Efficient Low-Rank Solution of Large-Scale Matrix Equations

Dissertation
zur Erlangung des akademischen Grades

doctor rerum naturalium
(Dr. rer. nat.)

von **Patrick Kürschner, M. Sc.**

geb. am **26.02.1983** in Frankenberg

genehmigt durch die Fakultät für Mathematik
der Otto-von-Guericke-Universität Magdeburg

Gutachter: **Prof. Dr. Peter Benner**

Prof. Dr. Serkan Gugercin

eingereicht am: **15.10.2015**

Verteidigung am: **19.02.2016**

Forschungsberichte aus dem Max-Planck-Institut
für Dynamik komplexer technischer Systeme

Band 45

Patrick Kürschner

Efficient Low-Rank Solution of Large-Scale Matrix Equations

Shaker Verlag
Aachen 2016

Bibliographic information published by the Deutsche Nationalbibliothek
The Deutsche Nationalbibliothek lists this publication in the Deutsche Nationalbibliografie; detailed bibliographic data are available in the Internet at http://dnb.d-nb.de.

Zugl.: Magdeburg, Univ., Diss., 2016

Printed in Germany.

ISBN 978-3-8440-4385-3
ISSN 1439-4804

Shaker Verlag GmbH • P.O. BOX 101818 • D-52018 Aachen
Phone: 0049/2407/9596-0 • Telefax: 0049/2407/9596-9
Internet: www.shaker.de • e-mail: info@shaker.de

PUBLICATIONS

Large parts of this thesis have been published in the papers summarized below. The Chapters 3 and 4 are rearranged, partly revised, and extended versions of

> [37]: Peter Benner, Patrick Kürschner, Jens Saak: An Improved Numerical Method for Balanced Truncation for Symmetric Second Order Systems, Mathematical and Computer Modelling of Dynamical Systems, 19(6), pp. 593–615, 2013.
>
> [38]: Peter Benner, Patrick Kürschner, Jens Saak: Efficient Handling of Complex Shift Parameters in the Low-Rank Cholesky Factor ADI Method, Numerical Algorithms, 62(2), pp. 225–251, 2012.
>
> [36]: Peter Benner, Patrick Kürschner, Jens Saak: A Reformulated Low-Rank ADI Iteration with Explicit Residual Factors, Proceedings of Applied Mathematics and Mechanics, 19(1), pp. 585–586, 2013.
>
> [32]: Peter Benner, Patrick Kürschner: Computing Real Low-Rank Solutions of Sylvester Equations by the Factored ADI Method, Computers & Mathematics with Applications 67(9), pp. 1656–1672, 2014.

Chapter 5 includes

> [39]: Peter Benner, Patrick Kürschner, Jens Saak: Self-Generating and Efficient Shift Parameters in ADI Methods for Large Lyapunov and Sylvester Equations, Electronic Transactions on Numerical Analysis, 43, pp. 142–162, 2014.

but several substantial extensions were added.

Chapter 6, especially Section 6.2, contains a modified version of

> [41]: Peter Benner, Patrick Kürschner, Jens Saak: Low-Rank Newton-ADI methods for Large Nonsymmetric Algebraic Riccati Equations, Journal of the Franklin Institute 353(5), pp. 1147—1167, 2016.

Chapter 7 contains selected material from [37] as well as

> [33]: Peter Benner, Patrick Kürschner, Jens Saak: A Goal-Oriented Dual LRCF-ADI for Balanced Truncation, I. Troch and F. Breitenecker, eds., Vol. 7 of *Proceedings of the MathMod 2012, IFAC-PapersOnline, Mathematical Modelling*, Vienna, Austria, 2012, pp. 752–757.
>
> [178]: C. Nowakowski, Patrick Kürschner, Peter Eberhard, Peter Benner: Model Reduction of an Elastic Crankshaft for Elastic Multibody Simulations, ZAMM - Journal of Applied Mathematics and Mechanics 93(4), pp. 198–216, 2013.
>
> [40]: Peter Benner, Patrick Kürschner, Jens Saak: Frequency-Limited Balanced Truncation with Low-Rank Approximations, SIAM Journal on Scientific Computing 38(1), pp. A471–A499, 2016.

In particular, Section 7.3 is a slightly altered version of [40].

ACKNOWLEDGEMENTS

First of all, I want to thank my supervisor Professor Peter Benner for his guidance during all these years I have been employed in his research group. He always managed to push me towards the right directions. Without his inspiring lectures in my days as undergraduate, I would not have found so much delight in numerical linear algebra and control theory.
I also thank Professor Serkan Gugercin who agreed to be the second referee for this thesis.

During my time in the research groups *Mathematics in Industry and Technology Technik* it the TU Chemnitz and *Computational Methods in Systems and Control Theory* at the *Max Planck Institute for Dynamics of Complex Technical Systems* in Magdeburg, I had the pleasure to work or even share an office with several friendly colleagues. I am very grateful for all the countless inspiring research-related and private conversations. A few of those good colleagues I'd like to point out explicitly. I want to thank Jens Saak for introducing me to the topic of matrix equations and to TikZ a long time ago in Chemnitz. I would also like to thank my colleagues Tobias Breiten, Matthias Voigt, André Schneider, Norman Lang, and Heiko Weichelt for creating such a friendly atmosphere at work. The latter deserves my special gratitude for proofreading this thesis.

I am also thanking Michiel Hochstenbach from TU Eindhoven and Melina Freitag from the University of Bath. The scientific exchange with them inspired and helped me a lot in pursuing my second favorite topic in numerical linear algebra: large-scale eigenvalue problems. Both organized research visits for me at their universities.

Zum Abschluß gilt mein Dank meinen Eltern, die mich stets unterstützt und mir Rückhalt gegeben haben. Natürlich danke ich auch meiner Freundin Maria, die immer für mich da war, inbesondere in der Zeit des Aufschreibens der Arbeit. Ich wünsche Dir ein ebenso schönes Promotionsstudium.

ABSTRACT

In this thesis, we investigate the numerical solution of large-scale, algebraic matrix equations. The focus lies on numerical methods based on the alternating directions implicit (ADI) iteration, which can be formulated to compute approximate solutions of matrix equations in form of low-rank factorizations. These low-rank versions of the ADI iteration can be used to deal with large-scale Lyapunov and Sylvester equations. The major part of this thesis is devoted to improving the performance of these iterative methods. At first, we develop algorithmic enhancements that aim at reducing the computational effort of certain stages in each iteration step. This includes novel low-rank expressions of the residual matrix, which allows an efficient computation of the residual norm, and approaches for the reduction of the amount of occurring complex arithmetic operations. ADI based methods rely on shift parameters, which influence how fast the iteration generates an approximate solution. For this, we propose novel shift generation strategies which improve the convergence speed of the low-rank ADI iteration and, at the same time, can be performed in an automatic and cost efficient numerical way. Later on, the improved low-rank ADI methods for Lyapunov and Sylvester equations are used in Newton type methods for finding approximate solutions of quadratic matrix equations in the form of symmetric, continuous-time, but also more general nonsymmetric, algebraic Riccati equations. In the last part of this thesis, the methods for solving large-scale Lyapunov equations are applied in order to execute balanced truncation model order reduction for linear control systems in a numerically feasible way. For frequency-limited balanced truncation, which is a special variant of balanced truncation, a novel algorithmic framework is proposed that enables an efficient numerical execution of this model reduction technique.

ZUSAMMENFASSUNG

Die vorliegende Arbeit befasst sich mit der numerischen Lösung von großskaligen, algebraischen Matrixgleichungen. Der Fokus liegt hierbei auf numerischen Verfahren, die auf der Methode der alternierenden Richtungen (ADI) basieren und so formuliert werden können, dass sie Näherungslösungen von Matrixgleichungen in Form von Niedrigrangfaktorisierungen berechnen. Diese Niedrigrang-Versionen der ADI Iteration sind in der Lage, hochdimensionale Lyapunov- und Sylvestergleichungen zu behandeln. Im Hauptteil dieser Arbeit werden numerische Verbesserungen dieser iterativen Verfahren untersucht. Dazu werden einige algorithmische Erweiterungen entwickelt, um den Rechenaufwand von bestimmten Abschnitten der einzelnen Iterationsschritte zu senken. Dies beinhaltet neue Niedrigrangdarstellungen des Residuums, welche eine recheneffiziente Berechnung der Residuumsnorm erlauben, sowie Strategien zur Verringerung der Anzahl auftretender Rechenoperationen in komplexer Arithmetik. Die ADI Iteration benötigt Shiftparameter, die beeinflussen, wie schnell das Verfahren eine Näherungslösung findet. Neuartige Strategien zur Shiftberechnung werden vorgestellt, die sowohl die Konvergenzgeschwindigkeit der Niedrigrang-ADI Iteration verbessern, als auch automatisch und kosteneffizient durchgeführt werden können. Die verbesserten Niedrigrang-ADI Iterationen werden später innerhalb Newton-artiger Verfahren angewendet, um Näherungslösungen von quadratischen Matrixgleichungen zu berechnen. Dabei werden symmetrische, zeit-kontinuierliche und auch die allgemeineren unsymmetrischen, algebraischen Riccatigleichungen betrachtet. Im letzten Abschnitt der vorliegenden Arbeit finden die Verfahren zur numerischen Lösung von großskaligen Lyapunovgleichungen Anwendung, um eine Modellordnungsreduktion für lineare Regelungssysteme mittels balancierten Abschneidens mit geringen Rechenaufwand durchzuführen. Für balanciertes Abschneiden in begrenzten Frequenzintervallen, eine Sonderform des balancierten Abschneidens, wird eine neue numerische Strategie erarbeitet, die eine effiziente numerische Durchführung dieses speziellen Modellordnungsreduktionsverfahrens ermöglicht.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ALGORITHMS

