its.	β_X	Time	Res.	τ	β_{X_B}	s	Time	Res.
$4\cdot10^4$	1000	6.61e3	290	289	3.77e2	9.87e-4	2.73	488	7	1.75e3	3.64e-4
	500	2.68e4	583	582	1.57e3	9.92e-4	0.56	578	340	1.94e3	9.73e-4
	200	1.72e5	1475	1474	1.09e4	9.99e-4	0.08	594	366	1.97e3	9.42e-4
$7\cdot10^4$	1800	6.19e3	281	280	6.20e2	9.82e-4	2.98	474	7	3.06e3	2.94e-4
	1000	2.02e4	507	506	2.02e3	9.89e-4	0.76	572	576	3.95e3	9.75e-4
	400	1.29e5	1277	1276	1.41e4	9.98e-4	0.11	592	624	4.16e3	9.92e-4
10^5	2500	6.53e3	288	287	9.11e2	9.94e-4	2.77	486	7	4.40e3	3.17e-4
	1500	1.82e4	481	480	2.56e3	9.96e-4	0.84	570	812	6.77e3	9.73e-4
	500	1.67e5	1456	1455	2.65e4	9.96e-4	0.08	594	892	7.15e3	9.87e-4

Table 3.4: Example 3.3.1. Results for different values of n and γ. $s = \mathrm{rank}(S_m)$. Time is CPU time in seconds.

Algorithm 3.1 is very effective up to $\kappa(A) \approx \mathcal{O}(10^4)$, while for the same $\kappa(A)$ LYAP_BANDED is rather expensive in terms of CPU time compared to CG. The role of the two methods is reversed for $\kappa(A) = \mathcal{O}(10^5)$. In this case, CG takes very many iterations to meet the stopping criterion; the costs of LYAP_BANDED grow far less dramatically, making the method competitive, both in terms of CPU time and storage demand. The bandwidth obtained by CG is lower than that obtained by the banded portion in LYAP_BANDED for the smaller conditions numbers, while the situation is reversed for the largest value of $\kappa(A)$.

Regarding LYAP_BANDED, we notice that for fixed n both β_{X_B} and $\mathrm{rank}(S_m)$ grow with $\kappa(A)$. In particular, $\mathrm{rank}(S_m)$ is consistently much lower for the first value of γ than for the other ones. This can be explained by noticing the quite different value of τ taken as γ varies. This dramatically influences the exponential $\exp(-2\tau)$, and thus the expected error bound for the banded part of the approximation. For instance, for $n = 4\cdot10^4$ we obtain

$$\tau = 2.73, \quad \exp(-2\tau) = 4.3\cdot10^{-3}$$

$$\tau = 0.56, \quad \exp(-2\tau) = 3.2 \cdot 10^{-1}$$
$$\tau = 0.08, \quad \exp(-2\tau) = 8.5 \cdot 10^{-1}$$

Taking into account the error upper bound in (3.2.19), we have $\|X - X_B\| \le \|X - X(\tau)\| + \|X(\tau) - X_B\| \le e^{-2\tau}\|X\| + \|X(\tau) - X_B\|$. Therefore, if X_B is a good approximation to $X(\tau)$, the leading term in the bound is $e^{-2\tau}\|X\|$. For $\tau = 2.73$, the small value of $e^{-2\tau}$ shows that the banded part X_B is already a good approximation to the final solution, so that a very low rank S_m is sufficient to finalize the procedure. This is not the case for the other values of τ.

For similar values of $\kappa(A)$, only $\mathrm{rank}(S_m)$ is affected by an increment in the problem size. This phenomenon is associated with the strategy we adopt for choosing τ. Indeed, a fixed value $\beta_{\max}$ is employed and τ is computed according to (3.2.23); this way τ only depends on the (rescaled) extreme eigenvalues of A, whose magnitude is similar for comparable $\kappa(A)$. Since the n eigenvalues of A seem to spread quite evenly in the interval $[1, \kappa(A)]$ the number $\bar{\ell}$ of eigenvectors required to get an equally accurate low-rank matrix X_L in Corollary 3.2.5 increases with n.

We next set $n = 40000$, $\gamma = 500$. All the other parameters are as before. We vary τ to study how its choice affects the performance of the algorithm. The reference value of τ (first line in Table 3.5) is obtained with the default values of the parameters, as in Table 3.3, and with the automatic procedure of Section 3.2.5. All the other values of τ are selected as 10^j, $j = -2, \dots, 1$.

τ	β_{X_B}	$\mathrm{rank}(S_m)$	Time	Res.
0.56	578	340	1.94e3	9.73e-4
0.01	92	1894	4.46e3	1.14e-2
0.1	270	861	1.52e3	9.81e-4
1	720	270	2.74e3	9.75e-4
10	878	213	5.78e3	1.49e-3

Table 3.5: Example 3.3.1. Results for different values of τ.

As expected, a small τ leads to a very tight bandwidth of X_B but a too large rank of S_m. On the other hand, a very large τ causes an increment in the bandwidth of X_B while a very low-rank S_m is computed. Notice that a proper value of τ is essential also in terms of accuracy of the numerical solution. Indeed, for $\tau = 0.01$, Algorithm 3.3 stops because the maximum number of iterations $m_{\max} = 2000$ is reached, while for $\tau = 10$ a too small residual norm reduction causes a stagnation flag. Good performance is obtained for $\tau = 0.1, 1$, although both values lead to larger memory requirements than those obtained with τ computed by (3.2.23).

Example 3.3.2. We consider the matrix $A \in \mathbb{R}^{n\times n}$ stemming from the discretization by centered finite differences of the 1D differential operator

$$\mathcal{L}(u) = -u_{xx} + \gamma \log(10(x+1))u,$$

on $\Omega = (0,1)$ with zero Dirichlet boundary conditions and $\gamma > 0$. If Ω is discretized by n nodes $(x_1, \dots, x_n)$, we have

$$A = -\frac{(n-1)^2}{12}\mathrm{pentadiag}(-1, 16, \underline{-30}, 16, -1) + \gamma \mathrm{diag}(\chi_1, \dots, \chi_n), \quad \chi_j = \log(10(x_j + 1)),$$

where four neighboring points were used for each grid node. As in the previous example, the matrix A is asymptotically ill-conditioned and γ is chosen to control its condition number, so that $A = A(\gamma)$. The right-hand side C of (3.0.1) is a symmetric tridiagonal matrix with random entries and unit Frobenius norm. Both A and C are banded, with $\beta_A = 2$ and $\beta_C = 1$.

n	γ	$\kappa(A)$	τ	Time X_B (β_{X_B})	Time S_m (s)	Time Tot.	Res.
$4 \cdot 10^4$	5000	7.00e5	2.07e-2	2.69e3 (522)	2.54e2 (431)	2.95e3	9.39e-4
	800	4.20e6	3.45e-3	2.77e3 (522)	2.68e2 (431)	3.04e3	9.49e-4
	300	1.08e7	1.32e-3	2.54e3 (522)	2.58e2 (430)	2.79e3	9.61e-4
$7 \cdot 10^4$	15000	7.27e5	1.99e-2	4.69e3 (522)	1.25e3 (736)	5.94e3	9.88e-4
	2000	5.27e6	2.78e-3	5.05e3 (522)	1.42e3 (735)	6.47e3	9.99e-4
	800	1.28e7	1.16e-3	4.85e3 (522)	1.34e3 (735)	6.19e3	9.97e-4
10^5	50000	4.51e5	3.22e-2	6.61e3 (522)	3.61e3 (1049)	1.02e4	9.85e-4
	5000	4.38e6	3.34e-3	6.87e3 (522)	3.57e3 (1049)	1.04e4	9.94e-4
	200	6.78e7	2.13e-4	7.15e3 (522)	3.64e3 (1046)	1.08e4	9.99e-4

Table 3.6: Example 3.3.2. Results for different values of n and γ. The timings reported are in seconds. $s = \text{rank}(S_m)$.

We solve this problem only by LYAP_BANDED as the large n's and the moderate values of γ we considered lead to sizeable values of $\kappa(A)$. All the thresholds and parameters of the procedure are set as in Table 3.4. In Table 3.6 we collect the results as n and γ vary. We also report the CPU time devoted to the computation of X_B and S_m respectively.

We notice that in this example, the fixed value $\beta_{\max}$ leads to a constant β_{X_B} for all the tested n's. Moreover, for a given n, also the rank of the computed S_m turns out to be almost independent of $\kappa(A)$. This can be intuitively explained by referring to Figure 3.3, where the values of $\exp(-\tau\lambda_j)$ above 10^{-8} are plotted for three automatic selections of τ - as the operator parameter γ changes - and for the smallest eigenvalues of A. The legend also gives the *number* of values above the threshold, for the given τ. Both the distribution and the number of eigenvalues of $A = A(\gamma)$ giving an exponential above the threshold 10^{-8} are approximately the same for all selections of τ, showing that the automatic selection of τ well adapts to the change in the spectrum given by the different γ's.

3.4 Closing considerations

The numerical solution of large-scale Lyapunov equations with non low-rank right-hand side is a very challenging task. In this Chapter we have faced the case of banded symmetric data and positive definite coefficient matrix A.

In case of well-conditioned A, the numerical solution can be satisfactorily approximated by a banded matrix, so that the matrix-oriented CG method has been shown to be a valid candidate for its computation.

If the coefficient matrix is ill-conditioned, no banded good approximation can be determined in general. However, we showed that the solution X can be represented in terms of the splitting $X_B + S_m S_m^T$, with X_B banded and S_m low-rank, and an efficient procedure for computing the pair (X_B, S_m) was presented. Our preliminary numerical results show that the new method is able to compute a quite accurate approximate solution, and that the tuning of the required parameters is not too troublesome.

Both the derivation and the algorithm were extended to the case of Sylvester equations with banded symmetric data and positive definite coefficient matrices.

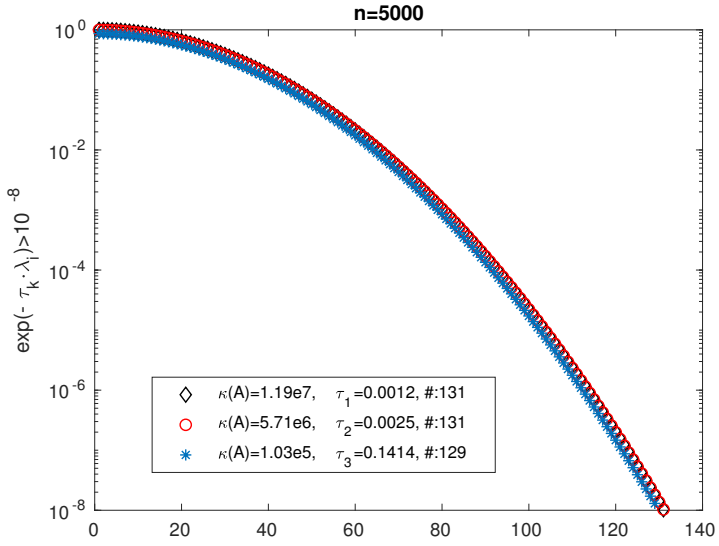

Figure 3.3: Values of $\exp(-\tau_k \lambda_j)$ above the threshold 10^{-8}. Larger eigenvalues of A contribute very little to the value of the exponential.

Chapter 4

Sylvester and Lyapunov equations with quasiseparable symmetric data

In this Chapter we consider equations with a very general structure in their data. In particular, we are interested in Sylvester equations of the form

$$AX + XB - C = 0, \tag{4.0.1}$$

where $A, B, C \in \mathbb{R}^{n\times n}$ are rank-structured and A, B are SPD. More precisely, we assume that the matrices A, B and C are *quasiseparable*. Informally, a matrix is said to be quasiseparable if its off-diagonal blocks are low-rank matrices, and the quasiseparable rank is defined as the maximum of the ranks of the off-diagonal blocks. Therefore, the structures in the equations studied in the previous Chapters, that is a (standard) low-rank right-hand side or banded data, can be viewed as particular instances of the more general quasiseparable framework.

We say that a matrix is numerically quasiseparable when the above property holds only up to a certain ϵ, i.e., only few singular values of each off-diagonal block are above a certain threshold. In this sense, the quasiseparability is often numerically present in X when we have it in A, B and C, so that a low memory requirement is demanded for storing the solution.

A simple yet meaningful example arises from the context of PDEs: consider the differential equation

$$\begin{cases} -u_{xx} - u_{yy} = \log\left(\tau + |x-y|\right), & (x,y) \in \Omega, \\ u(x,y) \equiv 0, & (x,y) \in \partial\Omega, \end{cases} \tag{4.0.2}$$

where $\Omega = [0,1]^2$ and $\tau > 0$. The discretization by centered finite differences of equation (4.0.2) with n nodes in each direction, (x_i, y_j), $i,j = 1,\dots,n$, yields the following Lyapunov equation

$$AX + XA - C = 0, \qquad A, C \in \mathbb{R}^{n\times n},$$
$$C_{i,j} = \log\left(\tau + |x_i - y_j|\right), \qquad A = \frac{1}{h^2}\begin{bmatrix} 2 & -1 & & & \\ -1 & 2 & -1 & & \\ & \ddots & \ddots & \ddots & \\ & & -1 & 2 & -1 \\ & & & -1 & 2 \end{bmatrix},$$
$$h := \tfrac{1}{n-1},$$

Since A is banded it is quasiseparable, but also C shares this property. This is due to the presence of the modulus function that is not regular in the whole domain but it is analytic

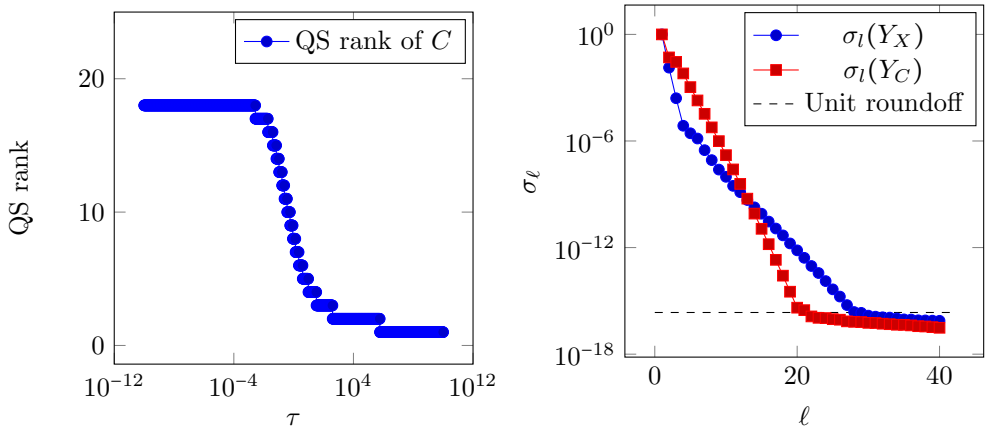

Figure 4.1: On the left, maximum numerical ranks of the off-diagonal blocks of C for different values of τ and $n = 300$, using a threshold of 10^{-14} for truncation. On the right, singular values of $Y_C := C(\frac{n}{2}+1:n, 1:\frac{n}{2})$ and $Y_X := X(\frac{n}{2}+1:n, 1:\frac{n}{2})$ rescaled by the 2-norm of the two blocks respectively for $\tau = 10^{-4}$. The black dashed line indicates the machine precision $2.22 \cdot 10^{-16}$.

when the sign of $x - y$ is constant. This happens in the sub-domains corresponding to the off-diagonal blocks. Separable approximation (and thus low-rank) can be obtained by expanding the source $\log(\tau + |x + y|)$ in the Chebyshev basis. In Figure 4.1 (on the right) we have reported the decay of the singular values of one off-diagonal block of C and X for the case of $\tau = 10^{-4}$ and $n = 300$. In this case the numerical quasiseparable rank of the right-hand side C and the solution X does not exceed 20 and 30, respectively. This property holds for any $\tau > 0$: in Figure 4.1 (on the left) we have checked the quasiseparable rank of the matrix C for various values of τ, and one can see that it is uniformly bounded. The rank is higher when τ is small, because the function is "less regular", and tends to 1 as $\tau \to \infty$, because the off-diagonal blocks tend to a constant in this case.

The problem of solving linear matrix equations whose coefficients are represented as $\mathcal{H}$-matrices has already been addressed in [47, 48]. Recently, in [21, 23] the use of hierarchical matrices in the cyclic reduction iteration for solving quadratic matrix equations has been deeply studied. We will exploit the framework of $\mathcal{H}$-matrices to store quasiseparable matrices and to perform matrix operations at an almost linear cost (up to logarithmic factors).

From a computational point of view, we compare the use of Hierarchical matrices in the matrix sign iteration [91], and in the estimation of the integral formula (3.2.25).

The representation of the solution X by means of the integral formula (3.2.25) has been used in [47] as a theoretical tool to estimate the quasiseparable rank of the solution, but the derived bounds may be very pessimistic, and are linked with the convergence of the integral formula, which cannot be easily made explicit. We improve these estimates by developing a theoretical analysis which relies on some recent results [9], exploited also in [21], where the numerical rank of the solution X is determined by estimating the exponential decay in the singular values of its off-diagonal blocks.

The results that follow are presented in [77] where further analysis and numerical experiments are also discussed.

4.1 Quasiseparable structure in the solution

The main purpose of this section is to prove that, under some reasonable assumptions on the spectrum of A and B, the solution X to the matrix equation (4.0.1) is numerically quasiseparable if A, B and C are quasiseparable.

4.1.1 Quasiseparability

The literature on quasiseparable (or semiseparable) matrices is rather large, and the term is often used to denote slightly different objects. Therefore, we recall the definition of quasiseparable matrices that we will use throughout this Chapter. We refer to [111, 110, 37, 109] and the references therein for a complete survey about quasiseparable and semiseparable structures.

Definition 4.1.1. *A matrix A is* quasiseparable *of order k if the maximum of the ranks of all its submatrices contained in the strictly upper or lower part is exactly k.*

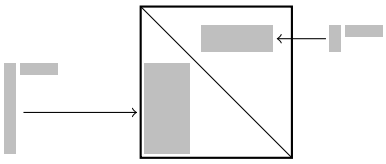

Figure 4.2: Pictorial description of the quasiseparable structure; the off-diagonal blocks can be represented as low-rank outer products.

Example 4.1.1. *A banded matrix with bandwidth k is quasiseparable of order (at most) k. In particular, diagonal matrices are quasiseparable of order 0, tridiagonal matrices are quasiseparable of order 1, and so on.*

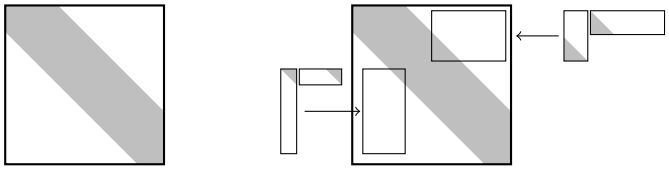

Figure 4.3: Graphic description of the quasiseparability of banded matrices; in grey, the nonzero entries.

4.1.2 Zolotarev problems and decay in the off-diagonal singular values

We are interested in exploiting the quasiseparable rank in numerical computations. In many cases, the request of the exact preservation of a certain structure is too strong – and it cannot be guaranteed. However, for computational purposes, we are satisfied if the property holds in an approximate way, i.e., if our data are well-approximated by structured ones. This can be rephrased by asking that the off-diagonal blocks of the solution X of (4.0.1) have a low

numerical rank. More precisely, given a generic off-diagonal block of the sought solution X, we want to prove that only a limited number of its singular values are larger than $\epsilon \cdot \|X\|_2$, where ϵ is a given threshold. This kind of analysis has been already carried out in [21, 23, 78] for studying the numerical preservation of quasiseparability when solving quadratic matrix equations and computing matrix functions. See also the Ph.D. thesis [76] for more details.

In order to formalize this approach, we extend a result on the singular values of the solution to a Sylvester equation with a low-rank right hand-side. Altough by means of a very different machinery, this result justifies the use of low-rank methods for equation (2.0.1), similarly to the argument presented at the beginning of Chapter 2.

The former approach is based on an old problem considered by Zolotarev at the end of the 19th century [118], which concerns rational approximation in the complex plane. The following version can be found, along with the proof, in [9, Theorem 2.1] or in a similar form in [21, Theorem 4.2].

Theorem 4.1.2. *Let X be an $n \times n$ matrix that satisfies the relation $AX + XB - C = 0$, where C is of rank k and A, B are normal matrices. Let E, F be two disjoint sets containing the spectra of A and $-B$, respectively. Then, the following upper bound on the singular values of X holds,*

$$\frac{\sigma_{1+k\ell}(X)}{\sigma_1(X)} \leq Z_\ell(E,F) := \inf_{r(x)\in\mathcal{R}_{\ell,\ell}} \frac{\max_{x\in E}|r(x)|}{\min_{y\in F}|r(y)|}, \qquad \ell \geq 1,$$

where $\mathcal{R}_{\ell,\ell}$ is the set of rational functions of degree at most (ℓ,ℓ).

Theorem 4.1.2 provides useful information only if one manages to choose the sets E and F well separated. In general it is difficult to explicitly bound $Z_\ell(E,F)$, but some results exist for specific choices of domains, especially when E and F are real intervals; see, e.g., [50, 9]. The combination of these results with Theorem 4.1.2 proves the well-known fact that a Sylvester equation with positive definite coefficients and with a low rank right-hand side has a numerically low-rank solution.

Theorem 4.1.3. *Let A, B be symmetric positive definite matrices with spectrum contained in $[a,b]$, $0 < a < b$. Consider the Sylvester equation $AX + XB - C = 0$, with C of rank k. Then the solution X satisfies*

$$\frac{\sigma_{1+k\ell}(X)}{\sigma_1(X)} \leq 4\rho^{-2\ell}$$

where $\rho = \exp\left(\frac{\pi^2}{2\mu(\frac{b}{a})}\right)$ and $\mu(\cdot)$ is the Grötzsch ring function

$$\mu(\lambda) := \frac{\pi}{2}\frac{K(\sqrt{1-\lambda^2})}{K(\lambda)}, \qquad K(\lambda) := \int_0^1 \frac{1}{(1-t^2)(1-\lambda^2 t^2)}dt.$$

Proof. Applying Theorem 4.1.2 with $E = [a,b]$ and $F = [-b,-a]$ we get

$$\frac{\sigma_{1+k\ell}(X)}{\sigma_1(X)} \leq Z_\ell(E,F).$$

Using Corollary 3.2 in [9] for bounding $Z_\ell(E,F)$ we get the claim. □

Remark 4.1.4. A slightly weaker bound which does not involve elliptic functions is the following [9]

$$Z_\ell([a,b],[-b,-a]) \leq 4\rho^{-2\ell}, \qquad \rho = \exp\left(\frac{\pi^2}{2\log\left(4\frac{b}{a}\right)}\right), \qquad 0 < a < b < \infty.$$

It is easy to see that in case of Lyapunov equations with symmetric positive definite coefficients we can replace the quantity b/a with the condition number of A.

Corollary 4.1.5. *Let A be a symmetric positive definite matrix with condition number $\kappa(A)$, and consider the Lyapunov equation $AX + XA - C = 0$, with C of rank k. Then the solution X satisfies*

$$\frac{\sigma_{1+k\ell}(X)}{\sigma_1(X)} \leq 4\rho^{-2\ell}$$

where $\rho = \exp\left(\frac{\pi^2}{2\mu(\kappa(A))}\right)$ and $\mu(\cdot)$ is defined as in Lemma 4.1.3.

We are now interested in proving that the solution of a Sylvester equation with low-order quasiseparable data is numerically quasiseparable. An analogous task has been addressed in [47]. The approach developed by the authors can be used for estimating either the rank of X in the case of a low-rank right-hand side or the rank of the off-diagonal blocks of X when the coefficients are hierarchical matrices. In particular, it has been shown that if the coefficients are efficiently represented by means of the hierarchical format then also the solution shares this property. The estimates provided in [47] exploit the convergence of a numerical integrating scheme for evaluating the closed integral formula (3.2.25). These bounds are however quite implicit, and are more pessimistic than the estimates provided in [86], and in [102] for the case of a low-rank right hand-side (which is the setting where all the previous results are applicable).

Here, we directly characterize the off-diagonal singular values of the solution applying Theorem 4.1.2 block-wise.

Theorem 4.1.6. *Let A and B be symmetric positive definite matrices of quasiseparable rank k_A and k_B, respectively, and suppose that the spectra of A and B are both contained in the interval $[a, b]$. Then, if X solves the Sylvester equation $AX + XB - C = 0$, with C of quasiseparable rank k_C, a generic off-diagonal block Y of X satisfies*

$$\frac{\sigma_{1+k\ell}(Y)}{\sigma_1(Y)} \leq 4\rho^{-2\ell},$$

where $k := k_A + k_B + k_C$, $\rho = \exp\left(\frac{\pi^2}{2\mu(\frac{b}{a})}\right)$ and $\mu(\cdot)$ is defined as in Lemma 4.1.3.

Proof. Consider the following block partitioning for the Sylvester equation

$$\begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix}\begin{bmatrix} X_{11} & X_{12} \\ X_{21} & X_{22} \end{bmatrix} + \begin{bmatrix} X_{11} & X_{12} \\ X_{21} & X_{22} \end{bmatrix}\begin{bmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{bmatrix} - \begin{bmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{bmatrix} = 0.$$

where the off-diagonal blocks — in each matrix — do not involve any elements of the main diagonal and all the dimensions are compatible. Without loss of generality we can consider the case $Y = X_{21}$. Observe that, writing the above system block-wise we get the following relation

$$A_{21}X_{11} + A_{22}X_{21} + X_{21}B_{11} + X_{22}B_{21} - C_{21} = 0.$$

In particular the block X_{21} solves the Sylvester equation

$$A_{22}X_{21} + X_{21}B_{11} - (C_{21} - A_{21}X_{11} - X_{22}B_{21}) = 0,$$

in which the right-hand side has (standard) rank bounded by k. Since A_{22} and B_{11} are principal submatrices of symmetric positive definite matrices, they are again symmetric positive definite and such that $\kappa(A_{22}) \leq \frac{b}{a}$, and $\kappa(B_{11}) \leq \frac{b}{a}$. Therefore, using Lemma 4.1.3 we get the claim. □

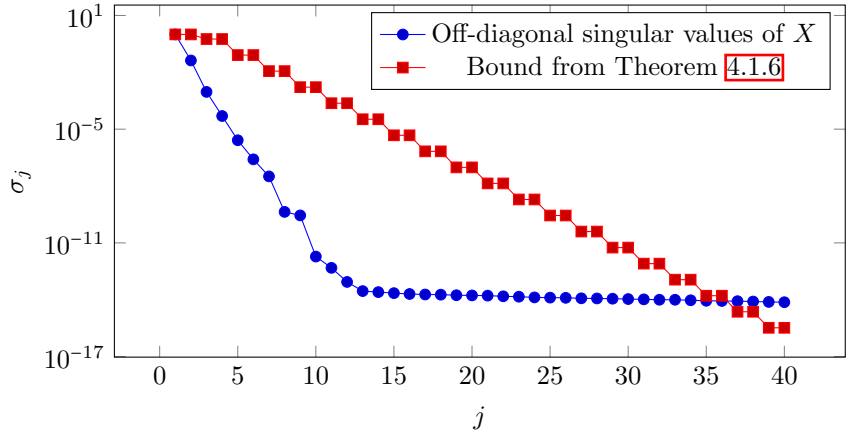

Figure 4.4: Off-diagonal singular values in the solution X to (4.0.1) where C is a random diagonal matrix and $A = B = MM^T$ with M bidiagonal matrix with ones on the main diagonal and random elements – chosen in $(0,1)$ – in the subdiagonal. The dimension of the matrices is $n \times n$ with $n = 300$. The blue dots represent the most significant singular values of the off-diagonal block $X(\frac{n}{2}+1 : n, 1 : \frac{n}{2})$. The red squares represent the theoretical bound given by Theorem 4.1.6.

In Figure 4.1.2 we compare the bound given in Theorem 4.1.6 with the off-diagonal singular values of the solution. In this experiment the matrix $C \in \mathbb{R}^{n\times n}$, $n = 300$, is diagonal with random entries and $A = B = MM^T$ where $M \in \mathbb{R}^{n\times n}$ is bidiagonal with ones on the main diagonal and random elements – chosen in $(0,1)$ – in the subdiagonal. The theoretical bound manages to describe the superlinear decay of the off-diagonal singular values. On the other hand, there is a significant gap between this estimate and the real behavior of the singular values. This is due to the fact that we are bounding the quantity $Z_\ell(E,F)$ where E and F are the convex hulls of $\Lambda(A)$ and $\Lambda(-B)$ respectively, instead of considering the Zolotarev problem directly on the discrete spectra. This is done in order to find explicit bounds but it can cause an overestimation as outlined in [8].

4.1.3 Preservation of the quasiseparable and banded structures

The results of the previous Section guarantee the presence of a numerical quasiseparable structure in the solution X to (4.0.1) when the spectra of A and $-B$ are well separated in the sense of the Zolotarev problem.

In Chapter 3 we studied the preservation of the banded pattern and the banded plus low-rank structure in the solution X in case of Lyapunov and Sylvester equations with banded, thus quasiseparable, data. It is worth noticing that these results do not require the separation property on the spectra of the coefficient matrices. This means that there are cases – not covered by the results of Section 4.1.2 – where the quasiseparability is still preserved.

In order to validate this consideration we set up some experiments concerning the solution to (4.0.1) varying the structure of the coefficients and of the right-hand side. In particular, the features of the solution we are interested in are: the distribution of the singular values σ_ℓ of the off-diagonal block $X(\frac{n}{2}+1 : n, 1 : \frac{n}{2})$[††] and the decay in the magnitude of the elements

[††] Notice that, in order to obtain a good hierarchical representation of the given matrices, the same structure

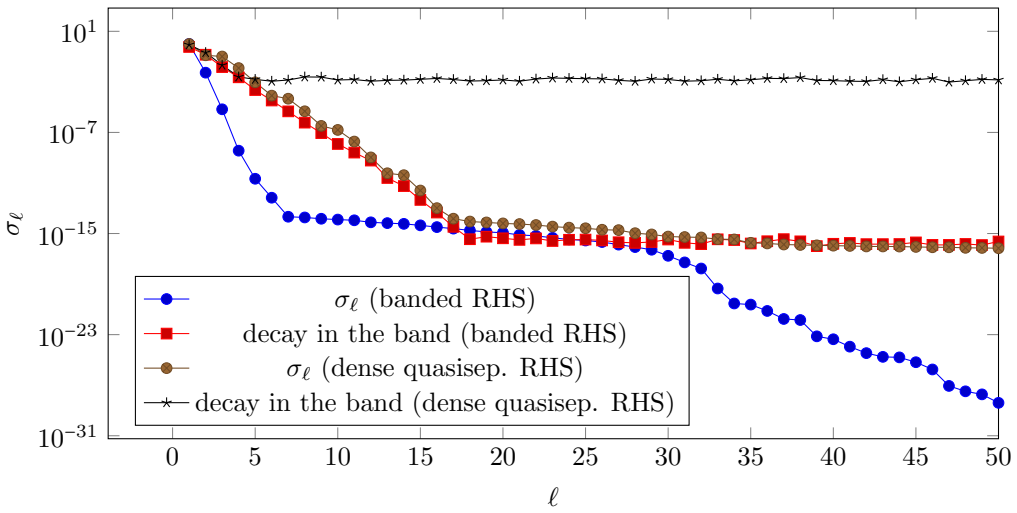

Figure 4.5: Test 1. We compute $X_i \in \mathbb{R}^{n\times n}$, $n = 300$, as the solution of $AX_i + X_iA - C_i = 0$ for $i = 1, 2$ respectively. A is symmetric and tridiagonal with eigenvalues in $[0.2, +\infty)$ (positive definite and well-conditioned). C_1 is tridiagonal symmetric while C_2 is a dense random symmetric quasiseparable matrix of rank 1.

$|(X)_{i,j}|$ getting far from the main diagonal. The latter quantity is represented with the distribution of the maximum magnitude along the subdiagonal ℓ as ℓ varies from 1 to n. In all the performed tests we set $n = 300$ and the solution X is computed by the Bartels-Stewart algorithm [5].

Test 1: We compute X_i as the solution of $AX_i + X_iA - C_=0i$ for $i = 1, 2$. The matrix A is chosen symmetric tridiagonal with eigenvalues in $[0.2, +\infty)$, in particular A is positive definite and well-conditioned. The right-hand side C_1 is taken tridiagonal symmetric with random entries while C_2 is a random dense symmetric matrix with quasiseparable rank 1. In the first case, Theorem 3.1.2 ensures that – numerically – the banded structure is maintained in the solution and this is shown in Figure 4.1.3. Notice that the decay in the off-diagonal singular values is much stronger than the decay in the bandwidth so that, in this example, it is more advantageous to look at the solution as a quasiseparable matrix instead of a banded one. Theorem 4.1.6 guarantees the solution to be quasiseparable also in the second case whereas the banded structure is completely lost.

Test 2: We compute the solution X of $AX + XA - C = 0$. We consider $A = \text{tridiag}(-1, \underline{2}, 1) - 1.99 \cdot I$, so that it is indefinite and ill-conditioned, and we set C equal to a random diagonal matrix. As highlighted in Figure 4.1.3, both the quasiseparable and the band structure are not present in the solution X.

Test 3: We compute X_i as the solution of $AX_i + X_iB - C_i = 0$ for $i = 1, 2$. The matrix A and $-B$ are chosen symmetric and tridiagonal with eigenvalues in $[0.2, 14]$ and $[0.5, 14]$, so

needs to be present also in the upper off-diagonal block, and in the smaller off-diagonal blocks obtained in the recursion. Here we check just the larger off-diagonal block for simplicity; in the generic case, one may expect the quasiseparable rank to be given by the rank of this block.

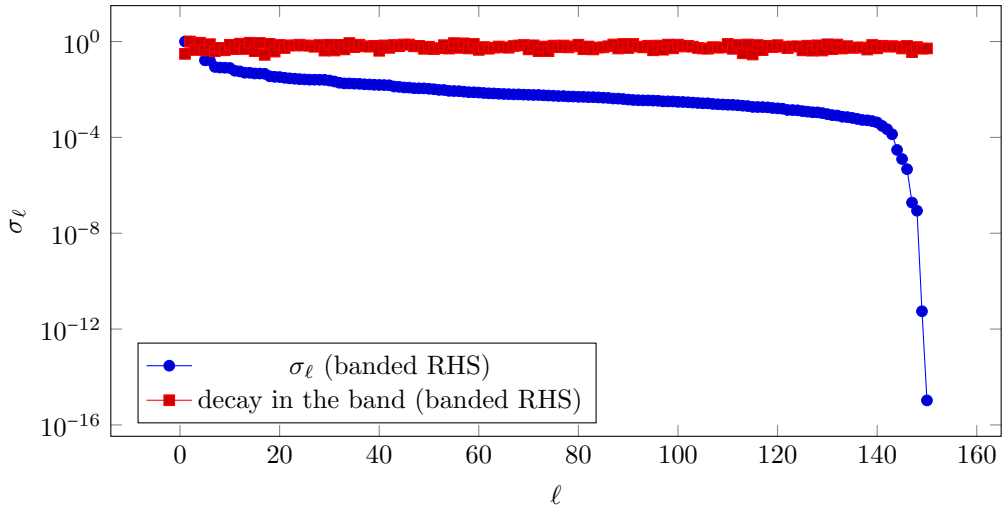

Figure 4.6: Test 2. We compute the solution X of $AX + XA - C = 0$ and we analyze the off-diagonal block $X(\frac{n}{2}+1:n, 1:\frac{n}{2})$. $A = \text{tridiag}(-1, \underline{2}, 1) - 1.99 \cdot I$ (indefinite and ill-conditioned) while C is a random diagonal matrix.

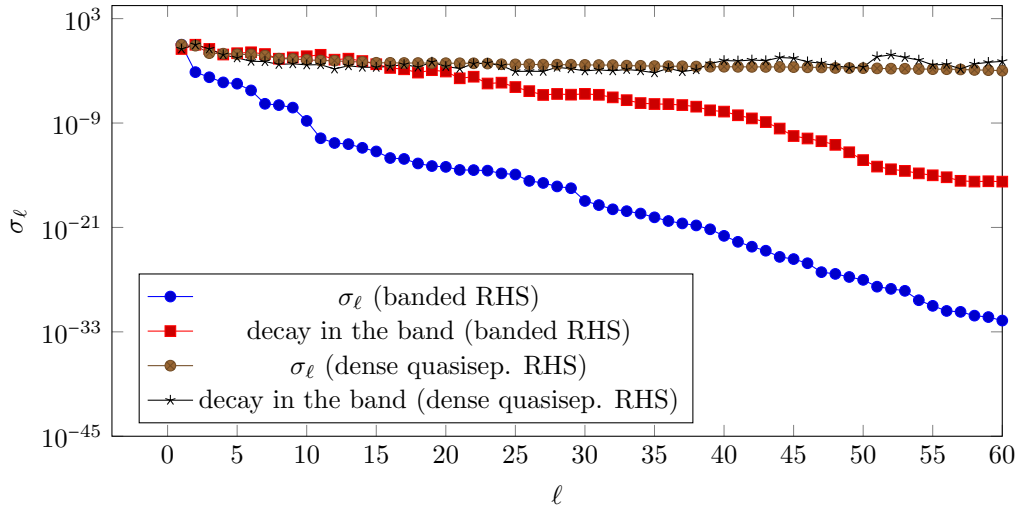

Figure 4.7: Test 3. We compute $X_i \in \mathbb{R}^{n\times n}$, $n = 300$, as the solution of $AX_i - X_iB - C_i = 0$ for $i = 1, 2$, respectively. A and B are symmetric and tridiagonal with eigenvalues in $[0.2, 14]$ and $[0.5, 14]$ (well conditioned but without separation of the spectra). C_1 is tridiagonal symmetric while C_2 is a dense random symmetric quasiseparable matrix of rank 1.

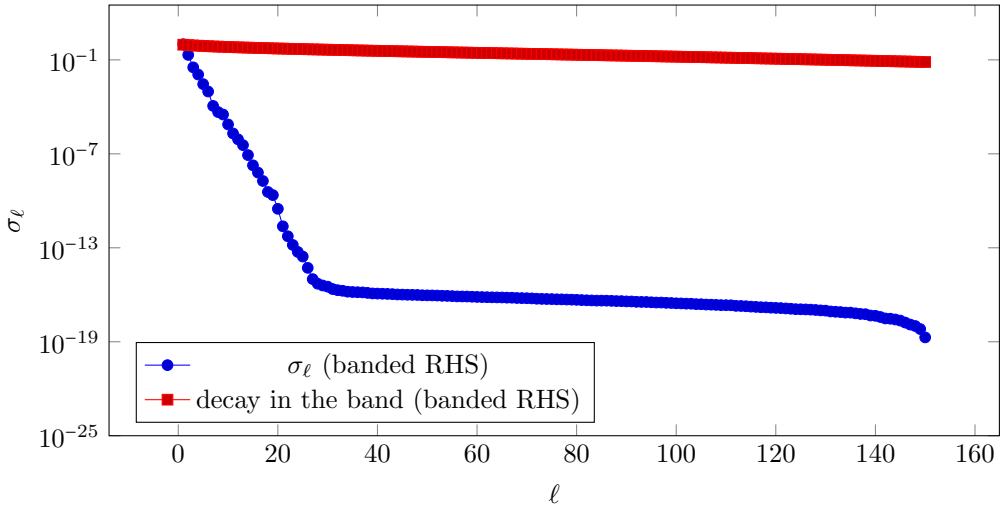

Figure 4.8: Test 4. We compute the solution $X \in \mathbb{R}^{n\times n}$, $n = 300$, of $AX + XA - C = 0$. $A = \text{tridiag}(-1, \underline{2}, 1)$ (positive definite and ill-conditioned) while C is a random diagonal matrix.

both well conditioned but with interlaced spectra. The right-hand side C_1 is chosen tridiagonal symmetric while C_2 is set equal to a random dense symmetric matrix with quasiseparable rank 1. The results in Figure 4.1.3 suggest that both the structures are preserved in the first case and lost in the second case. Once again, in the case of preservation, the decay in the off-diagonal singular values is stronger than the decay in the bandwidth. Notice that, when present, the quasiseparability of the solution cannot be predicted by means of Theorem 4.1.6, but the generalization of Theorem 3.1.2 to the case of the Sylvester equation can be employed to estimate the banded structure of the solution. This test shows how the banded structure is a very particular instance of the more general quasiseparable one.

Test 4: We compute the solution X of $AX + XA - C = 0$. We chose $A = \text{tridiag}(-1, \underline{2}, 1)$, so it is positive definite and ill-conditioned, and we set C equal to a random diagonal matrix. Figure 4.1.3 clearly shows that quasiseparability is preserved while the banded structure is not present in the solution X. In this case, the quasiseparability of the solution can be shown by Theorem 4.1.6. Equivalently, one can exploits the arguments in Section 3.2 to show that the solution can be represented as the sum of a banded matrix and a low-rank one so that X is quasiseparable.

To summarize, the situations where we know that the quasiseparable structure is present in the solution of (4.0.1) are:

(i) A, B and C quasiseparable and spectra of A and $-B$ well separated*;

(ii) A, B and C banded and well-conditioned.

*We consider the spectra to be well separated if Theorem 4.1.6 can be used to prove the quasiseparability. As we have seen, this also includes cases where the spectra are close, such as when they are separated by a line.

On the other hand, for using the computational approach of Section 4.3 we need the spectra of A and $-B$ to be separated by a line.

4.2 HODLR-matrices

An efficient way to store and operate on matrices with an off-diagonal data-sparse structure is to use hierarchical formats. There is a vast literature on this topic. See, e.g., [52, 24, 54] and the references therein. We rely on a particular subclass of the set of Hierarchical representations sometimes called Hierarchically off-diagonal low-rank (HODLR), which can be described as follows; let $A \in \mathbb{C}^{n\times n}$ be a k-quasiseparable matrix, we consider the 2×2 block partitioning

$$A = \begin{bmatrix} A_{11} & A_{22} \\ A_{21} & A_{22} \end{bmatrix}, \qquad A_{11} \in \mathbb{C}^{n_1\times n_1}, \quad A_{22} \in \mathbb{C}^{n_2\times n_2},$$

where $n_1 := \lfloor \frac{n}{2} \rfloor$ and $n_2 := \lceil \frac{n}{2} \rceil$. Since the antidiagonal blocks A_{12} and A_{21} do not involve any element of the main diagonal of A, they have rank at most k, so they are represented as low-rank outer products. Then, the strategy is applied recursively on the diagonal blocks A_{11} and A_{22}. The process stops when the diagonal blocks reach a minimal dimension $n_{\min}$, at which they are stored as full matrices. The procedure is graphically described in Figure 4.9. If $n_{\min}$ and k are negligible with respect to n then the storage cost is linear-polylogarithmic with respect to the size of the matrix.

It is natural to compare the storage demand required by the HODLR representation and the truncation of banded structures, when they are both present in the solution. Consider the following test: we compute the solution X of a Lyapunov equation with a tridiagonal well-conditioned coefficient matrix A and a diagonal right hand-side with random entries. As discussed in the previous Section, the solution has a fast decay in the magnitude of the entries as getting far from the main diagonal. We compare the accuracy obtained when the solution X is stored in the HODLR format with different thresholds in the low-rank truncation of the off-diagonal blocks, and when a fixed number of diagonals are memorized. In particular, the accuracy achieved keeping $5k$ diagonals and truncating the SVD of the off-diagonal blocks using thresholds 10^{-k}, for $k = 0, \dots, 16$, is illustrated in Figure 4.10. We can see that the two approaches have comparable performances for this example. See also Example 4.4.2 in Section 4.4 for a further comparison of the two approaches.

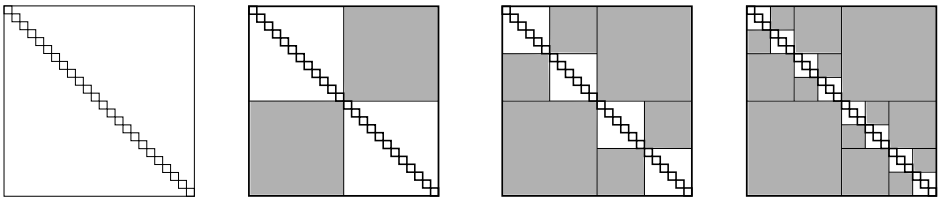

Figure 4.9: The behavior of the block partitioning in the HODLR-matrix representation. The blocks filled with grey are low rank matrices represented in a compressed form, and the diagonal blocks in the last step are stored as dense matrices.

The HODLR format has been studied intensively in the last decade and algorithms with almost linear complexity for computing matrix operations are available, see, e.g., [53, Chapter 3]. Intuitively, the convenience of using this representation in a procedure is strictly related with the growth of the numerical rank of the off-diagonal blocks in the intermediate results.

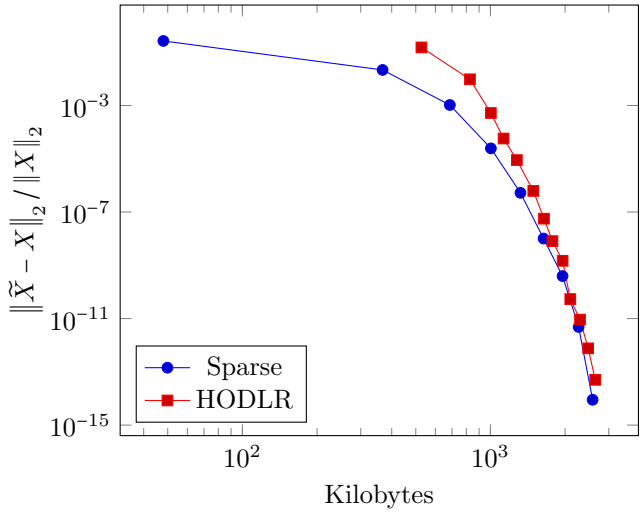

Figure 4.10: Accuracy obtained approximating the solution X to a Lyapunov equation keeping a certain number of diagonals and by truncating the HODLR representations with $n_{min} = 50$ and different thresholds. The plot reports the accuracy obtained with respect to the memory consumption when A is banded and well conditioned and C is a diagonal matrix with random entries. The matrices have dimension $n = 2048$, the storage cost for the dense matrix X is 32678 KB.