Acronyms and Abbreviations

ADI	alternating directions implicit
ARE	algebraic Riccati equation
(FL)BT	(frequency-limited) balanced trunction
DAE	differential-algebraic equation
EKSM	extended Krylov subspace method
EVD	eigenvalue decomposition
(G)CALE	(generalized) continuous-time algebraic Lyapunov equation
(G)CASE	(generalized) continuous-time algebraic Sylvester equation
(G)DALE	(generalized) discrete-time algebraic Lyapunov equation
(G)CARE	(generalized) continuous-time algebraic Riccati equation
(G)DARE	(generalized) discrete-time algebraic Riccati equation
IRKA	iterative rational Krylov algorithm
LTI	linear time-invariant
MATLAB®	software from The MathWorks Inc.
(M)NARE	nonsymmetric algebraic Riccati equation (associated to M-matrix)
MOR	model order reduction
RKSM	rational Krylov subspace method
SVD	singular value decomposition

Notation

$\mathbb{R}$, $\mathbb{C}$	fields of real and complex numbers
$\mathbb{C}_+$, $\mathbb{C}_-$	open right/open left complex half plane
$\mathbb{R}_+$, $\mathbb{R}_-$	strictly positive/negative real line
$\mathbb{R}^n$, $\mathbb{C}^n$	vector space of real/complex n-tuples
$\mathbb{R}^{m\times n}$, $\mathbb{C}^{m\times n}$	real/complex $m \times n$ matrices
$\lvert\xi\rvert$,	absolute value of real or complex scalar
$\arg \xi$	argument of complex scalar
$\mathbb{D}$	$:= \{z \in \mathbb{C} \,:\, \lvert z\rvert < 1\}$, the open unit disc
$\jmath$	imaginary unit ($\jmath^2 = -1$)
Re (A), Im (A)	real and imaginary part of a complex quantity $A = \text{Re}\,(A) + \jmath\,\text{Im}\,(A) \in \mathbb{C}^{n\times m}$

$\overline{A}$	$:= \operatorname{Re}(A) - \jmath \operatorname{Im}(A)$, complex conjugate of $A \in \mathbb{C}^{n\times m}$
a_{ij}	the i,j-th entry of A
$A(i:j,:)$, $A(:,k:\ell)$	rows $i,\dots,j$ of A and columns $k,\dots,\ell$ of A
$A(i:j,k:\ell)$	rows $i,\dots,j$ of columns $k,\dots,\ell$ of A
A^T	the transpose of A
A^H	$:= (\overline{A})^T$, the complex conjugate transpose
A^{-1}	inverse of nonsingular A
A^{-T}, A^{-H}	inverse of A^T, A^H
I_n, $I_{n,r}$	identity matrix of dimension n, first r columns of I_n
$\mathbf{1}_r$	$:= (1,\dots,1)^T \in \mathbb{R}^r$
$\Lambda(A)$, $\Lambda(A,M)$	spectrum of matrix A/matrix pair (A,M)
$\lambda_j(A)$, $\lambda_j(A,M)$	j-th eigenvalue of $A/(A,M)$
$\rho(A,M)$	$:= \max_j \lvert\lambda_j(A,M)\rvert$, spectral radius of (A,M)
$\sigma_{\max}(A)$	largest singular value of A
$\operatorname{tr}(A)$	$:= \sum_{i=1}^n a_{ii}$, trace of A
$\lVert u\rVert_p$	$:= \sqrt[p]{\sum_{i=1}^n \lvert u_i\rvert^p}$ for $u \in \mathbb{C}^n$ and $1 \le p < \infty$
$\lVert u\rVert_\infty$	the maximum norm ($\lVert u\rVert_\infty = \max_i \lvert u_i\rvert$)
$\lVert A\rVert_p$	$:= \sup\{\lVert Au\rVert_p : \lVert u\rVert_p = 1\}$, subordinate matrix p-norm, $1 \le p \le \infty$
$\lVert A\rVert_F$	$:= \sqrt{\sum_{i,j} \lvert a_{ij}\rvert^2} = \sqrt{\operatorname{tr}(A^H A)}$, the Frobenius-norm of matrix $A \in \mathbb{C}^{m\times n}$
$\lVert u\rVert$, $\lVert A\rVert$	Euclidean vector-, or subordinate matrix norm $\lVert\cdot\rVert_2$
$\kappa_p(A)$	$:= \lVert A\rVert_p \lVert A^{-1}\rVert_p$ the p-norm condition number for regular A
$\kappa(A)$	the 2-norm condition number for regular A
$A \succ (\succeq) 0$, $A \prec (\preceq) 0$	short form for A is self-adjoint positive/negative (semi)definite, also abbreviated by *s(s)pd* and *s(s)nd*
$A > (\ge) B$	$:\Leftrightarrow A - B > (\ge) 0$, i.e., element wise partial ordering: $(a_{ij} - b_{ij} > (\ge) 0,\ \forall ij)$
$A \otimes B$	the Kronecker product of A and B (Definition 2.17)
$\operatorname{vec}(A)$	vectorization operator applied to matrix A (Definition 2.17)
$\partial_{x_j} f := \frac{\partial}{\partial x_j} f$	partial derivative with respect to x_j of f
$\partial_{x_j x_k} f = \frac{\partial^2}{\partial x_j \partial x_k} f$	$:= \partial x_j \partial x_k f$, second order partial derivative with respect to x_j and x_k of f
$\partial^2_{x_j} f := \partial_{x_j x_j} f$	second order partial derivative with respect to x_j of f
$\dot f := \partial_t f := \frac{\partial}{\partial t} f$	the derivative with respect to time of f
$\ddot f := \frac{\partial^2}{\partial t^2} f$	second derivative with respect to time of f
$\nabla f := (\partial_{x_1} f, \dots, \partial_{x_n} f)^T$	the gradient of f
$\Delta f := \sum_{i=1}^n \partial^2_{x_i} f$	the *Laplacian operator* applied to f
$\nabla^2 f := \begin{bmatrix} \partial^2_{x_1} f & \cdots & \partial_{x_1 x_n} f \\ \vdots & \ddots & \vdots \\ \partial_{x_n x_1} f & \cdots & \partial^2_{x_n} f \end{bmatrix}$	the *Hessian matrix* of f

CHAPTER 1

INTRODUCTION

Contents

1.1. Motivation and Background

Matrix equations arise in many scientific areas, e.g., optimal control, observer design and approximation of control systems, image reconstruction, transport theory, game theory, damping optimization, and eigenvalue perturbation theory, to name only a few. In the last decades, especially the success and extensive research on model order reduction, an area devoted to the approximation of large control systems, has lead to a growing demand for efficient numerical methods for large-scale matrix equations. Here, large-scale should be understood as the situation when the size of the matrices defining a matrix equation is so large that using eigenvalue or related factorizations as well as storing the sought solution cannot be done in a reasonable amount of computation time and memory consumption. In practice one, therefore, tries to approximate the solution by a low-rank factorization which consumes much less memory and, at the same time, can be computed even for large dimensions using different numerical algorithms that employ techniques from large-scale numerical linear algebra. The use of such low-rank approximations is justified by the experimentally observed and theoretically investigated rapid decay of the singular values of the solutions. The following small experiment illustrates this strategy.