While designing our numerical tests, we could not find a MATLAB toolbox focusing on HODLR matrices. Therefore, we developed our own package, called `hm-toolbox`, which is freely available at `https://github.com/numpi/hm-toolbox`[†]. The basic linear algebra operations (sum, product and inversion, as well as LU factorization) are implemented by means of block-wise operations. The off-diagonal blocks are represented as low-rank outer products, and after each arithmetic operation the representation is updated using an economy size SVD which has a linear cost in the size of the matrix. This guarantees that we use an optimal representation at each step. For a complete description of the employed algorithms we refer to [53, 76]. The computational costs of operating with HODLR-matrices are reported in Table 4.1.

Operation	Computational complexity
Matrix-vector multiplication	$O(kn\log(n))$
Matrix-matrix addition	$O(k^2 n\log(n))$
Matrix-matrix multiplication	$O(k^2 n\log^2(n))$
Matrix-inversion	$O(k^2 n\log^2(n))$
Solve linear system	$O(k^2 n\log^2(n))$

Table 4.1: Computational complexity of the HODLR-matrix arithmetic. The integer k is the maximum of the quasiseparable ranks of the inputs while n is the the size of the matrices.

[†]We are in debt to L. Robol and S. Massei for the implementation of the routines in the software package.

4.3 Solving the Sylvester equation

In this Section we show how to deal with the issue of solving (4.0.1) taking advantage of the quasiseparable structure of the data. We first discuss the matrix sign function iteration and then we show how to efficiently evaluate the integral formula (3.2.25). Both these algorithms are implemented in the `hm-toolbox`.

4.3.1 Matrix Sign Function

In [48] Grasedyck, Hackbusch and Khoromskij first showed how to efficiently solve Sylvester equations whose coefficients are representable as hierarchical matrices with low-rank off-diagonal blocks. The procedure consists in plugging the hierarchical matrix arithmetic into the algorithm proposed by Roberts in [91], which requires the computation of the matrix sign function. More precisely, Roberts' algorithm relies on the following result.

Theorem 4.3.1. *Let $A, B \in \mathbb{C}^{n\times n}$ be positive definite, then the solution X to (4.0.1) verifies*

$$X = \frac{1}{2}N_{12}, \tag{4.3.3}$$

where

$$\begin{bmatrix} N_{11} & N_{12} \\ 0 & N_{22} \end{bmatrix} := sign\left(\begin{bmatrix} A & C \\ 0 & -B \end{bmatrix}\right),$$

and – given a square matrix M – we define $sign(M) := \frac{1}{\pi \mathbf{i}}\int_\gamma (zI - M)^{-1}dz$ with γ a path of index 1 around the eigenvalues of M with positive real part.

The sign function of a square matrix $S := \mathrm{sign}(M)$ can be approximated applying the Newton's method to the equation $X^2 - I = 0$ with starting point $S_0 = M$. This requires to compute the sequence

$$S_0 = M, \qquad S_{i+1} = \frac{1}{2}(S_i + S_i^{-1}), \tag{4.3.4}$$

which converges to S, provided that M has no eigenvalues on the imaginary axis [48]. Rewriting (4.3.4) block-wise, applied to $M := \begin{bmatrix} A & C \\ 0 & -B \end{bmatrix}$, yields

$$A_{i+1} = \frac{1}{2}(A_i + A_i^{-1}), \quad B_{i+1} = \frac{1}{2}(B_i + B_i^{-1}), \quad C_{i+1} = \frac{1}{2}(A_i^{-1}C_iB_i^{-1} + C_i), \tag{4.3.5}$$

where $A_0 = A, B_0 = B$, $C_0 = C$ and $C_{i+1} \to 2X$. As stopping criterion we used the condition

$$\|A_{i+1} - A_i\|_F + \|B_{i+1} - B_i\|_F + \|C_{i+1} - C_i\|_F \le \sqrt{\epsilon},$$

where ϵ is the selected accuracy. This can be heuristically justified saying that since the Newton method is quadratically convergent, if the above quantity is a good estimate of the error of the previous step then we have already obtained the solution at the required precision.

In [48] the computation of (4.3.5) is performed using the hierarchical matrix arithmetic. When an appropriate scaling is performed during the iteration, convergence is reached in few steps [57]. The scaling strategy is crucial to keep the number of iterations of the Newton scheme low and it consists in applying the iteration steps to the matrix

$$M_\alpha := \begin{bmatrix} \alpha A & C \\ 0 & -\alpha^{-1}B \end{bmatrix}.$$

The scaling parameter $\alpha > 0$ can be optimally chosen at every iteration, as shown in [57]. When the spectra of A and B are real, the optimal choice is $\alpha_i = \sqrt{\|S_i^{-1}\|_2 / \|S_i\|_2}$. However, if hierarchical matrix arithmetic is employed, the scaling strategy may introduce a non-negligible error propagation as outlined in [48]. A good trade-off is to scale only in the first iteration. This does not affect the accuracy of the iterative scheme if the matrix S_0 can be exactly represented in the hierarchical format [48, Remark 5.3], and allows to keep the number of iterations proportional to $\log(\max\{\kappa(A), \kappa(B)\})$ [48].

4.3.2 Solution by means of the integral formula

We now propose to apply a quadrature scheme for evaluating the semi-infinite integral in (3.2.25). Thanks to the $\mathcal{H}$-arithmetic employed in the solution process, we do not need to split the integral (3.2.25) into two components as illustrated in Theorem 3.2.1. Indeed, quadrature rules require the computation of matrices of the form $D_i := e^{-t_i A} C e^{-t_i B}$ for certain $t_i \in [0, +\infty[$ that mantain a low quasiseparable rank if A, B and C are quasiseparable. Indeed, as shown in Proposition 3.2.5, D_i is low-rank, and thus quasiseparable, for large t_i whereas arguments from [79] illustrate the preservation the quasiseparable structure in $e^{t_i A}$, $e^{t_i B}$ for moderate t_i. Therefore, all the matrices involved in the approximation of the integral have a low numerical quasiseparable rank and the memory requirements remain moderate.

We proceed as follows. We perform the change of variable $t = f(\vartheta) := L \cdot \cot\left(\frac{\vartheta}{2}\right)^2$ where θ is the new variable and L is a parameter chosen to optimize the convergence. In all our numerical experiments $L = 100$. This is a very common strategy for the approximation of integral over infinite domain, which is discussed in detail by Boyd in [25]. We transform (3.2.25) into

$$X = 2L \int_0^\pi \frac{\sin(\vartheta)}{(1-\cos(\vartheta))^2} \, e^{-Af(\vartheta)} C e^{-Bf(\vartheta)} d\vartheta, \tag{4.3.6}$$

which can be approximated by a Gauss-Legendre quadrature scheme. Other quadrature formulas, as Clenshaw-Curtis rules, can be employed. However, as discussed by Trefethen in [105], the difference between Gauss-Legendre and Clenshaw-Curtis formulas is small. Moreover, in most of our tests, Gauss-Legendre schemes showed some slight computational advantages over Clenshaw-Curtis rules as the cost of computing the integration points is negligible[‡].

The quadrature scheme yields an approximation of (4.3.6) of the form

$$X \approx \sum_{j=1}^{m} \omega_j \cdot e^{-Af_j} C e^{-Bf_j} =: \overline{X}, \tag{4.3.7}$$

where ϑ_j are the Legendre points, $f_j := f(\vartheta_j)$, $\omega_j = 2Lw_j \cdot \frac{\sin(\vartheta_j)}{(1-\cos(\vartheta_j))^2}$ and w_j are the Legendre weights.

Finally, we numerically approximate the quantities e^{-Af_j} and e^{-Bf_j}, which represents the dominant cost of the algorithm. For this task, the rational approximations outlined in Section 3.2.1 can be employed and they have been both implemented in our toolbox. The evaluations of the matrix exponentials $\text{EXPM}(-f_j \cdot A)$, $\text{EXPM}(-f_j \cdot B)$ are performed according to the strategy outlined in Remark 3.2.3 and the overall procedure is summarized in Algorithm 4.1.

[‡]In practice we have precomputed the points for the usual cases, so that an explicit computation of them is never carried out in the numerical experiments.

Algorithm 4.1: Solution of a Sylvester equation with quasiseparable data by (4.3.7).

input : $A, B, C \in \mathbb{R}^{n\times n}$, $m \in \mathbb{N}$
output: $\overline{X} \in \mathbb{R}^{n\times n}$, $\overline{X} \approx X$

1 Set L and compute w_j, ϑ_j, $f_j = L \cdot \cot(\frac{\vartheta_i}{2})^2$, for $j = 1, \dots, m$
2 Set $\overline{X} = 0$
for $j = 1, \dots, m$ **do**
3 | Set $\overline{X} = \overline{X} + w_j \frac{\sin(\vartheta_i)}{(1-\cos\vartheta_i)^2} \cdot \text{EXPM}(-f_j \cdot A) \cdot C \cdot \text{EXPM}(-f_j \cdot B)$
end
4 Set $\overline{X} = 2L \cdot \overline{X}$

4.4 Numerical examples

An extensive computational comparison among different approaches for quasiseparable Sylvester equations – as well as their implementation – is still lacking in the literature, and in this section we perform some numerical experiments trying to fill this gap. To this end, we employ the MATLAB `hm-toolbox` that we have developed while writing [77]. The toolbox – which includes all the tested algorithms – is now freely available at `https://github.com/numpi/hm-toolbox`. All the timings reported are relative to MATLAB 2016a run on a machine with a CPU running at 3066 MHz, 12 cores[§], and 192GB of RAM[¶].

To test the accuracy of our approach we report the relative residual on the linearized system of the computed solution. If $\mathcal{S}$ is the coefficient matrix of the linearized system we measure the relative residual,

$$r(\mathcal{S}, X) := \frac{\|\mathcal{S} \cdot x - c\|_2}{\|\mathcal{S}\|_F \cdot \|x\|_2}, \qquad x = \text{vec}(X), \qquad c = \text{vec}(C),$$

which can be easily shown to be the relative backward error in the Frobenius norm [56]. When we deal with Sylvester problems, we have $\mathcal{S} = I \otimes A + B \otimes I$ with A and B symmetric. This allows to use the — easier to compute — bound

$$\|\mathcal{S}\|_F^2 \geq n(\|A\|_F^2 + \|B\|_F^2),$$

so that

$$r(\mathcal{S}, X) = \frac{\|\mathcal{S} \cdot x - c\|_2}{\|\mathcal{S}\|_F \cdot \|x\|_2} = \frac{\|AX + XB - C\|_F}{\|I \otimes A + B \otimes I\|_F \cdot \|X\|_F} \leq \frac{\|AX + XB - C\|_F}{\sqrt{n(\|A\|_F^2 + \|B\|_F^2)} \cdot \|X\|_F},$$

and we actually compute and check the right-hand side in the above expression. Notice that this never requires to form the large system matrix $\mathcal{S}$, and can be evaluated using the arithmetic of hierarchical matrices when considering large scale problems.

Example 4.4.1. We consider the 2-dimensional (2D) Laplace equation on the unit square $\Omega = [0, 1]^2$

[§] All the available cores have only been used to run the parallel implementation of the solver based on the integral formula. All the other solvers did not exploit the parallelism in the machine.

[¶] Notice that is a different machine than the one used to obtain the results in Sections 2.2.6, 3.3 and 6.2.4.

```
>> n = 2048;
>> hmoption('threshold', 1e-12);
>> hmoption('block-size', 256);
>> f = @(x,y) log(1 + abs(x - y));
>> A = (n-1)^2 * spdiags(ones(n,1) * [ -1 2 -1 ], -1:1, n, n);
>> H = hm('tridiagonal', A);
>> C = hm('chebfun2', f, [-1,1], [-1,1], n);
>> X = lyap(H, C, 'method', 'sign');
>> qsrank(X)
ans =
      13
```

Figure 4.11: Example of a MATLAB session where the `hm-toolbox` is used to compute the solution of a Lyapunov equation involving the 2D Laplacian and a numerically quasiseparable right hand-side.

$$\begin{cases} -u_{xx} - u_{yy} = f(x,y), & (x,y) \in \Omega, \\ u(x,y) = 0, & (x,y) \in \partial\Omega. \end{cases}$$

We construct the matrix A corresponding to the discretization of the 1D second-order derivative by centered finite differences with n nodes

$$A = \frac{1}{h^2} \begin{bmatrix} 2 & -1 & & \\ -1 & \ddots & \ddots & \\ & \ddots & \ddots & -1 \\ & & -1 & 2 \end{bmatrix}, \qquad h = \frac{1}{n-1},$$

so that the discrete problem representing the above equation on an $n \times n$ grid using centered finite differences can be written as $AX + XA - C = 0$; C contains the values of the function $f(x,y)$ on the grid. We consider the case where $f(x,y) = \log(1 + |x - y|)$. As already discussed, this choice yields a numerically quasiseparable right hand-side. This is due to the fact that in the sub-domains corresponding to the off-diagonal blocks, f is analytic and it is well approximated by a sum of few separable functions. One can also exploit this property in order to retrieve the HODLR representation of C; the sampling of a separable function $g(x) \cdot h(y)$ on a square grid provides a matrix of rank 1 and the sampling of g and h yield its generating factors. The computation of the expansion of f in the sub-domains has been performed by means of Chebfun2 [104].

Using `hm-toolbox`, the equation can be solved with few MATLAB commands, as shown in Figure 4.11 for the case $n = 2048$. The function `hmoption` can be used to set some options for the toolbox. In this case we set the relative threshold for the off-diagonal truncation to 10^{-12}, and the minimum size of the blocks to 256. The class `hm` implements the hierarchical structure, and here we initialize it using a sparse tridiagonal matrix. Invoking the `lyap` function uses our implementation specialized for $\mathcal{H}$-matrices. In this example, we used the sign function iteration, which is the default method for the implementation of `lyap`. The quasiseparable rank of the solution (obtained using the function `qsrank`) is 13, which is reasonably small compared to the problem size.

n	$\text{Time}_{\text{Sign}}$	Res_{Sign}	QS rk	Time_{Exp}	$\text{Time}_{\text{ParExp}}$	Res_{Exp}	QS rk	$\text{Time}_{\text{lyap}}$
512	0.71	2.973e-12	13	3.69	1.51	3.921e-09	13	0.85
1024	1.73	4.328e-12	14	9.37	3.21	8.706e-10	14	7.52
2048	4.76	2.027e-11	13	22.78	6.34	7.213e-10	14	80.15
4096	13.33	5.189e-11	15	57.15	14.51	5.734e-11	12	523.16
8192	35.93	3.646e-11	13	136.42	31.82	9.233e-12	11	–
16384	92.83	1.004e-10	14	334.75	70.28	3.138e-12	11	–
32768	245.82	1.549e-10	16	790.28	154.65	1.424e-12	11	–
65536	609.86	1.334e-10	15	1825.20	351.82	8.861e-13	10	–
131070	1474.56	1.575e-10	17	4122.17	763.05	2.025e-12	9	–

Table 4.2: Example 4.4.1. Timings and features of the solution of the Laplacian equation for different grid sizes. For the methods based on the HODLR arithmetic the minimum block size is set to 256 and the relative threshold in truncation is $\epsilon = 10^{-12}$. For small problems we also report the timings of the `lyap` function included in the Control Toolbox in MATLAB. The relative residuals of the Lyapunov equation are reported as well for the different methods. The residuals for the parallel version of Algorithm 4.1 have been omitted since they coincide with the ones of the sequential one. In fact, the two algorithms perform exactly the same computations.

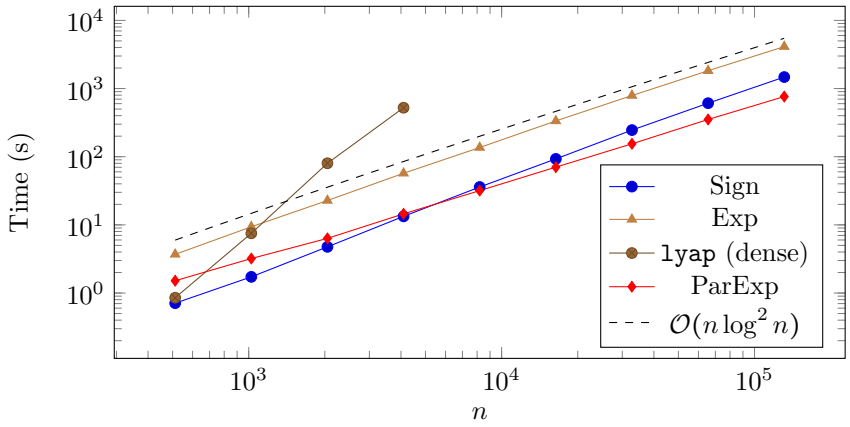

Figure 4.12: Example 4.4.1. Timings for the solution of the Laplacian equation for different grid sizes. The performances of the different algorithms are reported. The dashed line reports the theoretical complexity of $\mathcal{O}(n \log^2(n))$.

In Table 4.2 and Figure 4.12 we show the solution timings for different grid sizes. We stress that, since full matrices are never represented, memory requirements remain limited. The storage of dense matrices is required if the function `lyap` from the Matlab Control Toolbox is employed in the solution process, so that we have comparisons with the latter only for $n \leq 4096$.

The results in Table 4.2 show that the timings are just a little more than linear in the size of the problem. Figure 4.12 illustrates that the complexity is in fact $\mathcal{O}(n \log^2 n)$, for Algorithm 4.1.

The approach based on the sign function iteration is faster than the one that exploits the integral formula. Nevertheless the latter has a slightly better asymptotic cost since it requires $\mathcal{O}(n \log^2(n))$ flops instead of $\mathcal{O}(n \log^3(n))$. Another advantage of the integral formula is the easy parallelization. In fact, the evaluation of the integrand at the nodes can be carried out

$6n$	$\text{Time}_{\text{Sign}}$	Res_{Sign}	$\text{Time}_{\text{ParExp}}$	Res_{Exp}	QS rk	Time_{CG}	Res_{CG}
768	1.06	8.952e-13	1.96	9.441e-12	13	1.17	2.955e-11
1536	2.74	1.416e-12	4.98	4.916e-12	12	2.49	2.810e-11
3072	8.28	9.734e-12	13.11	1.527e-11	12	4.78	2.665e-11
6144	19.30	4.941e-12	32.21	1.075e-11	10	9.23	2.571e-11
12288	48.44	4.763e-12	79.45	1.357e-11	10	18.25	2.412e-11
24576	117.32	4.711e-12	189.84	1.805e-11	10	36.96	3.217e-11
49152	277.82	1.090e-11	445.03	1.621e-11	10	67.18	3.030e-11
98304	589.51	3.872e-11	1092.13	2.692e-11	10	121.31	2.867e-11
196610	1312.63	1.045e-10	2677.12	8.160e-11	9	213.08	2.750e-11

Table 4.3: Example 4.4.2. Timings and features of the solution of the heat equation for different grid sizes. For the methods based on the HODLR arithmetic the minimum block size is set to 256 and the relative threshold in truncation is $\epsilon = 10^{-12}$. In this example the quasiseparable rank of the solution coincides for the implementation based on the sign function and on the integral formula, so we have only reported it once.

in a parallel fashion on different machines or cores. In our tests we used 32 integration nodes (that is $m = 32$ in Algorithm 4.1) so that the maximum gain in the performances can be obtained using 32 cores. The results reported in Table 4.2 confirm the acceleration of the parallel implementation when using 12 cores.

Example 4.4.2. We now consider the same problem of Example 3.1.2. Both the matrices A and C are banded, with bandwidth 6 and 11, respectively. However, a more careful investigation shows that the quasiseparable rank of A is 6, but C is a rank-1 quasiseparable matrix: the quasiseparable representation can exploit more structure than the banded one in this problem.

We have solved this problem for different values of n, from $n = 128$ to $n = 32768$. For each n, the size of the associated matrices A and C is $6n \times 6n$. We have also compared the performance of the quasiseparable approach to the one of Algorithm 3.1 which instead exploits the banded structure of the problem leading to a $\mathcal{O}(n)$ computational cost as shown in Proposition 3.1.4.

Figure 4.13 confirms the predicted $\mathcal{O}(n \log^2 n)$ complexity for the methods that we propose. The timings of CG are comparable to the sign function iteration for small dimensions, but then the absence of the $\log^2(n)$ factor in the complexity is a big advantage for the former method.

The table in Figure 4.13 reports the memory demand when the solution is stored in the HODLR and in the sparse formats. We can see that the method using HODLR matrices, although slower, is more memory efficient compared to CG of a factor of about 2.

4.5 Closing considerations

In this Chapter we have generalized the structure that can be considered when dealing with large-scale Sylvester and Lyapunov equations. In particular, equations with symmetric quasiseparable data have been studied and, warranted by new theoretical results, two efficient solution strategies have been presented.

These methods can be applied also in the cases analyzed in the previous Chapters, that is equations with a low-rank right-hand side or banded symmetric data as the latter structures can be viewed as particular instances of the broaden quasiseparability. However, as shown

$6n$	$\text{Mem}_{\text{HODLR}}$	$\text{Mem}_{\text{Sparse}}$
768	1464	6015
1536	3169	13122
3072	6915	26637
6144	14023	52400
12288	29967	101513
24576	63774	190464
49152	135230	363543
98304	285820	691180
196610	571632	1309512

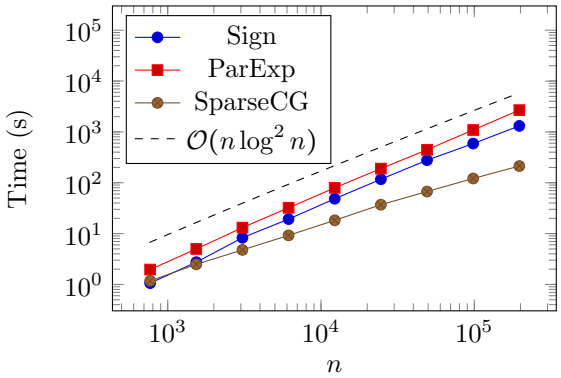

Figure 4.13: Example 4.4.2. On the left, the memory consumption in storing the solution X computed via the parallel version of Algorithm 4.1 and CG, respectively. The first exploits the HODLR representation while the second one makes use of the sparse format. The numerical values reported are in KB (Kilobytes). On the right, the timings for performing the different solution procedures.

in Example 4.4.2, we expect the algorithms exploiting the Hierarchical format to be less competitive with respect to state-of-the-art low-rank methods or LYAP_BANDED. Indeed, the latter algorithms are specifically designed for dealing with low-rank right-hand sides and banded data respectively and they fully exploit these structures.

Chapter 5

Generalized linear matrix equations

This Chapter serves as introduction to Chapter 6, and introduces a class of generalized linear matrix equations stemming from the FD discretization of PDEs under quite general hypotheses. We provide the details of this derivation, highlighting the role of each term in the continuous context. Moreover, closed-forms of the solution of some classes of equations are reported and an efficient method for small-scale problems is proposed.

We consider linear matrix equations of the form

$$AX + XB + \sum_{i=1}^{p} N_i X M_i - C = 0, \tag{5.0.1}$$

where $A, B, N_i, M_i, C \in \mathbb{R}^{n\times n}$ are large matrices and $p \ll n$. This equation is sometimes referred to as the *generalized Sylvester equation*, e.g., in [12], since it can be viewed as the sum of a Sylvester operator $\mathcal{L} : \mathbb{R}^{n\times n} \to \mathbb{R}^{n\times n}$, $\mathcal{L}(X) := AX + XB$, and a linear operator $\Pi : \mathbb{R}^{n\times n} \to \mathbb{R}^{n\times n}$, $\Pi(X) := \sum_{i=1}^{p} N_i X M_i$.

Linear matrix equations of the form (5.0.1) arise in different applications. In particular, in the following Section we show how the discrete problem stemming from the discretization by centered finite differences of the 2D and 3D convection-diffusion partial differential equation admits a representation in terms of (5.0.1). Other differential problems whose discretization leads to generalized linear matrix equations have been considered in the recent literature. For instance, in [90] the partial differential equation modeling a waveguide problem is considered, whereas elliptic PDEs with correlated random inputs are tackled in [88]. The special case of the *generalized Lyapunov equation*, which corresponds to equation (5.0.1) where $B = A^T$, $M_i = N_i$ and C symmetric, arises in model order reduction of bilinear and stochastic systems. See, e.g., [12, 31, 11] and references therein.

5.1 An example coming from PDEs

We consider the convection-diffusion partial differential equation

$$-\epsilon\Delta u + \mathbf{w}\cdot\nabla u = f, \qquad \text{in} \quad \Omega \subset \mathbb{R}^d, \tag{5.1.2}$$

with $d = 2, 3$, where $\mathbf{w}$ is the convection vector, while ϵ is the positive and constant viscosity parameter. In particular, we assume that the components of $\mathbf{w}$ are separable functions in

the space variables, and that Ω is a rectangle or a parallelepipedal domain. For simplicity the equation is equipped with Dirichlet boundary conditions; the analyzed procedures could be used with Neumann boundary conditions as well.

Standard finite difference or finite element discretizations yield the algebraic large non-symmetric linear system

$$A\mathbf{u} = \mathbf{f}. \qquad \text{with} \quad A \in \mathbb{R}^{N\times N}, \tag{5.1.3}$$

The algebraic problem (5.1.3) can be recast in terms of a multiterm linear matrix equation of the form (5.0.1). In [82], this reformulation has been used to develop a class of effective preconditioners for (5.1.3) that compare rather well with state-of-the-art and finely tuned algebraic multigrid preconditioners.

In the next Section we describe how the matrix equation form can be derived.

5.1.1 The two-dimensional case

For the ease of presentation, we shall first concentrate on the two-dimensional problem, that is equation (5.1.2) with $d = 2$, and then extend our derivation to the three-dimensional case in Section 5.1.3.

We start by recalling the matrix equation associated with the discretization by five-point stencil finite differences of the Poisson equation $-\Delta u = f$ on a rectangular domain $\Omega \subset \mathbb{R}^2$. For the sake of simplicity, we shall assume that $\Omega = (0,1)^2$. Let Ω_h be a uniform discretization of Ω, with nodes (x_i, y_j), $i,j = 1,\dots,n-1$. Then assuming homogeneous Dirichlet boundary conditions are used, centered finite difference discretization leads to the linear system (5.1.3) with

$$A = T_{n-1} \otimes I_{n-1} + I_{n-1} \otimes T_{n-1},$$

and $T_{n-1} = -1/h^2 \text{tridiag}(1, -2, 1) \in \mathbb{R}^{(n-1)\times(n-1)}$, h the mesh-size, is the symmetric tridiagonal matrix approximating the second-order derivative in one dimension, while the entries of $\mathbf{u}$ contain an approximation to u at the nodes, having used a lexicographic order of the entries.

We thus take a step back, and describe in detail the process leading to the Kronecker formulation, with the aim of deriving its matrix counterpart. This description will allow us to also include the boundary conditions in a systematic manner.

Let $\bar{\Omega}_h$ be a uniform discretization of the closed domain $\bar{\Omega}$, with equidistant points in each direction, (x_i, y_j), $i,j = 0,\dots,n$. Analogously, $U_{i,j} = U(x_i, y_j)$ is the value of the approximation U to u at the nodes. For each $i,j = 1,\dots,n-1$ we have the usual approximations

$$u_{xx}(x_i, y_j) \approx \frac{U_{i-1,j} - 2U_{i,j} + U_{i+1,j}}{h^2} = \frac{1}{h^2}[1,-2,1]\begin{bmatrix} U_{i-1,j} \\ U_{i,j} \\ U_{i+1,j} \end{bmatrix},$$

and analogously for the y direction, but from the right,

$$u_{yy}(x_i, y_j) \approx \frac{U_{i,j-1} - 2U_{i,j} + U_{i,j+1}}{h^2} = \frac{1}{h^2}[U_{i,j-1}, U_{i,j}, U_{i,j+1}]\begin{bmatrix} 1 \\ -2 \\ 1 \end{bmatrix}.$$

Let

$$T = -\frac{1}{h^2}\begin{pmatrix} * & * & & & & \\ * & -2 & 1 & & & \\ & 1 & \ddots & \ddots & & \\ & & \ddots & \ddots & 1 & \\ & & & 1 & -2 & * \\ & & & & * & * \end{pmatrix} \in \mathbb{R}^{(n+1)\times(n+1)}; \tag{5.1.4}$$

the unspecified values "*" are associated with boundary values of U and will be discussed in Section 5.1.2. Collecting these relations for all rows i's and for all columns j's, for the whole domain we obtain

$$-u_{xx} \approx TU, \qquad -u_{yy} \approx UT.$$

With these approximations we can write the following classical matrix form of the finite difference discretization of the Poisson equation on a square domain (see, e.g., [114])

$$TU + UT = F, \qquad \text{where} \quad (F)_{i,j} = f(x_i, y_j) + b.c.. \tag{5.1.5}$$

Except for the boundary conditions, the Kronecker formulation of (5.1.5) gives the same form as (5.1.3).

For the convection-diffusion equation with separable coefficients a similar derivation provides a generalized Sylvester matrix equation. We state the result in the following proposition, where separable convection coefficients are assumed. To this end, we define the matrix

$$B = \frac{1}{2h}\begin{bmatrix} * & * & & & & \\ * & 0 & 1 & & & \\ & -1 & \ddots & \ddots & & \\ & & \ddots & \ddots & 1 & \\ & & & -1 & 0 & * \\ & & & & * & * \end{bmatrix} \in \mathbb{R}^{(n+1)\times(n+1)}, \tag{5.1.6}$$

which represents the centered finite difference approximation of the first order one dimensional (1D) derivative on a uniformly discretized interval.

Proposition 5.1.1. *Assume that the convection vector* $\mathbf{w} = (w_1, w_2)$ *satisfies* $w_1 = \phi_1(x)\psi_1(y)$ *and* $w_2 = \phi_2(x)\psi_2(y)$. *Let* $(x_i, y_j) \in \bar{\Omega}_h$, $i, j = 0, \ldots, n$ *and set* $\Phi_k = diag(\phi_k(x_0), \ldots, \phi_k(x_n))$ *and* $\Psi_k = diag(\psi_k(y_0), \ldots, \psi_k(y_n))$, $k = 1, 2$. *Then with the previous notation, the centered finite-difference discretization of the differential operator in (5.1.2) leads to the following operator:*

$$\mathcal{L}_h : U \to \epsilon TU + \epsilon UT + (\Phi_1 B)U\Psi_1 + \Phi_2 U(B^T \Psi_2). \tag{5.1.7}$$

Proof. The first two terms of $\mathcal{L}_h(U)$ were derived for (5.1.5). We are left with showing that the first order term can be expressed in terms of the 1D discretization matrix B in (5.1.6). We have

$$\begin{aligned} \phi_1(x_i)\psi_1(y_j)u_x(x_i, y_j) &\approx \phi_1(x_i)\frac{u(x_{i+1}, y_j) - u(x_{i-1}, y_j)}{2h}\psi_1(y_j) \\ &= \frac{1}{2h}\phi_1(x_i)[-1, 0, 1]\begin{bmatrix} U_{i-1,j} \\ U_{i,j} \\ U_{i+1,j} \end{bmatrix}\psi_1(y_j), \end{aligned}$$

and analogously,

$$\phi_2(x_i)\psi_2(y_j)u_y(x_i, y_j) \approx \frac{1}{2h}\phi_2(x_i)[U_{i,j-1}, U_{i,j}, U_{i,j+1}]\begin{bmatrix} -1 \\ 0 \\ 1 \end{bmatrix}\psi_2(y_j).$$

Collecting these results for all grid nodes and recalling that $U_{i,j} = U(x_i, y_j)$, we obtain

$$(\phi_1(x_i)\psi_1(y_j)u_x(x_i, y_j))_{i,j=0,\ldots,n} \approx \Phi_1 BU\Psi_1,$$

$$(\phi_2(x_i)\psi_2(y_j)u_y(x_i, y_j))_{i,j=0,\ldots,n} \approx \Phi_2 UB^T\Psi_2,$$

and the result follows. □

5.1.2 Imposing the boundary conditions

The algebraic problem needs to be completed by imposing the boundary values. These will fill up the undefined entries in the coefficient matrices, and in the right-hand side matrix. To this end, we recall that with the given ordering of the elements in U, the first and last columns, Ue_1 and Ue_{n+1} respectively, correspond to the boundary sides $y = 0$ and $y = 1$, whereas the first and last rows take up the values at the boundary sides $x = 0$ and $x = 1$. With this notation, we wish to complete the corners of the matrices T and B, giving rise to the matrices T_1, T_2 and B_1, B_2, respectively, so that the following matrix equation is well defined for $(x_i, y_j) \in \bar{\Omega}_h$:

$$\epsilon T_1 U + \epsilon U T_2 + \Phi_1 B_1 U \Psi_1 + \Phi_2 U B_2 \Psi_2 = F. \tag{5.1.8}$$

With the same notation as for U, the entries of F corresponding to $i, j \in \{0, n\}$ will contain contributions from the boundary values of U, which are determined next.

For the boundary conditions to be satisfied, the operator $\mathcal{L}_h(U) = \epsilon T_1 U + \epsilon U T_2 + \Phi_1 B_1 U \Psi_1 + \Phi_2 U B_2 \Psi_2$ should act as the (scaled) identity operator for points at the boundary. To this end, from the generic matrix T we define the matrix T_1 as follows:

$$T_1 = -\frac{1}{h^2}\begin{pmatrix} -1 & 0 & & & & \\ 1 & -2 & 1 & & & \\ & 1 & -2 & 1 & & \\ & & \ddots & \ddots & \ddots & \\ & & & \ddots & \ddots & 1 \\ & & & & 0 & -1 \end{pmatrix} \in \mathbb{R}^{(n+1)\times(n+1)},$$

while the matrix corresponding to the first order operator (B in the generic case) can be written as

$$B_1 = \frac{1}{2h}\begin{pmatrix} 0 & 0 & & & & \\ -1 & 0 & 1 & & & \\ & -1 & 0 & 1 & & \\ & & \ddots & \ddots & \ddots & \\ & & & -1 & 0 & 1 \\ & & & & 0 & 0 \end{pmatrix} \in \mathbb{R}^{(n+1)\times(n+1)}.$$

For the derivative in the y direction, right multiplication should also act like the identity, therefore we can define $T_2 = T_1^T$ and correspondingly, $B_2 = B_1^T$. We are thus ready to define the missing entries in F so that (5.1.8) holds. For the first column, that is for the side $y = 0$, we write

$$\begin{aligned} Fe_1 &= (\epsilon T_1 U + \epsilon U T_2 + \Phi_1 B_1 U \Psi_1 + \Phi_2 U B_2 \Psi_2) e_1 \\ &= \epsilon T_1 U e_1 + \epsilon U T_2 e_1 + \Phi_1 B_1 U \Psi_1 e_1 + \Phi_2 U B_2 \Psi_2 e_1 \\ &= \epsilon T_1 U e_1 + \frac{\epsilon}{h^2} U e_1 + \Psi_1(y_0) \Phi_1 B_1 U e_1, \end{aligned}$$

where we used the fact that $B_2 \Psi_2 e_1 = 0$. Similar reasonings ensure that the boundary values at $y = 1$ are imposed, thus defining Fe_{n+1}. For the side $x = 0$ we have

$$\begin{aligned} e_1^T F &= e_1^T (\epsilon T_1 U + \epsilon U T_2 + \Phi_1 B_1 U \Psi_1 + \Phi_2 U B_2 \Psi_2) = e_1^T F \\ &= e_1^T \epsilon T_1 U + e_1^T \epsilon U T_2 + e_1^T \Phi_1 B_1 U \Psi_1 + e_1^T \Phi_2 U B_2 \Psi_2 \\ &= \frac{\epsilon}{h^2} e_1^T U + \epsilon e_1^T U T_2 + \phi_2(x_0) e_1^T U B_2 \Psi_2, \end{aligned}$$

where the fact that $e_1^T \Phi_1 B_1 = 0$ was used. The definition of $e_{n+1}^T F$ follows analogously.

5.1.3 The three-dimensional case

The 3D convection-diffusion equation can be stated as in (5.1.2), for $\Omega \subset \mathbb{R}^3$. To convey our idea, we again first focus on the Poisson equation, and then generalize the matrix formulation to the finite difference discretization of the non-self-adjoint problem (5.1.2). For the sake of simplicity, we shall assume that $\Omega = (0,1)^3$, though more general parallelepipedal domains could also be considered. We discretize $\bar{\Omega}$ with equidistant nodes in each direction, (x_i, y_j, z_k), for $i, j, k = 0, \ldots, n$. To fix the ideas, let $U_{i,j}^{(k)} = U(x_i, y_j, z_k)$ denote the value of the approximation U to u at the node (x_i, y_j, z_k) (other orderings may be more convenient depending on the equation properties). We also define the tall matrix

$$\mathcal{U} = \begin{bmatrix} U^{(0)} \\ \vdots \\ U^{(n)} \end{bmatrix} = \sum_{k=0}^{n} (e_{k+1} \otimes U^{(k)}) \in \mathbb{R}^{(n+1)^2 \times (n+1)}.$$

Let T be as defined in (5.1.4). Then, for $I \in \mathbb{R}^{(n+1)\times(n+1)}$ the identity matrix,

$$\begin{aligned} -u_{xx} &\approx \sum_{k=0}^{n} (e_{k+1} \otimes TU^{(k)}) = (I \otimes T) \sum_{k=0}^{n} (e_{k+1} \otimes U^{(k)}) = (I \otimes T)\mathcal{U}, \\ -u_{yy} &\approx \sum_{k=0}^{n} (e_{k+1} \otimes U^{(k)}T) = \sum_{k=0}^{n} (e_{k+1} \otimes U^{(k)})T = \mathcal{U}T, \\ -u_{zz} &\approx (T \otimes I)\mathcal{U}. \end{aligned}$$

With these approximations we can thus obtain the following matrix form of the finite difference discretization of the Poisson equation:

$$(I \otimes T)\mathcal{U} + \mathcal{U}T + (T \otimes I)\mathcal{U} = F, \tag{5.1.9}$$

where $F = \sum_{k=0}^{n} (e_{k+1} \otimes F^{(k)}) \in \mathbb{R}^{(n+1)^2 \times (n+1)}$ and $(F^{(k)})_{i,j} = f(x_i, y_j, z_k)$. The Kronecker formulation of the matrix equation (5.1.9) determines the usual approximation of the Laplacian operator by seven-point stencil finite differences,

$$\Delta \approx I \otimes I \otimes T + I \otimes T \otimes I + T \otimes I \otimes I \in \mathbb{R}^{(n+1)^3 \times (n+1)^3}.$$

For the convection-diffusion equation with separable coefficients a similar derivation provides a generalized Sylvester matrix equation. We state the result in the following proposition.

Proposition 5.1.2. *Assume that the convection vector* $\mathbf{w} = (w_1, w_2, w_3)$ *satisfies* $w_1 = \phi_1(x)\psi_1(y)\upsilon_1(z)$, $w_2 = \phi_2(x)\psi_2(y)\upsilon_2(z)$, *and* $w_3 = \phi_3(x)\psi_3(y)\upsilon_3(z)$. *Let* (x_i, y_j, z_k), $i, j, k = 0, \ldots, n$ *be the grid nodes discretizing* $\bar{\Omega}$ *with mesh size* h, *and set* $\Phi_\ell = diag(\phi_\ell(x_0), \ldots, \phi_\ell(x_n))$, $\Psi_\ell = diag(\psi_\ell(y_0), \ldots, \psi_\ell(y_n))$, *and* $\Upsilon_\ell = diag(\upsilon_\ell(z_0), \ldots, \upsilon_\ell(z_n))$, $\ell = 1, 2, 3$. *Then, with* B *as defined in (5.1.6), the centered finite-difference discretization of the differential operator in (5.1.2) leads to the following operator:*

$$\begin{aligned} \mathcal{L}_h : U \quad \to \quad & (I \otimes \epsilon T)\mathcal{U} + \epsilon \mathcal{U}T + (\epsilon T \otimes I)\mathcal{U} + \\ & (\Upsilon_1 \otimes \Phi_1 B)\mathcal{U}\Psi_1 + (\Upsilon_2 \otimes \Phi_2)\mathcal{U}B^T\Psi_2 + [(\Upsilon_3 B) \otimes \Phi_3]\mathcal{U}\Psi_3. \end{aligned} \tag{5.1.10}$$

Proof. The second order terms of $\mathcal{L}_h(U)$ correspond to a multiple of (5.1.9). We are thus left with showing that the first order term can be expressed by means of the 1D discretization

matrix B. We first fix $k = \bar{k}$ and we have

$$\begin{aligned}\phi_1(x_i)\psi_1(y_j)\upsilon_1(z_{\bar{k}})u_x(x_i,y_j,z_{\bar{k}}) &\approx \upsilon_1(z_{\bar{k}})\phi_1(x_i)\frac{u(x_{i+1},y_j,z_{\bar{k}})-u(x_{i-1},y_j,z_{\bar{k}})}{2h}\psi_1(y_j)\\ &= \frac{1}{2h}\upsilon_1(z_{\bar{k}})\phi_1(x_i)[-1,0,1]\begin{bmatrix}U^{(\bar{k})}_{i-1,j}\\ U^{(\bar{k})}_{i,j}\\ U^{(\bar{k})}_{i+1,j}\end{bmatrix}\psi_1(y_j).\end{aligned}$$

Analogously,

$$\phi_2(x_i)\psi_2(y_j)\upsilon_2(z_{\bar{k}})u_y(x_i,y_j,z_{\bar{k}}) \approx \frac{1}{2h}\upsilon_2(z_{\bar{k}})\phi_2(x_i)[U^{(\bar{k})}_{i,j-1},U^{(\bar{k})}_{i,j},U^{(\bar{k})}_{i,j+1}]\begin{bmatrix}-1\\0\\1\end{bmatrix}\psi_2(y_j).$$

Collecting these results for all grid nodes $(x_i,y_j,z_{\bar{k}})_{i,j=0,\dots,n}$ and recalling that $U^{(k)}_{i,j} = U(x_i,y_j,z_k)$, we obtain

$$(\phi_1(x_i)\psi_1(y_j)\upsilon_1(z_{\bar{k}})u_x(x_i,y_j,z_{\bar{k}}))_{i,j=0,\dots,n} \approx \upsilon_1(z_{\bar{k}})\Phi_1 B U^{(\bar{k})}\Psi_1,$$

and

$$(\phi_2(x_i)\psi_2(y_j)\upsilon_2(z_{\bar{k}})u_y(x_i,y_j,z_{\bar{k}}))_{i,j=0,\dots,n} \approx \upsilon_2(z_{\bar{k}})\Phi_2 U^{(\bar{k})}B^T\Psi_2.$$

Therefore, for all z nodes,

$$\begin{aligned}&(\phi_1(x_i)\psi_1(y_j)\upsilon_1(z_k)u_x(x_i,y_j,z_k))_{i,j,k=0,\dots,n}\\ &\quad\approx [\Upsilon_1\otimes I]\sum_{k=0}^{n}(e_{k+1}\otimes\Phi_1 B U^{(k)}\Psi_1) = [\Upsilon_1\otimes I](I\otimes\Phi_1 B)[\sum_{k=0}^{n}(e_{k+1}\otimes U^{(k)})]\Psi_1\\ &\quad= [\Upsilon_1\otimes I](I\otimes\Phi_1 B)\mathcal{U}\Psi_1 = (\Upsilon_1\otimes\Phi_1 B)\mathcal{U}\Psi_1,\end{aligned}$$

and

$$\begin{aligned}&(\phi_2(x_i)\psi_2(y_j)\upsilon_2(z_z)u_y(x_i,y_j,z_k))_{i,j,k=0,\dots,n}\\ &\quad\approx [\Upsilon_2\otimes I]\sum_{k=0}^{n}(e_{k+1}\otimes\Phi_2 U^{(k)}B^T\Psi_2) = [\Upsilon_2\otimes I](I\otimes\Phi_2)[\sum_{k=0}^{n}(e_{k+1}\otimes U^{(k)})]B^T\Psi_2\\ &\quad= [\Upsilon_2\otimes I](I\otimes\Phi_2)\mathcal{U}B^T\Psi_2 = (\Upsilon_2\otimes\Phi_2)\mathcal{U}B^T\Psi_2.\end{aligned}$$

On the other hand, for the z direction it holds

$$\begin{aligned}\phi_3(x_i)\psi_3(y_j)\upsilon_3(z_k)u_z(x_i,y_j,z_k) &\approx \upsilon_3(z_k)\phi_3(x_i)\frac{u(x_i,y_j,z_{k+1})-u(x_i,y_j,z_{k-1})}{2h}\\ &\approx \frac{1}{2h}\upsilon_3(z_k)\phi_3(x_i)[-1,0,1]\begin{bmatrix}U^{(k-1)}_{i,j}\\ U^{(k)}_{i,j}\\ U^{(k+1)}_{i,j}\end{bmatrix}\psi_3(y_j).\end{aligned}$$

Collecting this relation for all blocks,

$$\begin{aligned}&(\phi_3(x_i)\psi_3(y_j)\upsilon_3(z_k)u_z(x_i,y_j,z_k))_{i,j,k=0,\dots,n}\\ &\quad\approx (\Upsilon_3 B\otimes I)\sum_{k=0}^{n}[e_{k+1}\otimes(\Phi_3 U^{(k)}\Psi_3)] = (\Upsilon_3 B\otimes I)(I\otimes\Phi_3)[\sum_{k=0}^{n}(e_{k+1}\otimes U^{(k)})]\Psi_3\\ &\quad= (\Upsilon_3 B\otimes I)(I\otimes\Phi_3)\mathcal{U}\Psi_3 = [(\Upsilon_3 B)\otimes\Phi_3]\mathcal{U}\Psi_3.\end{aligned}$$

and the result follows. □

Imposing the boundary conditions completely determines the entries of T in all three instances, as well as the missing entries in B. Following the same steps as for the 2D case, the matrix equation (5.1.10) can be written as

$$((I \otimes \epsilon T_1) + (\epsilon T_2^T \otimes I))\, \mathcal{U} + \epsilon \mathcal{U}\, T_3 + (\Upsilon_1 \otimes \Phi_1 B_1)\, \mathcal{U}\, \Psi_1 + (\Upsilon_2 \otimes \Phi_2)\, \mathcal{U}\, B_3 \Psi_2 + [(\Upsilon_3 B_2^T) \otimes \Phi_3]\, \mathcal{U}\, \Psi_3 = F,$$

highlighting the presence of five distinct terms in the matrix equation. With this ordering of the variables, it holds that $B_3 = B_2$ and $T_3 = T_2$.