1.2. A First Illustrative Example

Consider the matrix equation

$$AX + XA^T + BB^T = 0, \quad A = -\operatorname{diag}(1, \ldots, n) \in \mathbb{R}^{n \times n}, \quad B = \mathbf{1}_n \in \mathbb{R}^n$$

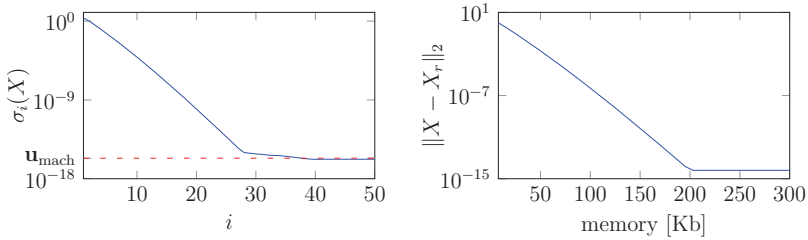


Figure 1.1.: Singular values of X (left) and approximation error against consumed storage (right).

with $n = 1,000$, and $X = X^T \in \mathbb{R}^{n\times n}$ is the sought solution. This matrix equation belongs to the class of Lyapunov equations which are discussed in more detail later on. Although $n = 1,000$ can not actually be considered as large, this small toy example will prove sufficient to explain the basic strategies to deal with large matrix equations. Obviously, the coefficient matrix A is sparse, i.e., only a small number of the total n^2 entries are nonzero. This property is not passed to the solution X for whose entries it is easy to see that

$$X_{ij} = \frac{-1}{a_{ii} + a_{jj}} = \frac{1}{i + j} \neq 0, \quad i, j = 1, \ldots, n.$$

Hence, taking the symmetry of X into account yields a total number of $\frac{1}{2}n(n-1) = 499,500$ entries which have to be computed and stored. In the usual double precision arithmetic, this amounts to roughly 7.6 Mb of storage. Let $X = U\Sigma V^T$ be a singular value decomposition (SVD) of X with $U^TU = I_n$ and $\Sigma = \operatorname{diag}(\sigma_1, \ldots, \sigma_n) \succeq 0$. The singular values are ordered non-increasingly and, due to the symmetry of X, it holds $V = U$. The left plot in Figure 1.1 shows the first 50 computed singular values which obviously decay very rapidly towards zero. The dashed line indicates the machine precision $\mathbf{u}_{\text{mach}} = 2.2204 \cdot 10^{-16}$ in the double-precision floating-point format w.r.t. the IEEE 754 standard. The constant behavior of the computed singular values below that line should not be trusted as it is merely an effect of the difficulties in computing quantities whose magnitude is close to or smaller than $\mathbf{u}_{\text{mach}}$. More information regarding what properties and conditions influence this decay will be given later. We can use the first r singular values and vectors to construct an approximate solution $X_r = Z_r Z_r^T \approx X$ with $Z_r := U(:, 1:r) \operatorname{diag}\left(\sqrt{\sigma_1}, \ldots, \sqrt{\sigma_r}\right) \in \mathbb{R}^{n\times r}$. The matrix Z_r is per construction of rank r and therefore referred to as *low-rank solution factor* which requires only nr entries to store. By the well known theory of such SVD based approximations [111], this yields an error $\|X - X_r\|_2 = \sigma_{r+1}$. For instance, taking $r = 20$ already leads to $\|X - X_{20}\|_2 = 2.25 \cdot 10^{-11}$ requiring only 20000 entries which corresponds to approximately 156 Kb. The right plot in Figure 1.1 shows the obtained error against the consumed memory as r is increased. It is evidently possible to accurately approximate

the solution X by $X_r = Z_r Z_r^T$ of low-rank which requires much less storage. For practical relevant problems, however, it will neither be possible to construct X explicitly via such a simple formula as above, nor computing its SVD in a reasonable numerical effort to construct such low-rank approximations. The numerical algorithms investigated in the thesis at hand construct the low-rank solution factors Z_r by different concepts than the above SVD based approach. These approaches will work without computing singular or eigenvalue decompositions of the defining matrices and mostly employ only operations with sparse matrices that can be performed efficiently even for large dimensions n.

1.3. Overview of this Thesis

In Chapter 2 some concepts of eigenvalue and mathematical control theory, which are required in this thesis, are reviewed. The different classes of algebraic matrix equations the thesis is concerned with are introduced along with a description of the already mentioned low-rank phenomenon which forms the backbone of the upcoming investigated algorithms. Afterwards, a concise overview of available numerical methods for various different classes of matrix equations is given. Large parts of this thesis are devoted to enhancements of numerical methods for matrix equations. These improvements will be evaluated using different test examples which are introduced as well.

This thesis is mainly concerned with one particular numerical approach for matrix equations which is based on the alternating directions implicit (ADI) iteration. This method is introduced in more depth in Chapter 3 for solving large-scale Lyapunov and Sylvester equations. There, we introduce the low-rank version of the ADI iteration for computing low-rank solutions of the considered large-scale linear matrix equations. Different issues of this low-rank iteration are investigated step by step to improve the efficiency of the algorithms. As first numerical enhancement, a novel result on the structure of the residual in the low-rank ADI iteration is derived, which not only enables a cheap way to compute the residual norm as an obvious stopping criterion, but also gives new alternative formulations of the low-rank ADI methods. Specially tailored variants for dealing with certain structured matrix equations are also investigated.

In general, ADI methods require shift parameters to converge rapidly. Although in almost every practical situation the arising matrix equations to be solved are defined in the field of real numbers, the shift parameters can nevertheless be complex in several situations. This introduces complex arithmetic operations in the ADI iteration and also leads to the generation of complex approximate solutions of the matrix equations. The efficient treatment of these undesirable effects is topic of Chapter 4.

The dependence on shift parameters itself is often considered as the biggest disadvantage of ADI based methods. Chapter 5 targets this problem and some new computationally efficient shift strategies are derived, where the shifts are computed automatically in the course of the low-rank ADI iteration, basically erasing the need to worry about shift parameters at all. It will turn out that those novel shift parameters are capable of outperforming several existing shift approaches in most cases, and even make the low-rank ADI iteration competitive to other low-rank solvers.

The next two chapters have a more application oriented character because the previously investigated low-rank ADI iterations are employed there for dealing with matrix equations that arise as part of a greater goal. Chapter 6 considers the numerical computation of low-rank solutions of algebraic Riccati equations via hybrid methods of Newton's schemes and ADI methods. All of the previously derived numerical enhancements are included in these algorithms.

Afterwards, Chapter 7 is concerned with a particular model order reduction technique, namely balanced truncation and related approaches. There, large-scale algebraic matrix equations have to be solved as intermediate task for which the previously examined low-rank methods are used. In the second part of this chapter, a novel efficient numerical framework for carrying out balanced truncation in limited frequency intervals is proposed. This modification of balanced truncation again involves large-scale matrix equations to be solved, but also further demanding tasks which haves to be dealt with numerically.

CHAPTER 2

MATHEMATICAL BASICS AND PRELIMINARIES

Contents

2.1. Useful Concepts from Matrix- and Eigenvalue Theory

2.1.1. Matrices and Their Eigenvalues

Definition 2.1:

For a matrix $A \in \mathbb{C}^{n\times n}$, an *eigenvalue* $\lambda \in \mathbb{C}$ and its *(right) eigenvector* $x \in \mathbb{C}^n\backslash\{0\}$ form an *eigenpair* $(\lambda,\ x)$ of A which satisfies the eigenvalue problem

$$Ax = \lambda x. \tag{2.1}$$

Likewise, the *left eigenvector* $y \in \mathbb{C}^n\backslash\{0\}$ corresponding to λ fulfills

$$y^H A = \lambda y^H. \tag{2.2}$$

Together, they form an *eigentriple* $(\lambda,\ x,\ y)$ of A . The eigenvalues are also the roots of the *characteristic polynomial* $p_A(\lambda) = \det(A - \lambda I_n)$. The *spectrum* $\Lambda(A)$ is the set of all eigenvalues $\{\lambda_1, \ldots, \lambda_n\}$ of A. ◊

The *algebraic multiplicity* $\alpha(\lambda)$ of a particular eigenvalue is the number of times λ appears as root of $p_A(\lambda)$. The number of linearly independent right and left eigenvectors $x,\ y$ associated to λ can, however, be smaller than $\alpha(\lambda)$ and is given by the *geometric multiplicity* $1 \leq \gamma(\lambda) := \dim\ker(A - \lambda I_n) \leq \alpha(\lambda)$. If $\gamma(\lambda) = \alpha(\lambda)$ then λ is called (semi)simple eigenvalue. For most parts of this thesis it will be sufficient to restrict to diagonalizable matrices in the sense of the next definition.