5.2 Solution in closed-form

It is not easy to derive necessary and sufficient conditions for the well-posedness of equation (5.0.1) in terms of the coefficient matrices A, B, N_i, M_i. If one wants to mimic the analysis carried out in Section 1.1 for standard Sylvester and Lyapunov equations, the linear system

$$\mathcal{A}\mathrm{vec}(X) = \mathrm{vec}(C), \quad \mathcal{A} := B^T \otimes I + I \otimes A + \sum_{i=1}^{p} M_i^T \otimes N_i, \tag{5.2.11}$$

has to be considered. Although its Kronecker form, the spectrum of the coefficient matrix $\mathcal{A}$ cannot be characterized in terms of $\Lambda(A)$, $\Lambda(B)$, $\Lambda(N_i)$, and $\Lambda(M_i)$. However, we can say that the generalized Sylvester equation (5.0.1) has a unique solution X for every $C \neq 0$ if and only if $\mathcal{A}$ is nonsingular.

Easier to handle sufficient conditions for the well-posedness of equation (5.0.1) can be deduced if further assumptions on the coefficient matrices are considered. In these cases, closed-forms of X can also be derived.

Theorem 5.2.1. *Consider equation (5.0.1) and suppose that $\Lambda(A) \cap \Lambda(-B) = \varnothing$ and that the matrices N_i, M_i are low-rank for all i, namely $N_i = P_i \widetilde{P}_i^T$, $P, \widetilde{P}_i \in \mathbb{R}^{n \times s_i}$ and $M_i = Q_i \widetilde{Q}_i^T$, $Q, \widetilde{Q}_i \in \mathbb{R}^{n \times t_i}$, $s_i, t_i \ll n$ for all $i = 1, \ldots, p$. Let $L := B^T \otimes I + I \otimes A \in \mathbb{R}^{n^2 \times n^2}$, $U := [\widetilde{Q}_1 \otimes P_1, \ldots, \widetilde{Q}_p \otimes P_p]$, $V := [Q_1 \otimes \widetilde{P}_1, \ldots, Q_p \otimes \widetilde{P}_p] \in \mathbb{R}^{n^2 \times q}$, $q := \sum_{i=1}^{p} s_i t_i$. Then, if the matrix $W = I_q + V^T L^{-1} U$ is nonsingular, there exists a unique solution X to (5.0.1) and it is such that*

$$vec(X) = L^{-1} vec(C) - U W^{-1} V^T L^{-1} vec(C). \tag{5.2.12}$$

Proof. The result follows by applying the Sherman-Morrison-Woodbury formula to the linear system (5.2.11) noticing that in this case it holds $\mathcal{A} = L + UV^T$ and that the nonsingularity of L comes from the assumption $\Lambda(A) \cap \Lambda(-B) = \varnothing$. □

A similar result has been derived in [90] whereas in [11] Benner and Breiten used the Sherman-Morrison-Woodbury argument to show that the solution of a generalized Lyapunov equation with low-rank N_i and C is expected to be low-rank. Moreover, in case of problems of moderate dimensions, efficient numerical procedures for computing X by (5.2.12) have been proposed in, e.g., [31, 89, 90].

A different framework where a closed-form solution can be derived is outlined in the following Theorem.

Theorem 5.2.2. *Let $\mathcal{L}, \Pi : \mathbb{R}^{n \times n} \to \mathbb{R}^{n \times n}$ be linear operators such that $\mathcal{L}$ is invertible, $\rho(\mathcal{L}^{-1}\Pi) < 1$ and let $C \in \mathbb{R}^{n \times n}$. The unique solution of the equation $\mathcal{L}(X) + \Pi(X) - C = 0$ can be represented as*

$$X = \sum_{j=0}^{\infty} Z_j, \tag{5.2.13}$$

where

$$\begin{cases} Z_0 & := \mathcal{L}^{-1}(C), \\ Z_{j+1} & := -\mathcal{L}^{-1}(\Pi(Z_j)), \qquad j \geq 0. \end{cases} \tag{5.2.14}$$

Proof. By using the invertibility of $\mathcal{L}$ we have $X = (I + \mathcal{L}^{-1}\Pi)^{-1}\mathcal{L}^{-1}(C)$ and with the assumption $\rho(\mathcal{L}^{-1}\Pi) < 1$ we can express the operator $(I + \mathcal{L}^{-1}\Pi)^{-1}$ as a convergent Neumann series (see, e.g., in [68, Example 4.5]). In particular, we obtain

$$X = \sum_{j=0}^{\infty} (-1)^j \left(\mathcal{L}^{-1}\Pi\right)^j \mathcal{L}^{-1}(C).$$

The relation (5.2.13) follows by defining $Z_j := (-1)^j \left(\mathcal{L}^{-1}\Pi\right)^j \mathcal{L}^{-1}(C)$. □

Theorem 5.2.2 can be used for any choice of linear operators $\mathcal{L}$ and Π. However, it gives constructive insights when $\mathcal{L}$ is easy to invert so that an approximation to the solution of $\mathcal{L}(X) + \Pi(X) - C = 0$ can be constructed by truncating the series (5.2.13). In particular, let

$$X^{(\ell)} := \sum_{j=0}^{\ell} Z_j, \tag{5.2.15}$$

where Z_j are given by (5.2.14), then the truncation error can be bounded as follows

$$\|X - X^{(\ell)}\| \leq \|\mathcal{L}^{-1}(C)\| \frac{\rho(\mathcal{L}^{-1}\Pi)^{\ell+1}}{1 - \rho(\mathcal{L}^{-1}\Pi)}. \tag{5.2.16}$$

In our case, $\mathcal{L}(X) = AX + XB$ and $\Pi(X) = \sum_{i=1}^{p} N_i X M_i$ and the truncated Neumann series (5.2.15) thus requires the solution of $\ell + 1$ (standard) Sylvester equations. In the next Section we derive a numerical procedure that efficiently computes an approximation to X by exploiting (5.2.15) in case of small-scale problems.

5.2.1 Solving small-scale generalized Sylvester equations

In case of problems of moderate dimension, say $n = \mathcal{O}(10^3)$, the Neumann series expansion derived in Theorem 5.2.2 can be exploited to design an efficient numerical procedure. Indeed, the computation of $X^{(\ell)}$ in (5.2.15) requires the solution of $\ell + 1$ Sylvester equations (5.2.14) which can be simultaneously reduced to triangular form (c.f., [89, Section 3]) as they are defined by the same coefficient matrices. In particular, let $A = Q_A U_A Q_A^T$ and $B^T = Q_B U_B Q_B^T$ denote the Schur decompositions of A and B^T respectively. Pre and post-multiplying the equations in (5.2.14) by Q_A^T and Q_B respectively, we obtain

$$U_A \widetilde{Z}_0 + \widetilde{Z}_0 U_B^T - \widetilde{C} = 0, \tag{5.2.17a}$$

$$U_A \widetilde{Z}_{j+1} + \widetilde{Z}_{j+1} U_B^T + \sum_{i=1}^{p} \widetilde{N}_i \widetilde{Z}_j \widetilde{M}_i = 0, \qquad j = 0, \ldots, \ell - 1, \tag{5.2.17b}$$

where we have defined

$$\widetilde{C} := Q_A^T C Q_B, \qquad \widetilde{N}_i := Q_A^T N_i Q_A, \qquad \widetilde{M}_i := Q_B^T M_i Q_B, \qquad \widetilde{Z}_i := Q_A^T Z_i Q_B. \tag{5.2.18}$$

The Sylvester equations with triangular coefficients (5.2.17) can be efficiently solved by backward substitution as in the Bartels-Stewart algorithm [5] and it holds $X^{(\ell)} = Q_A \left(\sum_{j=0}^{\ell} \widetilde{Z}_j\right) Q_B^T$. In the next Proposition we show how the Frobenius norm of the residual $\mathcal{R}^{(\ell)} := AX^{(\ell)} + X^{(\ell)}B + \sum_{i=1}^{p} N_i X^{(\ell)} M_i - C$ can be cheaply computed without explicitly constructing $X^{(\ell)}$.

Algorithm 5.1: Neumann series approach for (5.0.1).

input : Matrix coefficients: $A, B, N_1 \dots, N_p, M_1, \dots, M_p, C$
output: Truncated Neumann series $X^{(\ell)}$

Compute the Schur decompositions $A = Q_A U_A Q_A^T$, $B = Q_B U_B Q_B^T$
Compute $\widetilde{C}$, $\widetilde{N}_i$ $\widetilde{M}_i$ for all $i = 1, \dots, m$ according to (5.2.18)
Solve $U_A \widetilde{Z}_0 + \widetilde{Z}_0 U_B^T - \widetilde{C} = 0$ and set $\widetilde{X} = \widetilde{Z}_0$
for $j = 0, 1, \dots$ *till convergence* **do**
    Solve $U_A \widetilde{Z}_{j+1} + \widetilde{Z}_{j+1} U_B^T + \sum_{i=1}^p \widetilde{N}_i \widetilde{Z}_j \widetilde{M}_i^T = 0$ and set $\widetilde{X} = \widetilde{X} + \widetilde{Z}_{j+1}$
    Compute $\|\mathcal{R}^{(j+1)}\|_F = \| \sum_{i=1}^p \widetilde{N}_i \widetilde{Z}_{j+1} \widetilde{M}_i^T \|_F$
    if $\|\mathcal{R}^{(j+1)}\|_F \leq$ `tol` **then**
        Set $\ell = j + 1$
        Break
    end
end
Return $X^{(\ell)} = Q_A \widetilde{X} Q_B^T$

Proposition 5.2.3. *Let $X^{(\ell)}$ be as defined in (5.2.15). Then the residual matrix $\mathcal{R}^{(\ell)} := AX^{(\ell)} + X^{(\ell)}B + \sum_{i=1}^p N_i X^{(\ell)} M_i - C$ is such that*

$$\left\|\mathcal{R}^{(\ell)}\right\|_F = \left\| \sum_{i=1}^p \widetilde{N}_i \widetilde{Z}_\ell \widetilde{M}_i \right\|_F . \tag{5.2.19}$$

Proof. It holds

$$\begin{aligned}
\left\|\mathcal{R}^{(\ell)}\right\|_F &= \left\| AX^{(\ell)} + X^{(\ell)}B + \sum_{i=1}^p N_i X^{(\ell)} M_i - C \right\|_F = \left\| A \sum_{j=0}^{\ell} Z_j + \sum_{j=0}^{\ell} Z_j B + \sum_{i=1}^p N_i \sum_{j=0}^{\ell} Z_j M_i - C \right\|_F \\
&= \left\| \underbrace{AZ_0 + Z_0 B - C}_{=0} + A \sum_{j=1}^{\ell} Z_j + \sum_{j=1}^{\ell} Z_j B + \sum_{i=1}^p N_i \sum_{j=0}^{\ell} Z_j M_i \right\|_F = \\
&= \left\| \underbrace{AZ_1 + Z_1 B + \sum_{i=1}^p N_i Z_0 M_i}_{=0} + A \sum_{j=2}^{\ell} Z_j + \sum_{j=2}^{\ell} Z_j B + \sum_{i=1}^p N_i \sum_{j=1}^{\ell} Z_j M_i \right\|_F = \dots = \left\| \sum_{i=1}^p N_i Z_\ell M_i \right\|_F \\
&= \left\| Q_A \left(\sum_{i=1}^p N_i Q_A^T Q_A Z_\ell Q_B^T Q_B M_i \right) Q_B^T \right\|_F = \left\| \sum_{i=1}^p \widetilde{N}_i \widetilde{Z}_\ell \widetilde{M}_i \right\|_F .
\end{aligned}$$

□

In conclusion, the following iterative procedure can be used to approximate the solution to (5.0.1): the matrices (5.2.18) are precomputed, then the Sylvester equations in triangular form (5.2.17) are solved until the residual of the Neumann series (5.2.19) is sufficiently small. The final approximation $X^{(\ell)}$ is constructed only after the iteration has completed. The procedure is summarized in Algorithm 14.

Chapter 6

Generalized Sylvester and Lyapunov equations with low-rank right-hand side

In this Chapter we focus on computational strategies to solve large-scale generalized Sylvester equations with low-rank right-hand side

$$\mathcal{L}(X) + \Pi(X) - C_1 C_2^T = 0, \qquad C_1, C_2 \in \mathbb{R}^{n\times r},\ r \ll n. \tag{6.0.1}$$

We assume $\rho(\mathcal{L}^{-1}\Pi) < 1$ so that a unique solution X to (6.0.1) exists. See Section 5.2.

From a computational point of view, assuming $\mathcal{L}^{-1}\Pi$ to be a contraction allows us to employ solution processes based on a regular splitting of the overall operator $\mathcal{L} + \Pi$, that is a numerical solution X_k can be computed by the iterative procedure

$$X_{k+1} = -\mathcal{L}^{-1}(\Pi(X_k)) + \mathcal{L}^{-1}(C_1 C_2^T). \tag{6.0.2}$$

It easy to show how X_{k+1} computed by (6.0.2) is equivalent to the truncated Neumann series $X^{(\ell)}$ in (5.2.15).

The procedure in (6.0.2) is numerically inadequate in case of large-scale problems as it provides a dense matrix X_{k+1}. Therefore memory-saving strategies have to be devised also in this setting. Procedures belonging to the class of low-rank methods have been developed. In particular, in [11], the authors propose a bilinear ADI (BilADI) method which naturally extends the low-rank ADI algorithm for standard Lyapunov problems to generalized Lyapunov equations. A nonstationary iterative method based on (6.0.2) is derived in [96]. In particular, the inversion of $\mathcal{L}$ that corresponds to the solution of a large-scale (standard) Sylvester equation is computed by K-PIK which efficiently implements an extended Krylov subspace method. See Chapter 2 and [97, 26]. The projection method is coupled with a low-rank truncation of the current iterate X_{k+1}. In [70] a greedy low-rank technique for general equations of the form $\sum_{i=1}^{p} A_i X B_i - C_1 C_2^T = 0$ is presented.

The application of these methods has been often heuristically motivated. Indeed, the existence of a low-rank approximant $\widetilde{X}$ to X has been shown only in case of a low-rank Π by applying a Sherman-Morrison-Woodbury argument. See [11]. In the next Section , assuming that $\mathcal{L}^{-1}(C_1 C_2^T)$ can be well approximated by a low-rank matrix, we show that $\rho(\mathcal{L}^{-1}\Pi) < 1$ is a sufficient condition for the existence of a low-rank numerical solution $\widetilde{X}$ to (6.0.1).

6.1 Low-rank approximability

The closed-form of X derived in Theorem 5.2.2 can be exploited to show that the solution to (6.0.1) can be often approximated by a low-rank matrix. We base our reasonings on the low-rank approximability properties of the Sylvester operator $\mathcal{L}$ presented at the beginning of Chapter 2. Since the solution X to (6.0.1) can be obtained summing the solutions of equations (5.2.14), we can extend several results concerning the low-rank approximability of the solution to (standard) Sylvester equations to the case of generalized Sylvester equations assuming $\rho(\mathcal{L}^{-1}\Pi) < 1$. More precisely, under the low-rank approximability assumption of $\mathcal{L}$, the right-hand side of the Sylvester equations (5.2.14) is a low-rank matrix for every j since $C_1C_2^T$ is low-rank and $p \ll n$.

Theorem 6.1.1. *Let $\mathcal{L}$ be the Sylvester operator $\mathcal{L}(X) = AX + XB$, Π the linear operator $\Pi(X) = \sum_{i=1}^{p} N_i X M_i$, $C_1, C_2 \in \mathbb{R}^{n\times r}$ and k a positive integer. Let $X^{(\ell)}$ be the truncated Neumann series (5.2.15). Then there exists a matrix $\overline{X}^{(\ell)}$ such that*

$$\operatorname{rank}(\overline{X}^{(\ell)}) \le (2k+1)r + \sum_{j=1}^{\ell} (2k+1)^{j+1} p^j r, \tag{6.1.3}$$

and

$$\left\| X^{(\ell)} - \overline{X}^{(\ell)} \right\| \le \overline{K} e^{-\pi\sqrt{k}}, \tag{6.1.4}$$

where $\overline{K}$ is a constant that does not depend on k and only depends on $\mathcal{L}$ and ℓ.

Proof. For a given k, let $\mathcal{L}_k$ be such that $\mathcal{L}_k^{-1}(C) = \sum_{j=-k}^{k} \omega_j e^{t_j A} C e^{t_j B^T}$ so that $\|\mathcal{L}^{-1} - \mathcal{L}_k^{-1}\| \le Ke^{-\pi\sqrt{k}}$ as in (2.0.3). We then consider the sequence

$$\begin{cases} \overline{Z}_0 & := \mathcal{L}_k^{-1}(C_1C_2^T), \\ \overline{Z}_{j+1} & := -\mathcal{L}_k^{-1}(\Pi(\overline{Z}_j)), \quad j \ge 0. \end{cases} \tag{6.1.5}$$

Defining $\beta := \|\mathcal{L}^{-1}\Pi\|$ and $\beta_k := \|\mathcal{L}_k^{-1}\Pi\|$, we have

$$\begin{aligned} \|Z_{j+1} - \overline{Z}_{j+1}\| &\le \|\mathcal{L}^{-1}(\Pi(Z_j)) - \mathcal{L}^{-1}(\Pi(\overline{Z}_j))\| + \|\mathcal{L}^{-1}(\Pi(\overline{Z}_j)) - \mathcal{L}_k^{-1}(\Pi(\overline{Z}_j))\| \\ &\le \beta\|Z_j - \overline{Z}_j\| + Ke^{-\pi\sqrt{k}}\|\Pi\|\|\overline{Z}_j\|. \end{aligned}$$

From the above expression, a simple recursive argument shows that

$$\|Z_{j+1} - \overline{Z}_{j+1}\| \le \beta^{j+1}\|Z_0 - \overline{Z}_0\| + Ke^{-\pi\sqrt{k}}\|\Pi\| \sum_{t=0}^{j} \beta^{j-t}\|\overline{Z}_t\|. \tag{6.1.6}$$

Using the submultiplicativity of the operator norm, it holds that $\|\overline{Z}_j\| = \|\mathcal{L}_k^{-1}(\Pi(\overline{Z}_{j-1}))\| \le \beta_k\|\overline{Z}_{j-1}\|$. In particular $\|\overline{Z}_j\| \le \beta_k^j\|\mathcal{L}_k^{-1}\|\|C_1C_2^T\|$, and therefore, from (6.1.6) it follows that

$$\begin{aligned} \|Z_{j+1} - \overline{Z}_{j+1}\| &\le \beta^{j+1}\|\mathcal{L}^{-1} - \mathcal{L}_k^{-1}\|\|C_1C_2^T\| + Ke^{-\pi\sqrt{k}}\|\Pi\| \sum_{t=0}^{j} \beta^{j-t}\|\mathcal{L}_k^{-1}\|\|C_1C_2^T\| \\ &\le \left[\beta^{j+1} + \|\Pi\|\|\mathcal{L}_k^{-1}\| \sum_{t=0}^{j} \beta^{j-t}\beta_k^t\right] Ke^{-\pi\sqrt{k}}\|C_1C_2^T\|. \end{aligned} \tag{6.1.7}$$

Since $\mathcal{L}_k^{-1}$ converges to $\mathcal{L}^{-1}$, and by using the continuity of the operators, we have that $\|\mathcal{L}_k^{-1}\|$ and β_k are bounded by a constant independent of k. Therefore from (6.1.7) it follows that there exists a constant K_{j+1} independent of k such that $\|Z_{j+1} - \overline{Z}_{j+1}\| \leq K_{j+1} e^{-\pi\sqrt{k}}$. The relation (6.1.4) follows by defining $\overline{X}^{(\ell)} := \sum_{j=0}^{\ell} \overline{Z}_j$ and observing

$$\|X^{(\ell)} - \overline{X}^{(\ell)}\| \leq \sum_{j=0}^{\ell} \|Z_j - \overline{Z}_j\| \leq e^{-\pi\sqrt{k}} \sum_{j=0}^{\ell} K_j = \overline{K} e^{-\pi\sqrt{k}},$$

where $\overline{K} := \sum_{j=0}^{\ell} K_j$.

The upper-bound (6.1.3) follows by induction observing that $\operatorname{rank}(\overline{Z}_{j+1}) \leq (2k+1)p \cdot \operatorname{rank}(\overline{Z}_j)$. □

Notice that the right-hand side in (6.1.3) does not grow linearly with neither the number of terms p defining the operator Π nor the number ℓ of terms considered in the truncated Neumann series (5.2.15). Therefore, (6.1.3) is meaningful only in case of equations with a right-hand side of small rank r, p moderate and such that few terms in (5.2.15) are necessary to obtain an accurate numerical solution, that is $\rho(\mathcal{L}^{-1}\Pi) \ll 1$. In these cases the truncated Neumann series $X^{(\ell)}$ allows for a low-rank approximation and the employment of low-rank methods in the solution of (6.0.1) is motivated as

$$\|X - \overline{X}^{(\ell)}\| \leq \|X - X^{(\ell)}\| + \|X^{(\ell)} - \overline{X}^{(\ell)}\| \leq \|\mathcal{L}^{-1}(C_1 C_2^T)\| \frac{\rho(\mathcal{L}^{-1}\Pi)^{\ell+1}}{1 - \rho(\mathcal{L}^{-1}\Pi)} + \overline{K} e^{-\pi\sqrt{k}}.$$

See also [81] for further results on the low-rank approximability of X.

6.2 Generalized Sylvester equations with low-rank commuting coefficients

We now consider equation (6.0.1) with $\rho(\mathcal{L}^{-1}\Pi) < 1$ and equipped with a further assumption that may seem unusual at a first glance. In particular, we suppose that the commutators of the matrix coefficients of the operator Π and the coefficients defining the Sylvester operator $\mathcal{L}$ are low-rank. More precisely, if $\{A, B\} := AB - BA$ denotes the commutator of two matrices, we assume that there exist $P_i, \widetilde{P}_i \in \mathbb{R}^{n \times s_i}$ and $Q_i, \widetilde{Q}_i \in \mathbb{R}^{n \times t_i}$ such that $s_i, t_i \ll n$ and the commutators fulfill

$$\{A, N_i\} = AN_i - N_i A = P_i \widetilde{P}_i^T, \tag{6.2.8a}$$
$$\{B, M_i\} = BM_i - M_i B = Q_i \widetilde{Q}_i^T, \tag{6.2.8b}$$

for $i = 1, \ldots, p$. The property (6.2.8), which we refer to as *low-rank commutation*, is in this framework a generalization of the concept of commuting matrices. The case of pure commutation, which occurs for instance when $N_i = f_i(A), M_i = g_i(B)$ with f_i, g_i polynomials or analytic functions, is analysed in, e.g., [72, 11]. Also in case of a low-rank Π [11], equations (6.2.8) are satisfied. Moreover, many discretization procedures lead to structured matrices (e.g., Toeplitz, circulant, etc...) that commute up to low-rank factors [9].