Definition 2.2:

If for $A \in \mathbb{C}^{n\times n}$ there exists a nonsingular matrix $X \in \mathbb{C}^{n\times n}$ containing n right eigenvectors as columns such that

$$X^{-1}AX = \operatorname{diag}(\lambda_1, \ldots, \lambda_n) =: \Lambda,$$

where the eigenvalues $\lambda_i \in \Lambda(A)$ are now counted with their algebraic multiplicities, then A is called *diagonalizable* matrix. Otherwise, we speak of an *non-diagonalizable* matrix. ◊

Lemma 2.3 ([198]):

Let $A \in \mathbb{C}^{n\times n}$ be diagonalizable having distinct eigenvalues $\Lambda(A) = \{\lambda_1, \ldots, \lambda_{\hat{n}}\}$ with $\hat{n} \leq n$. Then it holds $\gamma(\lambda_i) = \alpha(\lambda_i)$ for all $i = 1, \ldots, \hat{n}$. ◊

If the right and left eigenvectors are scaled such that $y_i^H x_i = 1,\ \forall\ i = 1, \ldots, n$, i.e., they form bi-orthogonal bases, then the matrix $Y := X^{-H}$ contains the left eigenvectors as columns, and, hence, $Y^H AX = \Lambda$, $Y^H X = I_n$.

Definition 2.4:

A matrix $A \in \mathbb{C}^{n\times n}$ is called *normal* if it is diagonalizable and its left coincide with its right eigenvectors. Otherwise, A is a *non-normal* matrix. ◊

Lemma 2.5 ([151, 132]):

The following statements are equivalent:

a) A is a normal matrix.

b) $A^H A = AA^H$.

c) There exist a unitary matrix $Q \in \mathbb{C}^{n\times n}$, $Q^H Q = I_n$ such that $Q^H AQ = \Lambda$ with $\Lambda := \operatorname{diag}(\lambda_1, \ldots, \lambda_n)$. ◊

It is easy to see that Hermitian ($A = A^H$) and skew-Hermitian ($A = -A^H$) matrices are normal.

Definition 2.6:

For two matrices $A,\ M \in \mathbb{C}^{n\times n}$ the *generalized eigenvalue problem* is given by

$$Ax = \lambda Mx, \quad y^H A = \lambda y^H M, \tag{2.3}$$

where the definitions of eigenvalues, right and left eigenvectors, eigenpairs and -triples, as well as algebraic multiplicity, for (2.3) are directly adopted from Definition 2.1. In the following the matrices A, M linked to a generalized eigenvalue problem are referred to as *matrix pair* (A, M). The set of all eigenvalues is again called spectrum and denoted by $\Lambda(A, M)$. The characteristic polynomial of (A, M) is $p_{A,M}(\lambda) = \det(A - \lambda M)$. If M is singular, $\Lambda(A, M)$ contains eigenvalues at infinity and the *finite spectrum* $\Lambda_f(A, M)$ denotes the set of all finite eigenvalues of (A, M). A matrix pair (A, M) is called singular if $A - \lambda M$ is singular for all $\lambda \in \mathbb{C}$, otherwise it is called regular. $\Diamond$

The concept of diagonalizable and non-diagonalizable matrix pairs (A, M) is analogue to Definition 2.2 and Lemma 2.3. We restrict ourselves to regular, diagonalizable matrix pairs in the remainder. In the diagonalizable case, the eigenvectors corresponding to finite and infinite eigenvalues can be scaled such that $y_i^H M x_i = 1$ and $y_i^H M x_i = 0$, respectively, such that the right and left eigenvectors corresponding to finite eigenvalues form bi-orthonormal bases w.r.t. the matrix M. Hence, for nonsingular M there exist nonsingular matrices $X,\ Y \in \mathbb{C}^{n\times n}$ having the right and left eigenvectors as columns such that

$$Y^H A X = \operatorname{diag}(\lambda_1, \ldots, \lambda_n), \quad Y^H M X = I_n,$$

where $Y := (MX)^{-H}$. Moreover, $\Lambda(A, M) = \Lambda(M^{-1}A)$.

Definition 2.7:

A matrix pair $(A, M) \in \mathbb{C}^{n\times n} \times \mathbb{C}^{n\times n}$ with nonsingular M is called normal matrix pair if it is diagonalizable and its left coincide with its right eigenvectors. $\Diamond$

Lemma 2.8:

Let (A, M) be a regular matrix pair with nonsingular M. Then the following statements are equivalent:

a) (A, M) is a normal matrix pair.

b) There exists a $Q \in \mathbb{C}^{n\times n}$ such that $Q^H A Q = \operatorname{diag}(\lambda_1, \ldots, \lambda_n)$ and $Q^H M Q = I_n$.$\Diamond$

Remark 2.9:

So far everything was defined for complex matrices, but the main parts of this work are devoted to real matrices since they are more relevant in practice. A real matrix pair $(A, M) \in \mathbb{R}^{n\times n} \times \mathbb{R}^{n\times n}$ might have real as well as complex eigenvalues. The latter ones occur in complex conjugate pairs, i.e., it holds $\{\lambda, \overline{\lambda}\} \in \Lambda(A, M)$ if $\lambda \in \mathbb{C}$. Similarly, if $x,\ y$ are the right and left eigenvectors corresponding to the eigenvalue $\lambda \in \mathbb{C}$, then $\overline{x},\ \overline{y}$ are the eigenvectors corresponding to $\overline{\lambda}$. Note that if, e.g., $A = A^T$ and $M = M^T \succeq 0$, all eigenvalues, and hence all eigenvectors, are real. $\Diamond$

The next definition is for brevity only provided for eigenvalues of matrix pairs. In case of a single matrix, M needs to be replaced by I_n.

Definition 2.10:

Let $\lambda_i \in \Lambda(A, M)$, $i = 1, \ldots, n$, with nonsingular M. Then the eigenvalue λ_i is called

a) *c-(anti)stable* if Re $(\lambda_i) < 0$ (Re $(\lambda_i) > 0$),

b) *d-(anti)stable if* $|\lambda_i| < 1$ ($|\lambda_i| > 1$).

Likewise, (A, M) is called c-stable, c-antistable, d-stable, or, d-antistable if the above properties hold for all eigenvalues, i.e., $\Lambda(A, M) \subset \mathbb{C}_-$, $\Lambda(A, M) \subset \mathbb{C}_+$, $\Lambda(A, M) \subset \mathbb{D}$, or $\mathbb{D} \subsetneq \Lambda(A, M)$. A c-stable matrix A is also called *Hurwitz*. ◊

The prefixes c- and d- are motivated by stability concepts for the study of continuous-time and discrete-time differential equations, respectively, which will also play a role later. There, semisimple eigenvalues on the boundary of the stability region, i.e. Re $(\lambda_i) = 0$ ($|\lambda_i| = 1$), might also be called c-(d-)stable (cf. [129, Theorem 3.3.211], Definition 2.21).

A concept closely related to eigenvalue problems is the singular value problem.

Definition 2.11:

Let $A \in \mathbb{C}^{n\times m}$ and assume w.l.o.g. that $n \geq m$. The *singular value problem* of A is to be understood as finding non-negative *singular values* $\sigma \in \mathbb{R}$, *right* and *left singular vectors* $u \in \mathbb{C}^m$, $v \in \mathbb{C}^n$ satisfying

$$Au = \sigma v, \quad A^H v = \sigma u. \tag{2.4}$$

The singular values form, together with their singular vectors, a *singular triple* $(\sigma,\ u,\ v)$. The *singular value decomposition (SVD)* of A is given by

$$\begin{aligned} A &= V\Sigma U^H, \quad V = [v_1, \ldots, v_n] \in \mathbb{C}^{n\times n},\ U = [u_1, \ldots, u_m] \in \mathbb{C}^{m\times m} \text{ unitary}, \\ \Sigma &= \begin{bmatrix} \Sigma_1 \\ 0 \end{bmatrix} \in \mathbb{R}^{n\times m}, \quad \Sigma_1 = \operatorname{diag}(\sigma_1, \ldots, \sigma_m) \in \mathbb{R}^{m\times m}, \\ \sigma_{\max} &:= \sigma_1 \geq \ldots \geq \sigma_r > \sigma_{r+1} = \sigma_m = 0, \end{aligned}$$

where $r = \operatorname{rank}(A)$.

The *thin SVD* of A is obtained by taking only the first m singular triples into account:

$$A = V_1\Sigma_1 U_1^H, \quad V_1 = [v_1, \ldots, v_m] \in \mathbb{C}^{n\times m},\ U_1 = [u_1, \ldots, u_m] \in \mathbb{C}^{m\times m}. \qquad ◊$$

Next to the rank of a matrix A, the singular values also enable the definition of the numerical rank of A which is of tremendous importance for the algorithms considered in this thesis.

Definition 2.12:

Let $A \in \mathbb{C}^{n\times m}$.

a) If $r = \operatorname{rank}(A) \ll \min(m, n)$, we refer to A as *low-rank matrix*.

b) The *numerical rank* [58] of $A \in \mathbb{C}^m$ with respect to a threshold $\tau \in \mathbb{R}_+$ (typically $\tau \ll 1$) is defined by

$$\operatorname{rank}(A,\ \tau) = \arg\max_j \left\{ \frac{\sigma_j}{\sigma_1} \geq \tau,\ j = 1, \ldots, \min(m,n) \right\}.$$

c) The matrix A is said to be of low numerical rank (w.r.t. the threshold τ) if $\operatorname{rank}(A,\ \tau) \ll r \leq \min(m,n)$. This trivially includes low-rank matrices. ◊

Functions of Matrices and Spectral Transformations

At several occasions in this thesis we will also encounter functions of square matrices in the sense of the following definition.

Definition 2.13 (Matrix functions [128, Definition 1.11]):
Consider a matrix $A \in \mathbb{C}^{n\times n}$ and a function f which is analytic on and inside a closed contour Γ surrounding $\Lambda(A)$. Then the function f evaluated at A is to be understood in the sense

$$f(A) = \frac{1}{2\pi\jmath} \int_\Gamma f(z)(zI_n - A)^{-1} \mathrm{d}z. \qquad \Diamond$$

Lemma 2.14 (spectral mapping theorem [128, Theorem 1.13]):
Let $f : D \to \mathbb{C}$ be an analytic function defined in a domain $D \subset \mathbb{C}$. For any $A \in \mathbb{C}^{n\times n}$ with $\Lambda(A) = \{\lambda_1, \ldots, \lambda_n\} \subset D$ it holds that $\Lambda(f(A)) = \{f(\lambda_1), \ldots, f(\lambda_n)\}$. ◊

The following special rational matrix function is important for dealing with the matrix equations in this thesis.

Definition 2.15 (Cayley transformation):
Let $A \in \mathbb{C}^{n\times n}$ and $\mu,\ \nu \in \mathbb{C}$ with $-\mu \notin \Lambda(A)$.

a) A *generalized Cayley transformation* of A is defined by the rational matrix function

$$\mathcal{C}(A, \mu, \nu) =: (A + \mu I_n)^{-1}(A - \nu I_n).$$

In the special case $\nu = \overline{\mu}$, we simply write $\mathcal{C}(A, \mu) = (A + \mu I_n)^{-1}(A - \overline{\mu} I_n)$ and omit the prefixed generalized.

b) For matrix pairs (A, M), the generalized Cayley transformation is given by

$$\mathcal{C}(A, M, \mu, \nu) := (A + \mu M)^{-1}(A - \nu M)$$

with $-\mu \notin \Lambda(A, M)$. As above we use the notation $\mathcal{C}(A, M, \mu) := \mathcal{C}(A, M, \mu, \overline{\mu})$. ◊

Proposition 2.16:
Consider $(A, M) \in \mathbb{C}^{n\times n}$ with nonsingular M and $\mu,\ \nu \in \mathbb{C}$ with $-\mu \notin \Lambda(A, M)$.

a) The generalized Cayley transformations can be represented by

$$\begin{aligned}\mathcal{C}(A, \mu, \nu) &= (A + \mu I_n)^{-1}(A - \nu I_n) = (A - \nu I_n)(A + \mu I_n)^{-1} \\ &= I_n - (\mu + \nu)(A + \mu I_n)^{-1} \quad \text{and} \\ \mathcal{C}(A, M, \mu, \nu) &= (A + \mu M)^{-1}(A - \nu M) = M^{-1}(A - \nu M)(A + \mu M)^{-1}M \\ &= I_n - (\mu + \nu)(A + \mu M)^{-1}M = M^{-1}(I_n - (\mu + \nu)M(A + \mu M)^{-1})M.\end{aligned}$$

b) The eigenvalues of $\mathcal{C}(A, M, \mu, \nu)$ are given by $(\lambda - \nu)/(\lambda + \mu)$ for $\lambda \in \Lambda(A, M)$.

c) The transformation $\mathcal{C}(A, M, \mu)$ with $\mu \in \mathbb{C}_-$ maps the c-stable, c-antistable and imaginary eigenvalues of (A, M) to the interior, exterior and, respectively, boundary of the unit disc $\mathbb{D}$. Thus, if $\Lambda(A, M) \subset \mathbb{C}_-$, it holds $\Lambda(\mathcal{C}(A, M, \mu)) \subset \mathbb{D}$ so that $\rho(\mathcal{C}(A, M, \mu)) < 1$. Stated differently, $\mathcal{C}(A, M, \mu)$ is a d-stable matrix if (A, M) is a c-stable matrix pair. ◊

Proof. The expressions in part a) follow from simple algebraic manipulations. The part b) is a direct consequence of Lemma 2.14 and the invertibility of M. For the last part consider

$$|\lambda(\mathcal{C}(A, M, \mu))|^2 = |(\lambda - \overline{\mu})/(\lambda + \mu)|^2 = 1 - 4\,\mathrm{Re}\,(\mu)\,\mathrm{Re}\,(\lambda)\,/|\lambda + \mu|^2, \quad \lambda \in \Lambda(A, M)$$

from which the claims follow easily. □

2.1.2. Kronecker Product and Vectorization Operator

For certain theoretical investigations, the *Kronecker product* and *vectorization operator* are useful.

Definition 2.17 ([111, Section 12.1]):
Let $X = [x_1, \ldots, x_m] \in \mathbb{C}^{n\times m}$ and $Y \in \mathbb{C}^{p\times q}$. Then

$$\mathrm{vec}\,(X) := \begin{bmatrix} x_1 \\ \vdots \\ x_m \end{bmatrix} \in \mathbb{C}^{nm\times 1}, \quad X \otimes Y = \begin{bmatrix} x_{11}Y & \ldots & x_{1m}Y \\ \vdots & & \vdots \\ x_{n1}Y & \ldots & x_{nm}Y \end{bmatrix} \in \mathbb{C}^{np\times mq}. \qquad ◊$$

Proposition 2.18 ([111, Chapter 12.3]):
Let $X \in \mathbb{C}^{n\times m}$, $Z \in \mathbb{C}^{p\times q}$, $U \in \mathbb{C}^{m\times r}$, $Q \in \mathbb{C}^{q\times \ell}$, and $Y \in \mathbb{C}^{m\times p}$. Then the following statements hold.

a) $(X \otimes Z)(U \otimes Q) = (XU) \otimes (ZQ)$.

b) $\operatorname{vec}(XYZ) = (Z^T \otimes X)\operatorname{vec}(Y)$.

c) If $m = n = r$, $p = q = \ell$, and $U,\ Q$ are nonsingular, then

$$\Lambda(X \otimes Z,\ U \otimes Q) = \{\lambda_j \mu_k\ : \lambda_j \in \Lambda(X,\ U),\ \mu_k \in \Lambda(Z,\ Q), \\ j = 1, \ldots, n,\ k = 1, \ldots, p\}.$$

If x_j (y_j) and z_k (v_k) are the linearly independent right (left) eigenvectors of $(X,\ U)$ and, respectively, $(Z,\ Q)$, then the corresponding right (left) eigenvectors of $(X \otimes Z,\ U \otimes Q)$ are $x_j \otimes z_k$ ($y_j \otimes v_k$). ◊

The next theorem gives a result on the eigenvalues of certain functions of Kronecker products and is crucial for investigating the linear matrix equations discussed in this thesis.

Theorem 2.19 (Theorem of Stephanos [169, Theorem 43.8]):
Let $A \in \mathbb{C}^{n\times n},\ B \in \mathbb{C}^{m\times m}$ with $\Lambda(A) = \{\lambda_1, \ldots, \lambda_n\}$, $\Lambda(B) = \{\mu_1, \ldots, \mu_m\}$. For a bivariate polynomial $p(x, y) = \sum_{i,j=0}^{k} c_{ij} x^i y^j$, $c_{ij} \in \mathbb{C}$ we define by

$$p(A,\ B) := \sum_{i,j=0}^{k} c_{ij} (A^i \otimes B^j)$$

a polynomial of the two matrices. Then the spectrum of $p(A,\ B)$ is

$$\Lambda(p(A,\ B)) = \{p(\lambda_r, \mu_s),\ r = 1, \ldots, n,\ s = 1, \ldots, m\}.$$ ◊

2.2. Important Concepts from Linear Control Theory

Before we discuss matrix equations, we briefly introduce some preliminaries from linear control theory. This is motivated by the fact that several of the considered classes of matrix equations have important applications in control theory.