The framework of projection methods presented in Section 2.1 can be easily adapted to solve generalized Sylvester equations. Nevertheless, effective solvers based on projection are still lacking in the literature [11] and methods based on a regular splitting of $\mathcal{L} + \Pi$ are more commonly used. See Section 6.1. In the next Sections, we propose a projection

method where the projection spaces are selected by identifying certain features of the solution to (6.0.1) based on our characterization of X by a Neumann series expansion and the low-rank commutation properties (6.2.8). More precisely we use extended Krylov subspaces with an appropriate choice of the starting blocks.

To the best of our knowledge, exploiting the low-rank commutativity properties (6.2.8) was first proposed in [64] in the context of numerical methods for matrix equations.

6.2.1 Projection methods for generalized equations

As already mentioned, the framework of projection methods presented in Section 2.1 can be easily adapted to handle generalized Sylvester equations with a low-rank right-hand side (6.0.1). In particular, two sequences of nested subspaces of $\mathbb{R}^n$, i.e., $\mathcal{K}_{m-1} \subset \mathcal{K}_m$ and $\mathcal{C}_{m-1} \subset \mathcal{C}_m$, are computed. Motivated by the low-rank approximability of the solution illustrated in Theorem 6.1.1, projection methods construct approximations to X of the form

$$X_m = V_m Y_m U_m^T, \tag{6.2.9}$$

where V_m and U_m are matrices with orthonormal columns representing respectively an orthonormal basis of $\mathcal{K}_m$ and $\mathcal{C}_m$.

As in the standard case, the matrix Y_m can be obtained by imposing the Galerkin orthogonality condition, namely the residual

$$\mathcal{R}_m := AX_m + X_m B + \sum_{i=1}^{p} N_i X_m M_i - C_1 C_2^T, \tag{6.2.10}$$

is such that $V_m^T \mathcal{R}_m U_m = 0$. This condition is equivalent to Y_m satisfying the following small and dense generalized Sylvester equation

$$T_m Y_m + Y_m J_m^T + \sum_{i=1}^{p} G_{m,i} Y_m F_{m,i} - D_{m,1} D_{m,2}^T = 0, \tag{6.2.11}$$

where,

$$T_m := V_m^T A V_m, \qquad J_m := U_m^T B^T U_m, \qquad D_{m,1} = V_m^T C_1, \qquad D_{m,2} = U_m^T C_2, \tag{6.2.12a}$$

$$G_{m,i} := V_m^T N_i V_m, \qquad F_{m,i} := U_m^T M_i U_m, \qquad i = 1, \dots, p. \tag{6.2.12b}$$

The iterative procedure consists in expanding the spaces $\mathcal{K}_m$ and $\mathcal{C}_m$ until the norm of the residual matrix $\mathcal{R}_m$ (6.2.10) is sufficiently small.

As outlined in Section 2.1, a projection method is efficient only if the subspaces $\mathcal{K}_m$ and $\mathcal{C}_m$ generate important spectral information without the space dimensions being large. The extended Krylov subspace method presents this feature and it is nowadays recognized as one of the most effective procedure for solving standard linear equations with low-rank right-hand side. We want to exploit the approximation properties of $\mathbf{EK}_m^{\square}$ also in our framework and we thus select $\mathcal{K}_m = \mathbf{EK}_m^{\square}(A, \overline{C}_1)$, $\mathcal{C}_m = \mathbf{EK}_m^{\square}(B^T, \overline{C}_2)$ where $\overline{C}_1$, $\overline{C}_2$ are the starting blocks, which we will show how to select in our setting in Sections 6.2.2 and 6.2.3. The procedure is summarized in Algorithm 6.1 where the matrices $\mathcal{Z}_1$ and $\mathcal{Z}_2$ are the low-rank factors of (6.2.9), i.e., they are such that $X_m = \mathcal{Z}_1 \mathcal{Z}_2^T$. Notice that, in the case of generalized Lyapunov equations, the new basis blocks $\mathcal{V}_m$ and $\mathcal{U}_m$ are equal (hence also the basis matrices V_m and U_m) and Algorithm 6.1 can be optimized accordingly. Moreover, for the case of the (standard) Sylvester equation, i.e., equation (5.0.1) with $p = 0$, Algorithm 6.1 is equivalent

Algorithm 6.1: Galerkin projection method for the generalized Sylvester matrix equation.

input : $A, B, N_1 \dots, N_p, M_1, \dots, M_p \in \mathbb{R}^{n\times n}$, $C_1, C_2 \in \mathbb{R}^{n\times r}$ $\overline{C}_1 \in \mathbb{R}^{n\times \bar{r}_1}$ and $\overline{C}_2 \in \mathbb{R}^{n\times \bar{r}_2}$

output: $\mathcal{Z}_1, \mathcal{Z}_2 \in \mathbb{R}^{n\times t}$, $t \le 2m \cdot \min\{\bar{r}_1, \bar{r}_2\}$

1 Set $\beta_1 = \|C_1\|_F$, $\beta_2 = \|C_2\|_F$

2 Perform economy-size QR of $\overline{C}_1$, $\overline{C}_1 = V_1\boldsymbol{\gamma}_1$, and $\overline{C}_2$, $\overline{C}_2 = U_1\boldsymbol{\gamma}_2$. Set $\mathcal{V}_1 \equiv V_1$, $\mathcal{U}_1 \equiv U_1$

for $m = 2, 3, \dots,$ *till convergence,* **do**

3 Compute next basis blocks $\mathcal{V}_m$, $\mathcal{U}_m$ and set $V_m = [V_{m-1}, \mathcal{V}_m]$, $U_m = [U_{m-1}, \mathcal{U}_m]$

4 Update $T_m = V_m^T A V_m$, $J_m := U_m^T B^T U_m$, $D_{m,1} = V_m^T C_1$, $D_{m,2} = U_m^T C_2$, $G_{m,i} := V_m^T N_i V_m$, $F_{m,i} := U_m^T M_i U_m$, $i = 1, \dots, p$

5 Solve $T_m Y_m + Y_m J_m^T + \sum_{i=1}^p G_{m,i} Y_m F_{m,i} - D_{m,1} D_{m,2}^T = 0$

6 Compute $\|R_m\|_F^2 = \|\boldsymbol{\tau}_{m+1,m} E_m^T Y_m\|_F^2 + \|Y_m E_m \boldsymbol{\iota}_{m+1,m}^T\|_F^2$

7 **if** $\|R_m\|_F/(\beta_1\beta_2) < \epsilon$ **then**

8 **Stop**

end

end

9 Compute the singular value decomposition of Y_m and retain $\widehat{Y}_1 \in \mathbb{R}^{2m\bar{r}_1\times t}$, $\widehat{Y}_2 \in \mathbb{R}^{2m\bar{r}_2\times t}$, $t \le 2m \cdot \min\{\bar{r}_1, \bar{r}_2\}$

10 Set $\mathcal{Z}_1 = V_m \widehat{Y}_1$, $\mathcal{Z}_2 = U_m \widehat{Y}_2$

to Algorithm 2.3 with the choice of the starting blocks $\overline{C}_1 = C_1$ and $\overline{C}_2 = C_2$. Under the condition that $\|\mathcal{R}_m\|$ is small, X_m is an approximation of the solution to (5.0.1) such that $\operatorname{rank}(X_m) \le 2m \cdot \min(\bar{r}_1, \bar{r}_2)$, $\bar{r}_1 = \operatorname{rank}(\overline{C}_1)$, $\bar{r}_2 = \operatorname{rank}(\overline{C}_2)$.

As in case of standard equations, the block Arnoldi procedure equipped with the modified block Gram-Schmidt can be employed in the basis construction providing the matrices T_m, J_m in step 4 and the new basis blocks $\mathcal{V}_{m+1}, \mathcal{U}_{m+1}$ in step 3. See Section 2.1 and the discussion at the beginning of Section 2.2.4.

The matrices $G_{m,i}$ and $F_{m,i}$ in step 4 can be computed by extending the matrices $G_{m-1,i}$ and $F_{m-1,i}$ with a block-column and a block-row. Moreover, the matrix X_m is never explicitly formed. In particular, the Frobenius norm of the residual (6.2.10) can be computed as in (2.1.18). This can be shown by an argument similar to the one adopted in Proposition 2.1.1 as the block Arnoldi relations

$$AV_m = V_m T_m + \mathcal{V}_{m+1}\boldsymbol{\tau}_{m+1} E_m^T, \quad BU_m = U_m J_m + \mathcal{U}_{m+1}\boldsymbol{\iota}_{m+1} E_m^T,$$

still hold.

A breakdown in Algorithm 6.1 may occur in two situations. During the generation of the basis of the extended Krylov subspaces, (numerical) loss of orthogonality may occur in step 3. This issue is present already for the Sylvester equation [97, 26] and we refer to [49] for some safeguard strategies that may mitigate the problem. We assume that the bases V_m and U_m have full rank. The other situation where a breakdown may occur is in step 5. It may happen that the projected problem (6.2.11) is not solvable. For the Sylvester equation the solvability of the projected problem is guaranteed by the condition that the field of values of A and $-B$ are disjoint. See Section 2.1.1. We extend this result to the case of generalized Sylvester equations. In our setting we need an additional condition and instead of using the field of values, it is natural to employ the ratio field of values defined in, e.g., [38].

Proposition 6.2.1. *Consider the generalized Sylvester equation (5.0.1) and assume that the field of values of A and $-B$ are disjoint, and that the ratio field of values of $\sum_{i=1}^{p} M_i^T \otimes N_i$ and $B^T \otimes I + I \otimes A$, i.e.,*

$$R\left(\sum_{i=1}^{p} M_i^T \otimes N_i, B^T \otimes I + I \otimes A\right) := \left\{ \frac{y^* \left(\sum_{i=1}^{p} M_i^T \otimes N_i\right) y}{y^* \left(B^T \otimes I + I \otimes A\right) y} \;\middle|\; y \in \mathbb{C}^{n^2} \setminus \{0\} \right\},$$

is strictly contained in the open unit disk. Then the projected problem (6.2.11) has a unique solution.

Proof. Let $\mathcal{L}_{proj}(Y) := T_m Y + Y J_m^T$ and $\Pi_{proj}(Y) := \sum_{i=1}^{p} G_{m,i} Y F_{m,i}$. The projected problem (6.2.11) is equivalently written as $\mathcal{L}_{proj}(Y_m) + \Pi_{proj}(Y_m) - D_{m,1} D_{m,2}^T = 0$. Since A and $-B$ have disjoint fields of values, $\mathcal{L}_{proj}$ is invertible. See Section 2.1.1.

From Theorem 5.2.2 we know that there exists a unique solution Y_m to (6.2.11) if $\rho\left(\mathcal{L}_{proj}^{-1} \Pi_{proj}\right) < 1$. This condition is equivalent to asking $|\lambda| < 1$, where $(\lambda, v) \in \mathbb{C} \times \mathbb{C}^{4m^2 \bar{r}_1 \bar{r}_2} \setminus \{0\}$ is an eigenpair of the following generalized eigenvalue problem

$$\left(\sum_{i=1}^{p} F_{m,i}^T \otimes G_{m,i}\right) v = \lambda (J_m \otimes I + I \otimes T_m) v. \tag{6.2.13}$$

Using the properties of the Kronecker product, equation (6.2.13) can be written as

$$\sum_{i=1}^{p} (U_m^T \otimes V_m^T) \left(M_i^T \otimes N_i\right) (U_m \otimes V_m)\, v = \lambda (U_m^T \otimes V_m^T) \left(B^T \otimes I + I \otimes A\right) (U_m \otimes V_m)\, v.$$

By multiplying the above equation from the left by v^* we have that

$$|\lambda| = \left| \frac{x^* \left(\sum_{i=1}^{p} M_i^T \otimes N_i\right) x}{x^* \left(B^T \otimes I + I \otimes A\right) x} \right|, \quad x := (U_m \otimes V_m)\, v.$$

Since $R\left(\sum_{i=1}^{p} M_i^T \otimes N_i, B^T \otimes I + I \otimes A\right)$ is strictly contained in the unit circle we conclude that $|\lambda| < 1$. □

Proposition 6.2.1 guarantees the well-posedness of the projected problems (6.2.11) but it has a further computational implication. Indeed, assuming that $R\left(\sum_{i=1}^{p} M_i^T \otimes N_i, B^T \otimes I + I \otimes A\right)$ is strictly contained in the unit disk entails that the projected equations (6.2.11) can be solved by Algorithm 14 so that expensive linearizations via Kronecker transformations are not necessary.

6.2.2 Krylov subspace and low-rank commuting matrices

Algorithm 6.1 is efficient only if the starting blocks $\overline{C}_1$ and $\overline{C}_2$ are low-rank matrices and if the subspaces $\mathbf{EK}_m^{\square}(A, \overline{C}_1)$ and $\mathbf{EK}_m^{\square}(B^T, \overline{C}_2)$ have good approximation properties. Therefore, we now derive certain features of the solution to (6.0.1) that naturally suggest a proper choice of the starting blocks. The low-rank of the starting blocks will rely on the low-rank commutation property of the coefficients (6.2.8). We describe the steps of our reasoning as follows.

- The solution to the generalized Sylvester equation (6.0.1) can be represented as a converging Neumann series (5.2.13). By truncating this series, $X^{(\ell)}$ gives an approximation to the solution to (6.0.1).

- The terms in the truncated Neumann series (5.2.14) satisfy a sequence of Sylvester equations where, at each j, the right-hand side of the current equation involves the solution to the previous equation. We consider approximate solutions to this sequence. More precisely, we denote by $\widetilde{Z}_j$ the result of Algorithm 6.1 applied to each Sylvester equation (5.2.14).
- The matrix $\widetilde{X}^{(\ell)} = \sum_{j=0}^{\ell} \widetilde{Z}_j$ is an approximation to the solution to (6.0.1) and we show that it can be factorized as $\widetilde{X}^{(\ell)} = \mathcal{Z}_1 \mathcal{Z}_2^T$ such that $\mathrm{Range}(\mathcal{Z}_1) \subseteq \mathbf{EK}_m^{\square}(A, \overline{C}_1)$ and $\mathrm{Range}(\mathcal{Z}_2) \subseteq \mathbf{EK}_m^{\square}(B^T, \overline{C}_2)$ for certain $\overline{C}_1$ and $\overline{C}_2$. We give a characterization and a procedure for computing $\overline{C}_1$ and $\overline{C}_2$. One condition for these matrices to be low-rank concerns the commutators (6.2.8) being low-rank. These two matrices will be used as starting blocks in Algorithm 6.1.

Although the above reasoning is based on solving a sequence of Sylvester equations, our approach consists of applying Algorithm 6.1, only one time, directly to the generalized Sylvester equation (6.0.1).

We first need a technical result which shows that, if the commutator of two matrices has low rank, then the corresponding commutator, where one matrix is taken to a given power, has also low rank. The rank increases at most linearly with respect to the power of the matrix. The precise statement is presented in the following lemma.

Lemma 6.2.2. *Suppose that A and N are matrices such that $\{A, N\} = P\widetilde{P}^T$. Then,*

$$\{A^j, N\} = \sum_{k=0}^{j-1} A^k P \widetilde{P}^T A^{j-k-1}.$$

Proof. The proof is by induction. The basis of induction is trivially verified for $j = 1$. Assume that the claim is valid for j, then the induction step follows by observing that

$$\{A^{j+1}, N\} = A^{j+1}N - NA^{j+1} = A^j P\widetilde{P}^T + (A^j N - NA^j)A,$$

and applying the induction hypothesis on $A^j N - NA^j$. □

As already pointed out, the low-rank factors of the right-hand side are natural starting blocks for the (standard) Sylvester equation. If we apply this result to the sequence of Sylvester equations in Theorem 5.2.2, we obtain subspaces with a particular structure. For example, the approximation $\mathcal{Z}_{1,0}\mathcal{Z}_{2,0}^T$ to Z_0 provided by Algorithm 6.1 is such that $\mathrm{Range}(\mathcal{Z}_{1,0}) \subseteq \mathbf{EK}_m^{\square}(A, C_1)$ and $\mathrm{Range}(\mathcal{Z}_{2,0}) \subseteq \mathbf{EK}_m^{\square}(B^T, C_2)$. Since Z_0 is contained in the right-hand side of the definition of Z_1, in order to compute an approximation of Z_1, we should consider the subspaces $N_i \cdot \mathbf{EK}_m^{\square}(A, C_1)$ and $M_i^T \cdot \mathbf{EK}_m^{\square}(B^T, C_2)$ for $i = 1, \ldots, p$. By using the low-rank commutation property (6.2.8) such subspaces can be characterized by the following result.

Theorem 6.2.3. *Assume that $A \in \mathbb{R}^{n\times n}$ is nonsingular and let $N \in \mathbb{R}^{n\times n}$ such that $\{A, N\} = P\widetilde{P}^T$ with $P, \widetilde{P} \in \mathbb{R}^{n\times s}$. Let $C \in \mathbb{R}^{n\times r}$, then*

$$N \cdot \mathbf{EK}_m^{\square}(A, C) \subseteq \mathbf{EK}_m^{\square}(A, [NC, P]).$$

Proof. Since $\mathbf{EK}_m^{\square}(A, C) = \mathbf{K}_m^{\square}(A, C) + \mathbf{K}_m^{\square}(A^{-1}, A^{-1}C)$, an element of $N \cdot \mathbf{EK}_m^{\square}(A, C)$ can be written as $N \cdot \sum_{j=0}^{m-1} A^j C \boldsymbol{\varsigma}_j + N \cdot \sum_{j=1}^{m} A^{-j} C \boldsymbol{\upsilon}_j$ where $\boldsymbol{\varsigma}_j, \boldsymbol{\upsilon}_j$ are $r \times r$ matrices. See (2.1.7).

We focus on the first term $N \cdot \sum_{j=0}^{m-1} A^j C \boldsymbol{\varsigma}_j$. By using Lemma 6.2.2, we have

$$N \cdot \sum_{j=0}^{m-1} A^j C \boldsymbol{\varsigma}_j = \sum_{j=0}^{m-1} A^j N C \boldsymbol{\varsigma}_j - \sum_{j=0}^{m-1} \sum_{k=0}^{j-1} A^k P \left(\widetilde{P}^T A^{j-k-1} C \boldsymbol{\varsigma}_j \right),$$

so that, $\mathrm{Range}(N \cdot \sum_{j=0}^{m-1} A^j C \boldsymbol{\varsigma}_j) \subseteq \mathbf{K}_m^{\square}(A, [NC, P]) \subseteq \mathbf{EK}_m^{\square}(A, [NC, P])$.

We can show that $\mathrm{Range}(N \cdot \sum_{j=1}^{m} A^{-j} C \boldsymbol{\upsilon}_j)$ is a subset of $\mathbf{EK}_m^{\square}(A, [NC, P])$ with the same procedure since $\{A^{-1}, N\} = -(A^{-1}P)(A^{-T}\widetilde{P})^T$. □

In order to ease the notation and improve conciseness of the results that follow, we introduce the following multivariate generalization of the Krylov subspace for more matrices

$$\mathcal{G}_d(N_1, \ldots, N_p; P) = \mathrm{span}\left\{ N_{i_1} \cdots N_{i_s} P z \,|\, 1 \le i_j \le p, 0 \le s \le d, z \in \mathbb{R}^r \right\},$$

where $P \in \mathbb{R}^{n \times r}$, that is $\mathcal{G}_d(N_1, \ldots, N_p; P)$ is the space generated by the columns of the matrices obtained multiplying – in any order and with repetition – s of the p matrices N_i, $0 \le s \le d$, and the matrix P. For instance, $\mathcal{G}_2(N_1, N_2; P)$ is generated by the columns of

$$[P,\ N_1 P,\ N_2 P,\ N_1 N_2 P,\ N_2 N_1 P,\ N_1^2 P,\ N_2^2 P]. \tag{6.2.14}$$

Other characterizations of $\mathcal{G}_d$ may be derived via combinatorial tools.

Notice that the definition of $\mathcal{G}_d$ generalizes the definition of the standard block Krylov subspace as $\mathcal{G}_d(N; P) = \mathbf{K}_d^{\square}(N, P)$.

The solution strategy for (6.0.1) outlined at the beginning of this Section is formalized in the following Theorem. In order to state the Theorem we need the result of the application of the extended Krylov method to the (standard) Sylvester equations of the form

$$AZ + ZB^T - C_1 C_2^T = 0, \tag{6.2.15a}$$

$$AZ + ZB^T + \sum_{i=1}^{p} (N_i \mathcal{Z}_{1,j})(M_i \mathcal{Z}_{2,j})^T = 0, \tag{6.2.15b}$$

as described in [97, 26]. As already stated, this is identical to applying Algorithm 6.1 with $p = 0$.

Theorem 6.2.4. *Consider the generalized Sylvester equation (6.0.1), with coefficients commuting according to (6.2.8). Let $\widetilde{Z}_0 = \mathcal{Z}_{1,0}\mathcal{Z}_{2,0}^T$ be the result of Algorithm 6.1 applied to the (standard) Sylvester equation (6.2.15a) with starting blocks $\overline{C}_1 = C_1$ and $\overline{C}_2 = C_2$. Moreover, for $j = 0, \ldots, \ell - 1$, let $\widetilde{Z}_{j+1} = \mathcal{Z}_{1,j+1}\mathcal{Z}_{2,j+1}^T$ be the result of Algorithm 6.1 applied to the Sylvester equation (6.2.15b) with starting blocks $\overline{C}_1 = [N_1 \mathcal{Z}_{1,j}, \ldots, N_p \mathcal{Z}_{1,j}]$ and $\overline{C}_2 = [M_1^T \mathcal{Z}_{2,j}, \ldots, M_p^T \mathcal{Z}_{2,j}]$. Let $\widetilde{X}^{(\ell)}$ be the approximation of the truncated Neumann series (5.2.15) given by*

$$\widetilde{X}^{(\ell)} := \sum_{j=0}^{\ell} \widetilde{Z}_j.$$

Then, there exist matrices $\mathcal{Z}_1, \mathcal{Z}_2, \widehat{C}_1^{(\ell)}, \widehat{C}_2^{(\ell)}$ such that $\mathrm{Range}(\mathcal{Z}_1) \subseteq \mathbf{EK}_{(\ell+1)d}^{\square}(A, \widehat{C}_1^{(\ell)})$ and $\mathrm{Range}(\mathcal{Z}_2) \subseteq \mathbf{EK}_{(\ell+1)d}^{\square}(B^T, \widehat{C}_2^{(\ell)})$ and

$$\widetilde{X}^{(\ell)} = \mathcal{Z}_1 \mathcal{Z}_2^T,$$

where

$$\text{Range}(\widehat{C}_1^{(\ell)}) \subseteq \mathcal{G}_\ell(N_1,\dots,N_p;C_1) + \mathcal{G}_{\ell-1}(N_1,\dots,N_p;P), \tag{6.2.16a}$$

$$\text{Range}(\widehat{C}_2^{(\ell)}) \subseteq \mathcal{G}_\ell(M_1^T,\dots,M_p^T;C_2) + \mathcal{G}_{\ell-1}(M_1^T,\dots,M_p^T;Q), \tag{6.2.16b}$$

and $P \coloneqq [P_1,\dots,P_p]$, $Q \coloneqq [Q_1,\dots,Q_p]$.

Proof. We start proving that for $j = 0,\dots,\ell$, there exists a matrix S_j such that $\text{Range}(\mathcal{Z}_{1,j}) \subseteq \mathbf{EK}^{\square}_{(\ell+1)d}(A,S_j)$ and

$$\text{Range}(S_j) \subseteq \mathcal{G}_j(N_1,\dots,N_p;C_1) + \mathcal{G}_{j-1}(N_1,\dots,N_p;P). \tag{6.2.17}$$

We prove this claim by induction. The basis of induction is trivially verified with $S_0 \coloneqq C_1$ recalling that $\mathcal{G}_0(N_1,\dots,N_p;C_1) = \text{Range}(C_1)$ and assuming $\mathcal{G}_{-1}(N_1,\dots,N_p;C_1) = \varnothing$. We now assume that the claim is valid for $j > 0$ and we perform the induction step. By construction, we have $\text{Range}(\mathcal{Z}_{1,j+1}) \subseteq \mathbf{EK}^{\square}_d(A,[N_1\mathcal{Z}_{1,j},\ \dots,N_m\mathcal{Z}_{1,j}])$. By applying Theorem 6.2.3 and the induction hypothesis, $\text{Range}(N_i\mathcal{Z}_{1,j}) \subseteq \mathbf{EK}^{\square}_{(j+1)d}(A,[N_iS_j,P_i])$ for any $i = 1,\dots,p$. Therefore, $\text{Range}(\mathcal{Z}_{1,j+1}) \subseteq \mathbf{EK}^{\square}_{(j+2)d}(A,[N_1S_j,\dots,N_pS_j,P])$. We define $S_{j+1} \coloneqq [N_1S_j,\dots,N_pS_j,P]$ which concludes the induction.

From (6.2.17) we now obtain the relation

$$\text{Range}([S_0,\dots,S_j]) \subseteq \mathcal{G}_j(N_1,\dots,N_p;C_1) + \mathcal{G}_{j-1}(N_1,\dots,N_p;P),$$

that directly implies (6.2.16a) by setting $\widehat{C}_1^{(\ell)} \coloneqq [S_0,\dots,S_\ell]$. Equation (6.2.16b) follows from completely analogous reasoning. The final conclusion follows by defining $\mathcal{Z}_1 \coloneqq [\mathcal{Z}_{1,0},\dots,\mathcal{Z}_{1,\ell}]$ and $\mathcal{Z}_2 \coloneqq [\mathcal{Z}_{2,0},\dots,\mathcal{Z}_{2,\ell}]$. □

The main message of the previous Theorem can be summarized as follows. The low-rank factors of the approximation to $X^{(\ell)}$ (5.2.15) obtained by sequentially solving the Sylvester equations (5.2.14) by an extended Krylov subspace method are contained in particular extended Krylov subspaces with a specific choice of the starting blocks. In particular the starting blocks are selected as $\overline{C}_1 = \widehat{C}_1^{(\ell)}$, $\overline{C}_2 = \widehat{C}_2^{(\ell)}$ where $\widehat{C}_1^{(\ell)}$ and $\hat{C}_2^{(\ell)}$ fulfill (6.2.16a)-(6.2.16b). Our approach consists in applying Algorithm 6.1 directly to the generalized Sylvester equation (6.0.1) with this choice of the starting blocks.

A practical procedure that generates starting blocks that fulfill (6.2.16) consists in selecting $\overline{C}_1$ and $\overline{C}_2$ such that their columns are respectively a basis of the subspaces

$$\mathcal{G}_\ell(N_1,\dots,N_p;C_1) + \mathcal{G}_{\ell-1}(N_1,\dots,N_p;P)$$

and

$$\mathcal{G}_\ell(M_1^T,\dots,M_p^T;C_2) + \mathcal{G}_{\ell-1}(M_1^T,\dots,M_p^T;Q).$$

For instance, a basis of $\mathcal{G}_2(N_1,N_2;P)$ can be obtained from the columns of the matrix in (6.2.14).

The choice of the starting blocks involves the parameter ℓ. In theory, a suitable choice of ℓ could be derived by using the error estimate in (5.2.16). However, this is not always possible since the quantity $\rho(\mathcal{L}^{-1}\Pi)$ is, in many cases, not known and computationally demanding to approximate. The choice of ℓ is a trade-off between accuracy and efficiency. The starting blocks $\widehat{C}_1^{(\ell)}$ and $\widehat{C}_2^{(\ell)}$, for large ℓ, provide spaces $\mathbf{EK}^{\square}_m(A,\overline{C}_1)$ and $\mathbf{EK}^{\square}_m(B^T,\overline{C}_2)$ with better approximation features, but with potentially higher dimensions. In Figure 6.1 we plot the convergence history of Algorithm 6.1 applied to a (randomly generated) generalized

Lyapunov equation $AX + XA + NXN - cc^T = 0$ with $A \in \mathbb{R}^{n\times n}$ circulant and $N = L + uu^T \in \mathbb{R}^{n\times n}$, L circulant, $u, c \in \mathbb{R}^n$, $n = 1000$. All the matrices and vectors involved have random entries. Since circulant matrices commute, $\{A, N\} = P\widetilde{P}^T$, $P = [Au, u]$, $\widetilde{P} = [u, -Au]$. Algorithm 6.1 is tested for the starting blocks $\widehat{C}^{(\ell)}$ with $\ell = 0, 1, 2$ selected according to Theorem 6.2.3, i.e., $\widehat{C}^{(0)} = c$, $\widehat{C}^{(1)} = [c, Nc, P]$, $\widehat{C}^{(2)} = [c, Nc, N^2c, P, NP]$.

As it can be appreciated from the plot, an increase in ℓ leads to a more accurate approximation to the solution to (6.0.1), but to an increment in the computational cost of the whole procedure as spaces of larger dimensions are generated. Our approach is computationally

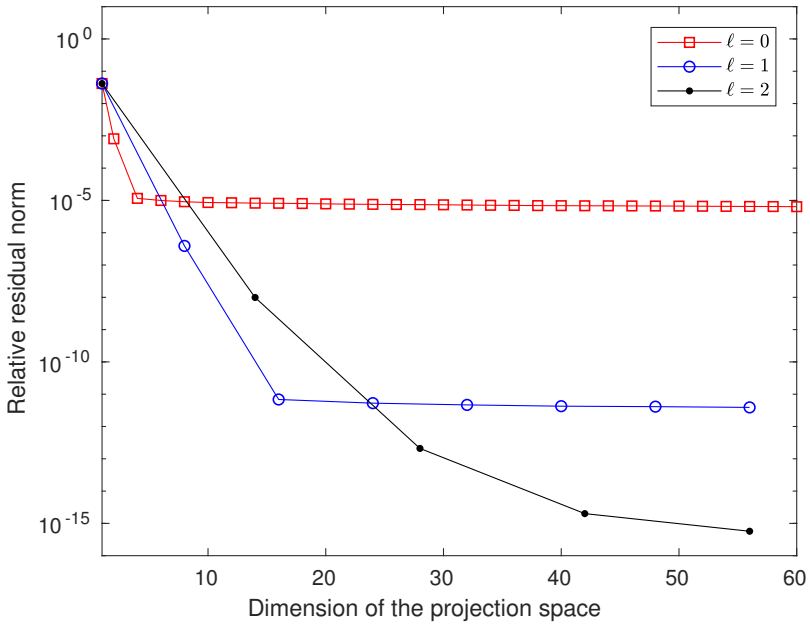

Figure 6.1: Convergence history of Algorithm 6.1 applied to a generalized Lyapunov equation $AX + XA + NXN - cc^T = 0$ with $A \in \mathbb{R}^{n\times n}$ circulant and $N = L + uu^T \in \mathbb{R}^{n\times n}$, $u, c \in \mathbb{R}^n$, $n = 1000$. All the matrices and vectors involved have random entries. Different starting blocks $\widehat{C}^{(\ell)}$ are tested: $\widehat{C}^{(0)} = c$, $\widehat{C}^{(1)} = [c, Nc, P]$, $\widehat{C}^{(2)} = [c, Nc, N^2c, P, NP]$, $P = [Au, u]$.

attractive only if the starting blocks $\overline{C}_1 = \widehat{C}_1^{(\ell)}$ and $\overline{C}_2 = \widehat{C}_2^{(\ell)}$ have low rank, otherwise the dimension of the generated subspaces would grow to quickly leading to unfeasible computational costs.

There are other methods based on generating several projection subspaces (with the same coefficient matrix) as in, e.g., [96, 31]. Since Algorithm 6.1 generates only one pair of extended Krylov subspaces with given starting blocks, an advantage of our approach with respect to these methods consists in avoiding redundancy in the approximation spaces. In particular, if several Krylov subspaces with the same coefficient matrix are generated independently of each other, they may have a nontrivial intersection or in general they may have similar approximation properties. From a computational point of view, this means that considerable efforts are wasted to breed similar information.

In certain cases the dimension of the subspaces $\mathcal{G}_\ell$ is bounded for all the ℓ, i.e., there exist matrices $\overline{C}_1 \in \mathbb{R}^{n\times \bar{r}_1}$ and $\overline{C}_2 \in \mathbb{R}^{n\times \bar{r}_2}$ such that $\mathrm{Range}(\widehat{C}_1^{(\ell)}) \subseteq \mathrm{Range}(\overline{C}_1)$ and $\mathrm{Range}(\widehat{C}_2^{(\ell)}) \subseteq$

Range($\overline{C}_2$) for all ℓ. This condition is satisfied, e.g., if the matrix coefficients N_i, M_i are nilpotent/idempotent or in general if they have low degree minimal polynomials. Therefore, it is possible to select the starting blocks such that Algorithm 6.1 provides an approximation of $X^{(\ell)}$ for all ℓ, i.e., the full series (5.2.13) is approximated. See [64, Section 4.3].

6.2.3 The case of a low-rank $\Pi(X) = \sum_{i=1}^{p} N_i X M_i$

Our numerical method can be improved when $\Pi(X) = \sum_{i=1}^{p} N_i X M_i$ is low-rank. Indeed, if we consider a generalized Sylvester equation (6.0.1) where $N_i = P_i \widetilde{P}_i^T$ and $M_i = \widetilde{Q}_i Q_i^T$ are low-rank matrices for all i, the commutators $\{A, N_i\}$ and $\{B, M_i\}$ are clearly low rank and the theory and the procedure presented in the previous Sections can be adopted as well. However, the solution to (6.0.1) can be further characterized and an efficient (and different) choice of the starting blocks $\overline{C}_1, \overline{C}_2$ can be derived. The assumption $\rho(\mathcal{L}^{-1}\Pi) < 1$ is no longer needed to show the low-rank approximability of X. This property can be illustrated via a Sherman-Morrison-Woodbury argument as proposed in [11]. The following proposition shows that the generalized Sylvester equation (6.0.1) can be implicitly written as a Sylvester equation with right-hand side involving the columns of the matrices P_i and Q_i for $i = 1, \dots, p$.

Proposition 6.2.5. *Consider the generalized Sylvester equation (6.0.1). Assume that $\mathcal{L}$ is invertible, and that $N_i = P_i \widetilde{P}_i^T$ and $M_i = \widetilde{Q}_i Q_i^T$ are such that $P_i, \widetilde{P}_i \in \mathbb{R}^{n \times s_i}$ and $Q_i, \widetilde{Q}_i \in \mathbb{R}^{n \times t_i}$, $s_i, t_i \ll n$. Then there exist $\boldsymbol{\alpha}_i \in \mathbb{R}^{s_i \times t_i}$, $i = 1, \dots, p$, such that*

$$AX + XB - \left(C_1 C_2^T - \sum_{i=1}^{p} P_i \boldsymbol{\alpha}_i Q_i^T \right) = 0. \tag{6.2.18}$$

Proof. The proof follows by defining $\boldsymbol{\alpha}_i := \widetilde{P}_i^T X \widetilde{Q}_i$. □

Proposition 6.2.5 leads to the natural choice of the starting blocks $\overline{C}_1 := [C_1, P_1, \dots, P_p]$ and $\overline{C}_2 := [C_2, Q_1, \dots, Q_p]$ for this kind of equations. Indeed, the right-hand side of the standard Sylvester equation (6.2.18) can be written as $[C_1, P_1, \dots, P_p][C_2, -Q_1\boldsymbol{\alpha}_1^T, \dots, -Q_p\boldsymbol{\alpha}_p^T]^T$, and, although the $\boldsymbol{\alpha}_i$'s are not computable as they involve X, it holds

$$\text{Range}(\overline{C}_2) = \text{Range}([C_2, -Q_1\boldsymbol{\alpha}_1^T, \dots, -Q_p\boldsymbol{\alpha}_p^T]),$$

so that $\mathbf{EK}_m^{\square}(B^T, \overline{C}_2) = \mathbf{EK}_m^{\square}(B^T, [C_2, -Q_1\boldsymbol{\alpha}_1^T, \dots, -Q_p\boldsymbol{\alpha}_p^T])$.

6.2.4 Numerical examples

We now illustrate our approach with some numerical examples where we compare our approach with two different methods for generalized Lyapunov equations: BilADI [11] and GLEK [96]. The results are generally in favor of our approach, since the other methods are less specialized to the specific structure. However, they have a wider applicable problem domain.

Two variants of BilADI are considered. In the first variant we select the Wachspress shifts, see e.g., [115], computed with the software available on Saak's webpage[*]. In the second variant $\mathcal{H}_2$-optimal shifts [10] are used. As proposed in [11], 4 Wachspress shifts and 8 $\mathcal{H}_2$-optimal shifts are considered in the solution process.

The GLEK code is available at the webpage of Simoncini[†]. In this algorithm we selected `tol_inexact`$= 10^{-2}$ while the default setting is used for all the other thresholds.

[*] https://www2.mpi-magdeburg.mpg.de/mpcsc/mitarbeiter/saak/Software/adipars.php
[†] http://www.dm.unibo.it/~simoncin/software.html

The implementation of our approach is based on the modification of K-PIK [97, 26] for generalized Sylvester equation as described in Algorithm 6.1. The projected problems, computed in step 5, are solved with the procedure described in the Section 5.2.1. A MATLAB implementation of Algorithm 6.1 is available online[‡].

In all the methods that we test, the stopping criterion is based on the relative residual norm and the algorithms are stopped when it reaches $\texttt{tol} = 10^{-6}$. We compare: number of iterations, memory requirements, rank of the computed approximation, number of linear solves (involving the matrices A and B, potentially shifted) and total execution CPU-times.

As memory requirement (denoted Mem. in the following tables) we consider the number of vectors of length n stored during the solution process. In particular, for Algorithm 6.1 it consists of the dimension of the approximation space. In GLEK, a sequence of extended Krylov subspaces is generated and the memory requirement corresponds to the dimension of the largest space in the sequence. For the bilinear ADI approach the memory requirement consists of the number of columns of the low-rank factor of the solution. For GLEK, we just report the number of outer iterations. The CPU–times reported for BilADI do no take into account the time for the shifts computation.

All the algorithms require solving linear systems with A and B – potentially shifted – and we precompute the LU-factors of these matrices. During the iterative procedures the resulting triangular systems are solved by the Matlab backslash.

All results were obtained with MATLAB R2015a on a computer with two 2 GHz processors and 128 GB of RAM.

Example 6.2.1. The time invariant multi-input and multi-output (MIMO) bilinear system described in [73, Example 2] yields the following generalized Lyapunov equation

$$AX + XA^T + \gamma^2 \sum_{i=1}^{2} N_i X N_i^T - CC^T = 0, \tag{6.2.19}$$

where $\gamma \in \mathbb{R}$, $\gamma > 0$, $A = \text{tridiag}(2, -\underline{5}, 2)$, $N_1 = \text{tridiag}(3, \underline{0}, -3)$ and $N_2 = -N_1 + I$. We consider $C \in \mathbb{R}^{n\times 2}$ being a normalized random matrix. In the context of bilinear systems, the solution to (6.2.19), referred to as *Gramian*, is used for computing energy estimates of the reachability of the states. The number γ is a scaling parameter selected in order to ensure the solvability of the problem (6.2.19) and the positive definiteness of the solution, namely $\rho(\mathcal{L}^{-1}\Pi) < 1$. This parameter corresponds to rescaling the input of the underlying problem with a possible reduction in the region where energy estimates hold. Therefore, it is preferable not to employ very small values of γ. See [12] for detailed discussions.

For this problem the commutators have low rank, more precisely $\{A, N_1\} = -\{A, N_2\} = P\widetilde{P}^T$, with $P = 2\sqrt{3}[e_1, e_n]$ and $\widetilde{P} = 2\sqrt{3}[e_1, -e_n]$. As proposed in Section 6.2.2 we use Algorithm 6.1 with starting blocks $\overline{C}_1 = \overline{C}_2 = [C, N_1C, P]$ since $\text{Range}(C_1^{(1)}) = \text{Range}([C, N_1C, N_2C, P]) = \text{Range}([C, N_1C, P])$. Notice that in this example we have prefered $\ell = 1$ when choosing the starting block as the selected accuracy has been always attained. If it had not been the case, larger values of ℓ would have been employed.

Table 6.1 illustrates the performances of our approach and the other low-rank methods, GLEK and the BilADI, as γ varies. We notice that the number of linear solves that our projection method requires is always much less than for the other methods. Moreover, it seems that moderate variations of γ, that correspond to variations of $\rho(\mathcal{L}^{-1}\Pi)$, have a smaller influence on the number of iterations in our method compared to the other algorithms.

[‡] http://www.dm.unibo.it/~davide.palitta3

	γ	Its.	Mem.	rank(X)	Lin. solves	CPU time
BilADI (4 Wach.)	1/6	10	55	55	320	51.26
BilADI (8 $\mathcal{H}_2$-opt.)		10	55	55	320	51.54
GLEK		9	151	34	644	14.17
Algorithm 6.1		6	72	60	36	3.77
BilADI (4 Wach.)	1/5	14	71	71	588	55.15
BilADI (8 $\mathcal{H}_2$-opt.)		14	69	69	586	54.31
GLEK		12	173	39	1016	22.06
Algorithm 6.1		6	72	61	36	4.23
BilADI (4 Wach.)	1/4	24	89	89	1454	67.61
BilADI (8 $\mathcal{H}_2$-opt.)		23	89	89	1371	66.83
GLEK		21	218	50	2348	51.49
Algorithm 6.1		8	96	81	48	6.72

Table 6.1: MIMO example. Comparison of low-rank methods for $n = 50000$.

Example 6.2.2. We now consider the following generalized Lyapunov equation

$$AX + XA^T + P\widetilde{P}^T X\widetilde{P}P^T - cc^T = 0, \tag{6.2.20}$$

where $A = n^2\text{tridiag}(1, -2, 1)$ and $P, \widetilde{P} \in \mathbb{R}^{n\times p}$, $c \in \mathbb{R}^n$ have random entries and unit norm. We use Algorithm 6.1, and as proposed in Section 6.2.3, we select $\overline{C}_1 = \overline{C}_2 = [c, P]$ as starting blocks. In Table 6.2 we report the results of the comparison to the other methods for $p = 1$. We notice that our approach requires the lowest number of linear solves. The ADI

	n	Its.	Mem.	rank(X)	Lin. solves	CPU time
BilADI (4 Wach.)	10000	60	57	57	2462	4.25
BilADI (8 $\mathcal{H}_2$-opt.)		42	55	55	1420	2.54
GLEK		4	240	28	310	3.10
Algorithm 6.1		46	184	49	92	2.77
BilADI (4 Wach.)	50000	327	61	61	18673	315.56
BilADI (8 $\mathcal{H}_2$-opt.)		96	61	61	4580	81.47
GLEK		4	454	28	565	24.78
Algorithm 6.1		78	312	47	156	21.09
BilADI (4 Wach.)	100000	-	-	-	-	-
BilADI (8 $\mathcal{H}_2$-opt.)		84	65	65	4058	174.04
GLEK		4	457	29	631	66.77
Algorithm 6.1		97	388	44	194	55.58

Table 6.2: Comparison of low-rank methods applied to (6.2.20) varying n with $p = 1$.

approaches demand the lowest storage because of the column compression strategy performed at each iteration. However, due to the large number of linear solves, these methods are slower compared to our approach. For large-scale problems the BilADI method with 4 Wachspress shifts does not converge in 500 iterations. GLEK provides the solution with the smallest rank.

We now consider (6.2.20) for $p > 1$. Notice that this corresponds to an operator Π defined by the sum of p terms of rank 1. In particular, we apply Algorithm 6.1 to equation (6.2.20) for $p = 5, 10, 15$. The results are collected in Table 6.3. The number of iterations performed decreases as p increases. However, since the rank of the starting block increases with p, the

n	p	Its.	Mem.	rank(X)	Lin. solves	CPU time
10000	5	33	396	50	198	9.38
	10	27	594	48	297	19.87
	15	24	768	44	384	27.35
50000	5	55	660	43	330	54.87
	10	45	990	41	495	117.26
	15	40	1280	42	640	245.87
100000	5	68	816	43	408	133.72
	10	56	1232	41	616	332.68
	15	50	1600	44	800	743.86

Table 6.3: Algorithm 6.1 applied to (6.2.20) varying n and p.

dimension of the approximation space increases, and thus does the number of linear solves. As a result, the computation time increases with p.

If we replace the matrix A with A/n^2 in equation (6.2.20), neither BilADI nor GLEK converge since the Lyapunov operator is no longer dominant, i.e., $\rho(\mathcal{L}^{-1}\Pi) > 1$. However, the existence of a low-rank numerical solution is guaranteed by [11, Theorem 1], and our algorithm still converges providing a solution X in 46 iterations with rank(X) = 184 for $n = 10000$ and $p = 1$. In this case, the projected problems cannot be solved with the approach described in Section 5.2.1. However, since the projected problems are also of the form (6.2.20), they can be solved with a Sherman-Morrison-Woodbury approach for matrix equations [31, 89, 90]. In this case we used the method presented in [31, Section 3].

6.3 Closing considerations

The novel projection spaces we have proposed in Section 6.2.2 deeply rely on the low-rank commutation feature of the matrix coefficients (6.2.8). The structured matrices that present this feature are already analyzed in literature although, to our knowledge, this was never exploited in the setting of Krylov-like methods for matrix equations. Low-rank commuting matrices are usually studied with the *displacement operators*. More precisely, for a given matrix Z, the displacement operator is defined as $F(A) \coloneqq AZ - ZA$. For many specific choices of the matrix Z, e.g., Jordan block, circulant, etc., it is possible to characterize the displacement operator and describe the matrices that are low-rank commuting with Z. See, e.g., [67, 9], [22, Chap. 2, Sec. 11] and references therein. The theory concerning the displacement operator may potentially be used to classify the problems that can be solved with our approach.