Definition 2.20:
Linear, time-invariant, continuous-time, control systems in generalized state space form are given by

$$\begin{aligned} E\dot{x}(t) &= Ax(t) + Bu(t), \quad x(0) = x_0, \\ y(t) &= Cx(t) + Du(t). \end{aligned} \tag{2.5}$$

The matrices $A,\ E \in \mathbb{R}^{n\times n}$, E nonsingular, $B \in \mathbb{R}^{n\times m}$, $C \in \mathbb{R}^{p\times n}$, and $D \in \mathbb{R}^{p\times m}$ are referred to as *generalized state-space* and *mass, input, output,* and *feed-through matrix*, respectively. The time-dependent, vector valued functions $x(t) \in \mathbb{R}^n$, $u(t) \in \mathbb{R}^m$, $y(t) \in \mathbb{R}^p$ are typically called *generalized state, input or control*, and *output vector*.

Likewise, *linear, time-invariant, discrete-time, control systems* in generalized state space form are for $k \in \mathbb{N}_0$ defined by

$$\begin{aligned} Ex(k+1) &= Ax(k) + Bu(k), \quad x(0) = x_0, \\ y(k) &= Cx(k) + Du(k). \end{aligned} \tag{2.6}$$

We will make use of the following notation $(E;\ A,\ B,\ C,\ D)$ to refer to systems (2.5) and (2.6) defined by $A,\ E,\ B,\ C,\ D$. For most parts of this thesis it is sufficient to consider the case $D = 0$ denoted by $(E;\ A,\ B,\ C)$ because $D \neq 0$ will not change the used concepts and derivations. If the matrix E in (2.5) above is singular one usually speaks of differential-algebraic equations (DAEs). For this thesis these will only play a minor role such that we restrict ourselves to nonsingular E here. The following concepts of stability, controllability, observability, stabilizability, and detectability in the next definition will be crucial for providing existence theorems regarding the solutions of matrix equations. More information as well as the proofs of the given (and further) equivalences can be found, e.g., in the textbooks [150, 72, 3, 129].

Definition 2.21:

Let $A,\ E,\ B,\ C$ be defined as above with E nonsingular. The system $(E;\ A,\ B,\ C)$ is

a) *asymptotically c-(d-)stable* if $\Lambda(A,\ E) \subset \mathbb{C}_-$ $(\Lambda(A,\ E) \subset \mathbb{D})$,

b) *c-(d-)stable* if $\Lambda(A,\ E) \subseteq \mathbb{C}_-$ $(\Lambda(A,\ E) \subseteq \mathbb{D})$ and $\alpha(\lambda) = \gamma(\lambda)$ for all eigenvalues with $\mathrm{Re}\,(\lambda) = 0$ $(|\lambda| = 1)$,

c) controllable if $\mathrm{rank}\,([A - \mu E,\ B]) = n,\ \forall \mu \in \mathbb{C}$, or equivalently $y^H B = 0$ for every left eigenvector of $(A,\ E)$,

d) observable if $\mathrm{rank}\left([A^T - \mu E^T,\ C^T]\right) = n,\ \forall \mu \in \mathbb{C}$, or equivalently $Cx = 0$ for every right eigenvector of $(A,\ E)$,

e) c-(d-)stabilizable if $\mathrm{rank}\,([A - \mu E,\ B]) = n,\ \forall \mu \in \mathbb{C}_+$ $(\mu \in (\mathbb{C} \backslash \mathbb{D}))$ and

f) c-(d-)detectable if $\mathrm{rank}\left([A^T - \mu E^T,\ C^T]\right) = n,\ \forall \mu \in \mathbb{C}_+$ $(\mu \in (\mathbb{C} \backslash \mathbb{D}))$, respectively. ♢

As before, the prefixes c- and d- point towards the use of these concepts w.r.t. (2.5) and (2.6), respectively. For instance, c-stabilizability is of importance only for (2.5) while d-stabilizability is typically not. We will employ the typical slight misuse of language and neglect the word asymptotically w.r.t. to stability definitions.

2.3. Matrix Equations

Throughout this thesis, a matrix equation should be understood as the problem of finding the solution $X \in \mathbb{C}^{m \times n}$ of

$$\mathcal{F}(X) = 0, \quad \mathcal{F} : \mathbb{C}^{m \times n} \to \mathbb{C}^{n \times k}, \tag{2.7}$$

where $\mathcal{F}$ is a function in X. It might, e.g., involve sums, and products of X with other, known matrices, but also nonlinear and other expressions of X. We consider only *algebraic* matrix equations, i.e., $\mathcal{F}$ is a purely algebraic expression. This excludes differential matrix equations, where X is a sought matrix with functions as entries and $\mathcal{F}$ involves its derivatives. Furthermore, we restrict to cases where $\mathcal{F}$ is a superposition of linear, quadratic, or rational functions of X and exclude cases involving, e.g., X^T, X^H, or $\overline{X}$. Although the majority of topics can be covered using complex matrices, we mainly consider the case of real matrix equations, because almost every practical application, where algebraic matrix equations arise, leads to real matrices. In what follows, we introduce the algebraic matrix equations covered by this thesis and also give some remarks on the existence of a unique solution. We distinguish between linear and certain nonlinear matrix equations.

2.3.1. Linear Matrix Equations

Definition 2.22 (Linear matrix equation):

A *linear matrix equation* is of the form

$$\sum_{i=1}^{\ell} A_i X B_i + Q = 0, \quad A_i \in \mathbb{R}^{n\times n},\ B_i \in \mathbb{R}^{m\times m},\ Q \in \mathbb{R}^{n\times m} \tag{2.8}$$

with the sought solution $X \in \mathbb{R}^{n\times m}$. The constant term Q is usually called inhomogeneity of (2.8). The number of summands $\ell \geq 1$ is referred to as *the length of the matrix equation*. ◊

This thesis is exclusively concerned with matrix equations with $Q \neq 0$.

Lemma 2.23:

The linear matrix equation (2.8) has a unique solution X if and only if $0 \notin \Lambda(\mathcal{A})$, where

$$\mathcal{A} := \sum_{j=1}^{\ell} B_i^T \otimes A_i \in \mathbb{R}^{nm\times nm}. \qquad \lozenge$$

Proof. Vectorization of (2.8) leads with Proposition 2.18b to the equivalent linear system

$$\mathcal{A}\,\mathrm{vec}\,(X) = -\,\mathrm{vec}\,(Q) \tag{2.9}$$

from which the claim follows. □

In general it is hard to specify the statement of the above lemma, e.g., in terms of the spectra of the coefficient matrices A_i, B_i. We only consider linear matrix equations of length $\ell = 2$ in this thesis. These cases are not only very important in practice but for this situation Lemma 2.23 can be made more precise. In the next subsections we discuss important classes of linear matrix equations of length two.

Lyapunov Equations

Lyapunov equations are symmetric matrix equations in the sense that $X = X^T \in \mathbb{R}^{n\times n}$ and $Q = Q^T \in \mathbb{R}^{n\times n}$. For the general linear matrix equation (2.8), this means that for each $i = 1, \ldots, \ell$ there exists a j such that $A_i = B_j^T$ and $B_i = A_j^T$. For matrix equations of length $\ell = 2$ there are the following very important cases which are introduced below.

Definition 2.24 (Continuous-time Lyapunov equations):

For $A \in \mathbb{R}^{n\times n}$ and $Q = Q^T \in \mathbb{R}^{n\times n}$ a *continuous-time, algebraic Lyapunov equation (CALE)* is of the form

$$AX + XA^T + Q = 0. \tag{2.10}$$

For a nonsingular $E \in \mathbb{R}^{n\times n}$ a *generalized, continuous-time, algebraic Lyapunov equation (GCALE)* is defined by

$$AXE^T + EXA^T + Q = 0. \tag{2.11}$$

The additional specification *continuous-time* originates from the relation of the above Lyapunov equations with (2.5). The next lemma gives conditions for the existence of a unique solution of (2.11). For (2.10), the result translates by setting $E = I_n$.

Lemma 2.25 ([3, 151, 129]):

The generalized Lyapunov equation has a unique solution if and only if $\lambda_j \neq -\lambda_k$ $\forall \lambda_j, \lambda_k \in \Lambda(A,\ E)$. Sufficient conditions for this case are $(A,\ E)$ being c-stable or c-antistable.

In the c-stable case, X can be written as

$$X = \int_0^\infty \exp\left(E^{-1}At\right)E^{-1}QE^{-T}\exp\left(E^{-1}At\right)^T \mathrm{d}t$$

which is positive (negative) semi-definite if Q is positive (negative) semi-definite. Strict definiteness of X can be also achieved if $(E;\ A,\ Q)$ is controllable.

For the c-antistable case similar results follow by changing the signs, the order of the integration limits, and by interchanging positive with negative semi-definiteness.$\Diamond$

Proof. Due to E nonsingular, (2.11) can be equivalently rewritten to the Lyapunov equation $E^{-1}AX + XA^TE^{-T} + E^{-1}QE^{-T} = 0$. This amounts to setting $A_1 = E^{-1}A$, $B_2 = A^TE^{-T} = A_1^T$, $A_2 = B_1 = I_n$ in (2.8) so that

$$\mathcal{A} = I_n \otimes E^{-1}A + E^{-1}A \otimes I_n.$$

By Theorem 2.19, $\Lambda(\mathcal{A}) = \{\lambda_j + \lambda_k\ :\ \forall \lambda_j, k \in \Lambda(E^{-1}A)\}$. Thus, $0 \in \Lambda(\mathcal{A})$ if and only if $\lambda_j + \lambda_k = 0$ for some j, k from which the first claim follows since $\Lambda(E^{-1}A) = \Lambda(A,\ E)$. The statements regarding c-stable and c-antistable spectra easily follow, too. For the rest

of the proof we restrict to the c-stable case, as the remaining results of the c-antistable case can be deduced easily. The integral expression can be verified by plugging it into the Lyapunov equations, using $\frac{\mathrm{d}}{\mathrm{d}t}\exp(Ht) = H\exp(Ht)\ \forall H \in \mathbb{R}^{n\times n}$, and because the matrix exponential converges to zero as $t \to \infty$ for c-stable matrices [129]. From the integral formula and the fact that $Q \succeq 0$, the definiteness properties of X can be read off.

For the strict positive definiteness of X in the case $Q \succeq 0$, $(E; A,\ Q)$ controllable, we restrict ourselves for simplicity as well as brevity to the standard case $E = I_n$ and follow [151, Chapter 13, Theorem 4]. Assume X is singular, i.e. $X \succeq 0$, which implies $\exists\ 0 \neq v \in \mathbb{R}^n$ such that $Xv = 0$. Consequently, $v^TQv = v^T(AX + XA^T)v = 0$ and, thus, $v^TAX = v^TQ - v^TXA^T$. Inserting (2.10) $n-1$ times into this equation leads to $v^TA^kQ = 0$ for $k = 0, \ldots, n-1$ which, by [3, Corollary 4.8], violates the controllability assumption. For the case $E \neq 0$ we refer to, e.g., [217]. □

Definition 2.26 (Discrete-time algebraic Lyapunov equations):

For $A \in \mathbb{R}^{n\times n}$ and $Q = Q^T \in \mathbb{R}^{n\times n}$, a *discrete-time, algebraic Lyapunov equation (DALE)* is given by

$$AXA^T - X + Q = 0 \tag{2.12}$$

and its generalized version, the *generalized, discrete-time, algebraic Lyapunov equation (GDALE))*,

$$AXA^T - EXE^T + Q = 0 \tag{2.13}$$

for nonsingular $E \in \mathbb{R}^{n\times n}$. ◊

Here, the additional specification *discrete-time* originates from discrete-time, linear, time-invariant, dynamical systems (2.6).

Remark 2.27:

In some works, the DALEs and GDALEs are called *Stein* and *generalized Stein equations*, respectively. These denominations are, however, not used consistently and sometimes also refer to discrete-time Sylvester equations (see Definition 2.31) whereas (2.12) is referred to as symmetric Stein equation. Hence, to avoid confusion we mainly stick to the term *(generalized) discrete-time algebraic Lyapunov equation* for (2.12),(2.13). ◊

Conditions regarding a unique solution of (2.11) are given by the next lemma which can be proved similarly as Lemma 2.25.

Lemma 2.28 ([151],[150]):

The GDALE (2.13) has a unique solution if and only if $\lambda_j\lambda_k \neq 1,\ \forall \lambda_j, \lambda_k \in \Lambda(A,\ E)$. Obvious sufficient conditions are $(A,\ E)$ being d-stable ($\Lambda(A,\ E) \subset \mathbb{D}$) or d-antistable ($\Lambda(A,\ E) \cap \mathbb{D} = \emptyset$). In the d-stable case, the solution can be expressed as

$$X = \sum_{k=0}^{\infty} (E^{-1}A)^k E^{-1} Q E^{-T} (A^T(E^{-T}))^k$$

and $X \succ (\succeq) 0$ is if $Q \succ (\succeq) 0$. For $(A,\ E)$ d-antistable, similar statements are given by changing the sign, the order of the limits in the infinite sum, and by replacing positive with negative semi-definiteness. ◊

Remark 2.29:

In the context of DAEs, some results concerning solutions of GCALEs and GDALEs with singular E can, e.g., be found in [217, 171, 218]. ◊

Remark 2.30:

The identification "generalized" is used differently for linear matrix equation with $\ell > 2$, e.g., in [20]. In this thesis, we exclusively use "generalized" for matrix equations with $\ell = 2$, where the coefficients can be related to a matrix pair $(A,\ E)$. ◊

Sylvester Equations

A Sylvester equation is a general matrix equations (2.8) without symmetry restrictions on the involved matrices. In the following, we discuss important cases for $\ell = 2$.

Definition 2.31 (Sylvester equations):

For $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{m\times m}$ and $Q \in \mathbb{R}^{n\times m}$, a matrix equation

$$AX - XB + Q = 0 \tag{2.14}$$

is referred to as *continuous-time, algebraic Sylvester equation (CASE)*. For nonsingular $E \in \mathbb{R}^{n\times n}$, $C \in \mathbb{R}^{m\times m}$ its generalized version, the *generalized, continuous-time, algebraic Sylvester equation (GCASE)*, is given by

$$AXC - EXB + Q = 0. \tag{2.15}$$

Discrete-time, algebraic, Sylvester and *generalized, discrete-time, algebraic Sylvester equations (DASEs and GDASEs)* are of the form

$$AXB - X + Q = 0 \quad \text{and} \quad AXB - EXC + Q = 0. \tag{2.16}$$

The minus sign in front of the second summand in X is chosen for notation purposes.

Remark 2.32:

For Sylvester equations, the terminologies continuous- and discrete-time are not as stringent as for Lyapunov equations. Since GDASEs (2.16) are special cases of GCASEs (2.15) and vice versa, we drop the distinction between continuous- and discrete-time in the following and only speak of GCASEs if this does not lead to confusion. In fact, notice that also GCALEs and GDALEs are special cases of GCASEs (2.15). We have already mentioned in Remark 2.27 that GDASEs (2.16) are occasionally also referred to as Stein equations in the literature. ◊

The following lemma gives conditions for the existence of a unique solution of the Sylvester equations defined above. We restrict to the generalized case (2.15) since the other Sylvester equations are just special cases of (2.15) (and vice versa) such that existence properties for those can be deduced easily.

Lemma 2.33 (Existence of unique GCASE solutions [169],[68]):

The generalized Sylvester equation (2.15) has a unique solution if and only if $\Lambda(A,\ E) \cap \Lambda(B,\ C) = \emptyset$. If both or one of the matrices $E,\ C$ is singular, then one has to additionally assume that the corresponding pairs $(A,\ E)$, $(B,\ C)$ are regular. If all matrices A, B, E, C are nonsingular, $\Lambda(A,\ E) \subset \mathbb{C}_-$, and $\Lambda(B,\ C) \subset \mathbb{C}_+$, the solution can be represented as

$$X = \int_0^\infty \exp\left(E^{-1}At\right)E^{-1}FG^TC^{-1}\exp\left(-BC^{-1}t\right). \tag{2.17}$$

2.3.2. Nonlinear Matrix Equations of Riccati Type

In this subsection we describe a certain class of nonlinear matrix equations which are commonly known as algebraic Riccati equations (AREs). For this thesis, these are given by adding a certain quadratic or rational term in X to the linear matrix equation introduced in the previous section. We mention three classes of algebraic Riccati equation: continuous- and discrete-time (symmetric), as well as nonsymmetric Riccati equations. These equations and some selected theoretical properties are introduced in the following.

Symmetric Algebraic Riccati Equations

Definition 2.34 (Continuous-time algebraic Riccati equations):

The *(continuous-time) algebraic Riccati equation (CARE)* and its generalized version, the *generalized (continuous-time) algebraic Riccati equation (GCARE)*, are defined by

$$A^TX + XA - XRX + Q = 0 \quad \text{and} \tag{2.18}$$
$$A^TXE + E^TXA - E^TXRXE + Q = 0, \tag{2.19}$$

respectively, where $A,\ E,\ R = R^T,\ Q = Q^T \in \mathbb{R}^{n\times n}$ with E nonsingular and $X \in \mathbb{R}^{n\times n}$ is the sought solution. ◊

(G)CAREs of the above form occur frequently in optimal control problems of control systems of the form (2.5). Due to the nonlinear nature, (2.18), (2.19) might have no unique, but several solutions. In practice, one is usually interested in the c-stabilizing solution X_*.