The pursued approach is based on the extended Krylov subspace. However, the more general rational Krylov subspace (2.1.9) can be employed as well. Indeed, the commutator $\{A, N\}$ is invariant under translations of the matrix A but further research is needed to characterize the spaces and study efficient shift-selection strategies as it has been done in [35] for standard Lyapunov equations.

In conclusion, we wish to point out that the low-rank approximability characterization given in Theorem 6.1.1 may be of use outside of the scope of projection methods. For instance, Riemannian optimization methods are designed to compute the best rank k approximation (in the sense of, e.g., [71, 112]) to the solution of a matrix equation. This approach is effective only if k is small, i.e., the solution is approximable by a low-rank matrix, for which we have provided sufficient conditions.

Chapter 7

Conclusions

In this thesis the numerical solution of large-scale Sylvester and Lyapunov equations has been deeply analyzed.

We have proposed enhanced projection methods for equations with symmetric coefficient matrices and low-rank right-hand side and, more remarkably, the open problem of Sylvester and Lyapunov equations with not necessarily low-rank right-hand side has been tackled.

In particular, in Chapter 3 we have addressed the case when the coefficient matrices and the right-hand side are generally banded. Depending on the conditioning of the coefficients, two different algorithms have been proposed. In case of well-conditioned coefficient matrices, we have illustrated how the solution X can be well approximated by a banded matrix $\widetilde{X}$ and the matrix-oriented CG method has been shown to be a valid candidate for its computation.

For ill-conditioned problems, X can be represented by a couple (X_B, S_m), $X \approx X_B + S_m S_m^T$, where X_B is banded and S_m low-rank so that a low memory allocation is required. The computation of the two terms X_B and S_m involves several state-of-the-art numerical linear algebra tools. More precisely, sophisticated adaptive quadrature rules, rational approximations to the matrix exponential and recent results on the decay of the inverse of matrices are employed in the computation of X_B whereas S_m requires the construction of a projection space where an ad-hoc procedure for calculating the residual norm has been developed in the stopping criterion.

The banded structure studied in Chapter 3 has been further generalized and in Chapter 4 Sylvester and Lyapunov equations with quasiseparable data have been analyzed. New bounds on the numerical quasiseparable rank of the solution X have been proposed for this broad kind of equations legitimizing the search for a low-rank quasiseparable approximant $\widehat{X} \approx X$. Moderate efforts are needed for operating and storing quasiseparable matrices with a low quasiseparable rank and two efficient numerical methods equipped with $\mathcal{H}$-matrix arithmetic have been illustrated.

In the last part of the thesis we have studied large-scale generalized Sylvester equations of the form

$$\mathcal{L}(X) + \Pi(X) + C = 0,$$

where $\mathcal{L}(X) = AX + XB$ and $\Pi(X) = \sum_{i=1}^{p} N_i X M_i$. If $\rho(\mathcal{L}^{-1}\Pi) < 1$, this equation admits a unique solution and we have shown that it can be well approximated by a low-rank matrix in case of a low-rank right-hand side C, a moderate p and $\rho(\mathcal{L}^{-1}\Pi) \ll 1$. In this case, if the coefficient matrices commute up to low-rank factors, novel approximation spaces for an effective numerical solution based on projection have been proposed.

Bibliography

[1] 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.

[2] O. Axelsson. *Iterative solution methods*. Cambridge University Press, Cambridge, 1994.

[3] G. A. Baker, Jr. and P. Graves-Morris. *Padé approximants*, volume 59 of *Encyclopedia of Mathematics and its Applications*. Cambridge University Press, Cambridge, second edition, 1996.

[4] J. Baker, M. Embree, and J. Sabino. Fast singular value decay for Lyapunov solutions with nonnormal coefficients. *SIAM J. Matrix Anal. Appl.*, 36(2):656–668, 2015.

[5] R. H. Bartels and G. W. Stewart. Algorithm 432: Solution of the Matrix Equation $AX + XB = C$. *Comm. ACM*, 15:820–826, 1972.

[6] U. Baur. Low rank solution of data-sparse Sylvester equations. *Numer. Linear Algebra Appl.*, 15(9):837–851, 2008.

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

[8] B. Beckermann and A. Gryson. Extremal rational functions on symmetric discrete sets and superlinear convergence of the ADI method. *Constr. Approx.*, 32(3):393–428, 2010.

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

[10] P. Benner and T. Breiten. Interpolation-based $\mathcal{H}_2$-model reduction of bilinear control systems. *SIAM J. Matrix Anal. Appl.*, 33(3):859–885, 2012.

[11] P. Benner and T. Breiten. Low rank methods for a class of generalized Lyapunov equations and related issues. *Numer. Math.*, 124(3):441–470, 2013.

[12] P. Benner and T. Damm. Lyapunov equations, energy functionals, and model order reduction of bilinear and stochastic systems. *SIAM J. Control Optim.*, 49(2):686–711, 2011.

[13] P. Benner and P. Kürschner. Computing real low-rank solutions of Sylvester equations by the factored ADI method. *Comput. Math. Appl.*, 67(9):1656–1672, 2014.

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

[15] P. Benner, R.-C. Li, and N. Truhar. On the ADI method for Sylvester equations. *J. Comput. Appl. Math.*, 233(4):1035–1045, 2009.

[16] M. Benzi and D. Bertaccini. Approximate inverse preconditioning for shifted linear systems. *BIT*, 43(2):231–244, 2003.

[17] M. Benzi and N. Razouk. Decay bounds and $O(n)$ algorithms for approximating functions of sparse matrices. *Electron. Trans. Numer. Anal.*, 28:16–39, 2007.

[18] M. Benzi and V. Simoncini. Decay bounds for functions of Hermitian matrices with banded or Kronecker structure. *SIAM J. Matrix Anal. Appl.*, 36(3):1263–1282, 2015.

[19] M. Benzi and M. Tůma. A comparative study of sparse approximate inverse preconditioners. *Appl. Numer. Math.*, 30(2-3):305–340, 1999. Iterative methods and preconditioners (Berlin, 1997).

[20] D. Bertaccini. Efficient preconditioning for sequences of parametric complex symmetric linear systems. *Electron. Trans. Numer. Anal.*, 18:49–64, 2004.

[21] D. A. Bini, S. Massei, and L. Robol. On the decay of the off-diagonal singular values in cyclic reduction. *Linear Algebra Appl.*, 519:27–53, 2017.

[22] D. A. Bini and V. Pan. *Polynomial and matrix computations: fundamental algorithms.* Springer Science & Business Media, 2012.

[23] D.A. Bini, S. Massei, and L. Robol. Efficient cyclic reduction for Quasi-Birth–Death problems with rank structured blocks. *Appl. Numer. Math.*, 2016.

[24] S. Börm, L. Grasedyck, and W. Hackbusch. Hierarchical matrices. *Lect. notes*, 21:2003, 2003.

[25] J. P. Boyd. Exponentially convergent Fourier-Chebyshev quadrature schemes on bounded and infinite intervals. *Journal of scientific computing*, 2(2):99–109, 1987.

[26] T. Breiten, V. Simoncini, and M. Stoll. Low-rank solvers for fractional differential equations. *Electron. Trans. Numer. Anal.*, 45:107–132, 2016.

[27] C. Canuto, V. Simoncini, and M. Verani. On the decay of the inverse of matrices that are sum of Kronecker products. *Linear Algebra Appl.*, 452:21–39, 2014.

[28] A. J. Carpenter, A. Ruttan, and R. S. Varga. Extended numerical computations on the "1/9" conjecture in rational approximation theory. In Peter Russell Graves-Morris, Edward B. Saff, and Richard S. Varga, editors, *Rational Approximation and Interpolation: Proceedings of the United Kingdom - United States Conference held at Tampa, Florida, December 12–16, 1983*, pages 383–411. Springer Berlin Heidelberg, Berlin, Heidelberg, 1984.

[29] W. J. Cody, G. Meinardus, and R. S. Varga. Chebyshev rational approximations to e^{-x} in $[0, +\infty)$ and applications to heat-conduction problems. *J. Approximation Theory*, 2:50–65, 1969.

[30] J. A. Cottrell, T. J. R. Hughes, and Y. Bazilevs. *Isogeometric Analysis: Toward Integration of CAD and FEA*. Wiley Publishing, 1st edition, 2009.

[31] T. Damm. Direct methods and ADI-preconditioned Krylov subspace methods for generalized Lyapunov equations. *Numer. Linear Algebra Appl.*, 15(9):853–871, 2008.

[32] S. Demko, W. F. Moss, and P. W. Smith. Decay rates for inverses of band matrices. *Math. Comp.*, 43(168):491–499, 1984.

[33] I. S. Dhillon. *A new $O(n^2)$ algorithm for the symmetric tridiagonal eigenvalue/eigenvector problem*. PhD thesis, University of California, Berkeley, 1997.

[34] I. S. Dhillon, B. N. Parlett, and C. Vömel. The design and implementation of the MRRR algorithm. *ACM Trans. Math. Software*, 32(4):533–560, 2006.

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

[36] V. Druskin, V. Simoncini, and M. Zaslavsky. Adaptive tangential interpolation in rational Krylov subspaces for MIMO dynamical systems. *SIAM J. Matrix Anal. Appl.*, 35(2):476–498, 2014.

[37] Y. Eidelman, I. Gohberg, and I. Haimovici. *Separable type representations of matrices and fast algorithms. Vol. 1*, volume 234 of *Operator Theory: Advances and Applications*. Birkhäuser/Springer, Basel, 2014.

[38] E. Einstein, C. R. Johnson, B. Lins, and I. Spitkovsky. The ratio field of values. *Linear Algebra Appl.*, 434(4):1119–1136, 2011.

[39] K. V. Fernando and H. Nicholson. On a fundamental property of the cross-Gramian matrix. *IEEE Trans. Circ. Syst.*, 31(5):504–505, 1984.

[40] K. V. Fernando and H. and Nicholson. On the structure of balanced and other principal representations of SISO systems. *IEEE Trans. Automat. Control*, 28(2):228–231, 1983.

[41] R. Freund. On polynomial approximations to $f_a(z)(z-a)^{-1}$ with complex a and some applications to certain non-Hermitian matrices. *Approx. Theory Appl.*, 5(1):15–31, 1989.

[42] A. Frommer and V. Simoncini. Stopping criteria for rational matrix functions of Hermitian and symmetric matrices. *SIAM J. Sci. Comput.*, 30(3):1387–1412, 2008.

[43] W. Gander and W. Gautschi. Adaptive quadrature—revisited. *BIT*, 40(1):84–101, 2000.

[44] G. H. Golub, S. Nash, and C. Van Loan. A Hessenberg-Schur method for the problem $AX + XB = C$. *IEEE Trans. Automat. Control*, 24(6):909–913, 1979.

[45] G. H. Golub and C. F. Van Loan. *Matrix computations*. Johns Hopkins Studies in the Mathematical Sciences. Johns Hopkins University Press, Baltimore, MD, fourth edition, 2013.

[46] L. Grasedyck. Existence and computation of low Kronecker-rank approximations for large linear systems of tensor product structure. *Computing*, 72(3):247–265, 2004.

[47] L. Grasedyck. Existence of a low rank or $\mathscr{H}$-matrix approximant to the solution of a Sylvester equation. *Numer. Linear Algebra Appl.*, 11(4):371–389, 2004.

[48] 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.

[49] M. H. Gutknecht. Krylov subspace algorithms for systems with multiple right hand sides: an introduction. 2006. Available at `http://www.sam.math.ethz.ch/~mhg/pub/delhipap.pdf`.

[50] S. Güttel, E. Polizzi, P. T. P. Tang, and G. Viaud. Zolotarev quadrature rules and load balancing for the FEAST eigensolver. *SIAM J. Sci. Comput.*, 37(4):A2100–A2122, 2015.

[51] A. Haber and M. Verhaegen. Sparse solution of the Lyapunov equation for large-scale interconnected systems. *Automatica J. IFAC*, 73:256–268, 2016.

[52] W. Hackbusch. A sparse matrix arithmetic based on H-matrices. Part I: Introduction to H-matrices. *Computing*, 62(2):89–108, 1999.

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

[54] W. Hackbusch, B. Khoromskij, and S. Sauter. *On H2-matrices*. Springer, 2000.

[55] S. J. Hammarling. Numerical solution of the stable, nonnegative definite Lyapunov equation. *IMA J. Numer. Anal.*, 2(3):303–323, 1982.

[56] N. J. Higham. *Accuracy and stability of numerical algorithms*. SIAM, 2002.

[57] N. J. Higham. *Functions of matrices : theory and computation*. Society for Industrial and Applied Mathematics (SIAM, 3600 Market Street, Floor 6, Philadelphia, PA 19104), 2008.

[58] M. Hochbruck and G. Starke. Preconditioned Krylov subspace methods for Lyapunov matrix equations. *SIAM Matrix Anal. and Appl.*, 16(1):156–171, 1995.

[59] R. Horn and C. Johnson. *Topics in Matrix Analysis*. Cambridge Univ. Press, Cambridge, UK, 1991.

[60] D. Y. Hu and L. Reichel. Krylov-subspace methods for the Sylvester equation. *Linear Algebra Appl.*, 172:283–313, 1992. Second NIU Conference on Linear Algebra, Numerical Linear Algebra and Applications (DeKalb, IL, 1991).

[61] C. Jagels and L. Reichel. The extended Krylov subspace method and orthogonal Laurent polynomials. *Linear Algebra Appl.*, 431(3-4):441–458, 2009.

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

[63] I. M. Jaimoukha and E. M. Kasenally. Oblique projection methods for large scale model reduction. *SIAM J. Matrix Anal. Appl.*, 16(2):602–627, 1995.

[64] E. Jarlebring, G. Mele, D. Palitta, and E. Ringh. Krylov methods for low-rank commuting generalized sylvester equations. *ArXiv: 1704.02167*, 2017.

[65] K. Jbilou and A. J. Riquet. Projection methods for large Lyapunov matrix equations. *Linear Algebra Appl.*, 415(2-3):344–358, 2006.

[66] I. Jonsson and B. Kågstrøm. Recursive blocked algorithms for solving triangular systems — Part II: Two-sided and generalized Sylvester and Lyapunov matrix equations. *ACM Trans. Math. Softw.*, 28(4):416–435, 2002.

[67] T. Kailath and A. H. Sayed. Displacement structure: theory and applications. *SIAM Rev.*, 37(3):297–386, 1995.

[68] T. Kato. *Perturbation Theory for Linear Operators*. Springer-Verlag, Berlin, 1995.

[69] D. Kressner. Memory-efficient Krylov subspace techniques for solving large-scale Lyapunov equations. In *IEEE International Symposium on Computer-Aided Control Systems*, pages 613–618, San Antonio, 2008.

[70] D. Kressner and P. Sirković. Truncated low-rank methods for solving general linear matrix equations. *Numer. Linear Algebra Appl.*, 22(3):564–583, 2015.

[71] D. Kressner, M. Steinlechner, and B. Vandereycken. Preconditioned low–rank Riemannian optimization for linear systems with tensor product structure. *SIAM J. Sci. Comput.*, 38(4):A2018–A2044, 2016.

[72] P. Lancaster. Explicit solutions of linear matrix equations. *SIAM Rev.*, 12(4):544–566, 1970.

[73] Y. Lin, L. Bao, and Y. Wei. Order reduction of bilinear MIMO dynamical systems using new block Krylov subspaces. *Comput. Math. Appl.*, 58(6):1093–1102, 2009.

[74] Y. Lin and V. Simoncini. Minimal residual methods for large scale Lyapunov equations. *Appl. Numer. Math.*, 72:52–71, 2013.

[75] J. Martens. *Second-order optimization for neural networks*. PhD thesis, University of Toronto, 2016.

[76] S. Massei. *Exploiting rank structures in the numerical solution of Markov chains and matrix functions*. PhD thesis, Scuola Normale Superiore di Pisa, 2017.

[77] S. Massei, D. Palitta, and L. Robol. Solving rank structured Sylvester and Lyapunov equations. *ArXiv: 1711.05493*, 2017.

[78] S. Massei and L. Robol. Decay bounds for the numerical quasiseparable preservation in matrix functions. *Linear Algebra Appl.*, 516:212–242, 2017.

[79] Stefano Massei and Leonardo Robol. Decay bounds for the numerical quasiseparable preservation in matrix functions. *Linear Algebra Appl.*, 516:212–242, 2017.

[80] C. Mertens and R. Vandebril. Short recurrences for computing extended Krylov bases for Hermitian and unitary matrices. *Numer. Math.*, 131(2):303–328, 2015.

[81] A. Merz. *Computation of generalized gramians for model reduction of bilinear control systems and time-delay systems*. Dr. Hut, München, 2013.

[82] D. Palitta and V. Simoncini. Matrix-equation-based strategies for convection-diffusion equations. *BIT*, 56(2):751–776, 2016.

[83] D. Palitta and V. Simoncini. Computationally enhanced projection methods for symmetric Sylvester and Lyapunov equations. *J. Comput. Appl. Math.*, 330:648–659, 2018.

[84] B. N. Parlett and D. S. Scott. The Lanczos algorithm with selective orthogonalization. *Math. Comp.*, 33(145):217–238, 1979.

[85] T. Penzl. A cyclic low-rank Smith method for large sparse Lyapunov equations. *SIAM J. Sci. Comput.*, 21(4):1401–1418, 2000.

[86] T. Penzl. Eigenvalue decay bounds for solutions of Lyapunov equations: the symmetric case. *Systems Control Lett.*, 40(2):139–144, 2000.

[87] M. Popolizio and V. Simoncini. Acceleration techniques for approximating the matrix exponential operator. *SIAM Journal on Matrix Analysis and Applications*, 30(2):657–683, 2008.

[88] C. E. Powell, D. Silvester, and V. Simoncini. An efficient reduced basis solver for stochastic Galerkin matrix equations. *SIAM J. Sci. Comput.*, 39(1):A141–A163, 2017.

[89] S. Richter, L. D. Davis, and E. G. Collins Jr. Efficient computation of the solutions to modified Lyapunov equations. *SIAM J. Matrix Anal. Appl.*, 14(2):420–431, 1993.

[90] E. Ringh, G. Mele, J. Karlsson, and E. Jarlebring. Sylvester-based preconditioning for the waveguide eigenvalue problem. *(in press) Linear Algebra Appl. (2017)*, 2017.

[91] J. D. Roberts. Linear model reduction and solution of the algebraic Riccati equation by use of the sign function. *Internat. J. Control*, 32(4):677–687, 1980.

[92] W. E. Roth. The equations $AX - YB = C$ and $AX - XB = C$ in matrices. *Proc. Amer. Math. Soc.*, 3:392–396, 1952.

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

[94] Y. Saad. *Numerical solution of large Lyapunov equations*. In M. A. Kaashoek and J.H. van Schuppen and A. C. Ran, editors, Boston, 1990. Birkhauser.

[95] J. Sabino. *Solution of large-scale Lyapunov equations via the block modified Smith method*. PhD thesis, Rice University, 2006.

[96] S. D. Shank, V. Simoncini, and D. B. Szyld. Efficient low-rank solution of generalized Lyapunov equations. *Numer. Math.*, 134(2):327–342, 2016.

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

[98] V. Simoncini. The Lyapunov matrix equation. Matrix analysis from a computational perspective. *Quaderno UMI*, (55):157–174, 2015. Topics in Mathematics, Bologna - UMI 2015.

[99] V. Simoncini. Computational methods for linear matrix equations. *SIAM Rev.*, 58(3):377–441, 2016.

[100] G. D. Smith. *Numerical Solution of Partial Differential Equations*. 2nd ed. edition, 1978.

[101] J. Snyders and M. Zakai. On nonnegative solutions of the equation $AD + DA' = -C$. *SIAM J. Appl. Math.*, 18:704–714, 1970.

[102] D. C. Sorensen and Y. Zhou. Bounds on eigenvalue decay rates and sensitivity of solutions to Lyapunov equations. *Dept. of Comp. Appl. Math., Rice Univ., Houston, TX, Tech. Rep. TR02-07*, 2002.

[103] F. Stenger. *Numerical methods based on sinc and analytic functions*, volume 20 of *Springer Series in Computational Mathematics*. Springer-Verlag, New York, 1993.

[104] A. Townsend and L. N. Trefethen. An extension of chebfun to two dimensions. *SIAM Journal on Scientific Computing*, 35(6):C495–C518, 2013.

[105] L. N. Trefethen. Is Gauss quadrature better than Clenshaw–Curtis? *SIAM review*, 50(1):67–87, 2008.

[106] L. N. Trefethen, J. A. C. Weideman, and T. Schmelzer. Talbot quadratures and rational approximations. *BIT*, 46(3):653–670, 2006.

[107] J. van den Eshof and M. Hochbruck. Preconditioning Lanczos approximations to the matrix exponential. *SIAM Journal on Scientific Computing*, 27(4):1438–1457, 2006.

[108] H. J. van Waarde, P. Tesi, and M. K. Camlibel. Topology reconstruction of dynamical networks via constrained Lyapunov equations. *ArXiv: 1706.09709*, 2016.

[109] R. Vandebril, M. Van Barel, G. Golub, and N. Mastronardi. A bibliography on semiseparable matrices. *Calcolo*, 42(3):249–270, 2005.

[110] R. Vandebril, M. Van Barel, and N. Mastronardi. *Matrix computations and semiseparable matrices. Eigenvalue and singular value methods*, volume 2. Johns Hopkins University Press, Baltimore, MD, 2008.

[111] R. Vandebril, M. Van Barel, and N. Mastronardi. *Matrix computations and semiseparable matrices. Linear systems*, volume 1. Johns Hopkins University Press, Baltimore, MD, 2008.

[112] B. Vandereycken and S. Vandewalle. A Riemannian optimization approach for computing low-rank solutions of Lyapunov equations. *SIAM J. Matrix Anal. Appl.*, 31(5):2553–2579, 2010.

[113] E. L. Wachspress. Extended application of alternating direction implicit iteration model problem theory. *J. Soc. Indust. Appl. Math.*, 11:994–1016, 1963.

[114] E. L. Wachspress. *Iterative solution of elliptic systems, and applications to the neutron diffusion equations of reactor physics*. Prentice-Hall, Inc., Englewood Cliffs, N.J., 1966.

[115] E. L. Wachspress. *The ADI model problem*. Springer, New York, 2013.

[116] D. S. Watkins. *The Matrix Eigenvalue Problem: GR and Krylov Subspace Methods*. SIAM, Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 1st edition, 2007.

[117] H. Yuan, R. Xi, C. Chen, and M. Deng. Differential network analysis via lasso penalized d-trace loss. *Biometrika*, 104(4):755–770, 2017.

[118] E. Zolotarev. Application of elliptic functions to questions of functions deviating least and most from zero. *Zap. Imp. Akad. Nauk. St. Petersburg*, 30(5):1–59, 1877.