Theorem 2.35 (Uniqueness of the c-stabilizing GCARE solution [55, 150]):

If $R,\ Q \succeq 0$, $(E;\ A,\ R,\ Q)$ c-stabilizable and c-detectable, there exists a unique, symmetric c-stabilizing solution X_* of the GCARE (2.19) such that $\Lambda(A - ERX_*,\ E) \subset \mathbb{C}_-$. If $(E;\ A,\ R,\ Q)$ is even observable, then $X_* \succ 0$. ◊

For the sake of completeness, we also mention discrete-time AREs which correspond to discrete-time, control systems (2.6).

Definition 2.36 (Discrete-time algebraic Riccati equations):
For $A,\ Q = Q^T \in \mathbb{R}^{n\times n},\ B,\ C^T \in \mathbb{R}^{n\times m}$, and $R = R^T \in \mathbb{R}^{m\times m}$, the rational matrix equation

$$A^T XA - X - (B^T XA + C)^T (R + B^T XB)^{-1}(B^T XA + C)^T + Q = 0 \tag{2.20}$$

is referred to as *(discrete-time) algebraic Riccati equation (DARE)*. Introducing a nonsingular $E \in \mathbb{R}^{n\times n}$,

$$A^T XA - E^T XE - (B^T XA + C)^T (R + B^T XB)^{-1}(B^T XA + C)^T + Q = 0 \tag{2.21}$$

is called *generalized discrete-time algebraic Riccati equation (GDARE)*. ◊

Usually, a d-stabilizing solution is of interest such that
$\Lambda\left(A - B(R + BX_*B^T)^{-1}(B^T X_* A + Q),\ E\right) \subset \mathbb{D}$. Uniqueness and existence conditions can be found, e.g., in [150, Theorem 13.1.1., Corollary 13.1.2.-3.].

Nonsymmetric Algebraic Riccati Equations

In the same way Sylvester equations generalize Lyapunov equations, we can find general variants of the symmetric Riccati equations above.

Definition 2.37 (Nonsymmetric algebraic Riccati equations):
A *nonsymmetric algebraic Riccati equation (NARE)* is defined by

$$AX + XB - XRX + Q = 0 \tag{2.22}$$

with $A \in \mathbb{R}^{n\times n},\ B \in \mathbb{R}^{m\times m},\ R \in \mathbb{R}^{m\times n},\ C \in \mathbb{R}^{n\times m}$, and $X \in \mathbb{R}^{n\times m}$ is the sought solution. If the matrix

$$\mathcal{M} := \begin{bmatrix} B & -R \\ Q & A \end{bmatrix} \in \mathbb{R}^{m+n\times m+n}$$

is an M-matrix, i.e., $\mathcal{M} = \sigma I_{m+n} - \mathcal{N}$ with $\mathcal{N} \geq 0$ and $\sigma \geq \rho(\mathcal{N})$, then (2.22) is abbreviated MNARE. ◊

Here, $\leq,\ \geq$ should be understood as the element wise partial orderings, i.e., $\mathcal{X} \geq \mathcal{Y}$ when all elements of $\mathcal{X} - \mathcal{Y}$ are greater or equal to zero. MNAREs originate in certain applications, e.g., in the study of fluid queue models [187] and transport equations [140]. One is typically interested in the minimal or maximal non-negative solutions $X_{\min},\ X_{\max} \geq 0$ which satisfy $X_{\min} \leq X,\ X_{\max} \geq X$ for all possible solutions X of (2.22). Conditions for the existence of these solutions are given in the next theorem which summarizes results from [55, Theorem 2.9, 2.13].

Theorem 2.38 (Existence of nonnegative solutions of MNAREs [55]):
If $\mathcal{M}$ associated with (2.22) is a nonsingular M-matrix or a singular, irreducible M-matrix, then there exists a nonnegative solution $X_{\min} \geq 0$. It holds $X_{\min} > 0$ if $\mathcal{M}$

is irreducible. Let, for a singular, irreducible M-matrix $\mathcal{M}$, u and v be the vectors satisfying $\mathcal{M}u = \mathcal{M}^T v = 0$ with $v^T u = 0$. If $\mu = 0$, where

$$\mu := v^T \left[\begin{smallmatrix} 0 & I_m \\ -I_n & 0 \end{smallmatrix}\right] u$$

is the so called *drift*, then $X_{\min}$ is unique. If $\mu \neq 0$ or $\mathcal{M}$ is nonsingular, there is a second nonnegative solution $X_{\max}$. ◊

2.3.3. Low-Rank Phenomena

For solving algebraic matrix equations of large dimensions, one of the most crucial properties exploited in several algorithms is the often observed very small numerical rank of the solution X, i.e., $\operatorname{rank}(X, \tau) \ll \min(n, m)$. It can be frequently seen that the singular values of the solution decay rapidly to zero. A main requirement for this effect is that the right hand side Q has a small (exact) rank $r = \operatorname{rank}(Q) \ll \min(n, m)$, but also other quantities of the defining coefficient matrices contribute to this decay. For Lyapunov and Sylvester equations, there are several theoretical results [158, 184, 213, 4, 113, 112, 222, 20, 7] that qualitatively describe the singular value decay of the solution X and how well it can be approximated by a rank k approximation $\tilde{X}_k$. We briefly mention two approaches: a quadrature based approach for GCASEs and an approach using a certain factorization of solutions of GCALEs. For the first one we need the following representation of the inverse of a c-stable matrix.

Lemma 2.39 (Integral representation of the inverse of a matrix [112]):

Let R be a square, Hurwitz matrix. Then

$$R^{-1} = -\int_0^\infty \exp(Rt)\mathrm{d}t. \qquad \text{◊}$$

The following theorem shows how well R^{-1} can be approximated by applying quadrature to the above integral.

Theorem 2.40 (Quadrature approximation of the inverse [112]):

Assume that $\Lambda(R) \subset -[2,\ u] \oplus \jmath[-c, c] \subset \mathbb{C}_-$ with $u,\ c \in \mathbb{R}_+$ and let Γ be the boundary of $\Psi := -[1,\ u+1] \oplus \jmath[-c-1, c+1] \subset \mathbb{C}_-$. For a $k \in \mathbb{N}$ define the quadrature weights and points

$$t_i := \ln\left(\exp(ih) + \sqrt{1 + \exp(2ih)}\right), \quad \omega_i := \frac{h}{\sqrt{1 + \exp(-2ih)}}$$

with $h := \pi/\sqrt{k}$ and $i = -k, \ldots, k$. Then there exists a problem independent constant $\phi > 0$ such that

$$\left\| \int_0^\infty \exp(Rt)\mathrm{d}t - \sum_{i=-k}^{k} \omega_i \exp(Rt_i) \right\| \leq \frac{\phi}{2\pi} \exp\left(\frac{1+c}{\pi} - \pi\sqrt{k}\right) \oint_\Gamma \|(\tau I_n - R)^{-1}\| \mathrm{d}_\Gamma \tau. \tag{2.23}$$

in any matrix norm. ◊

If R has only real eigenvalues, i.e, $c = 0$, the error expression (2.23) can be simplified further. The above theorem enables us to prove the existence of a low-rank solution $\tilde{X}$ of the generalized Sylvester equation (2.15). It is an adaption of more general results in [113, 20].

Corollary 2.41 (Low-rank solutions of Sylvester equations [113, 20]):

Consider the generalized Sylvester equation (2.15) defined by $A,\ E \in \mathbb{R}^{n\times n}$, $B,\ C \in \mathbb{R}^{m\times m}$, $E,\ C$ nonsingular, $Q = FG^T$ with $F \in \mathbb{R}^{n\times r}$, $G \in \mathbb{R}^{m\times r}$ having full column rank r. Assume that $\lambda - \mu \in \mathbb{C}_- \ \forall \lambda \in \Lambda(A,\ E),\ \mu \in \Lambda(B,\ C)$, and define $\gamma_{\min} := \min|\operatorname{Re}(\lambda-\mu)|$, $\gamma_{\max} := \max|\operatorname{Re}(\lambda-\mu)|$, and $d := \max|\operatorname{Im}(\lambda-\mu)|$. Let Γ be the boundary of the region Ψ from Theorem 2.40 with $u = \frac{2\gamma_{\min}}{\gamma_{\max}}$, $c = \frac{2d}{\gamma_{\min}}$. Then there exists an approximate solution $\tilde{X}$ of rank $(2k+1)r$ given by

$$\tilde{X} = -\sum_{i=-k}^{k} \tilde{\omega}_i \exp\left(\tilde{t}_i E^{-1}A\right) E^{-1}FG^TC^{-1} \exp\left(-\tilde{t}_i BC^{-1}\right), \tag{2.24}$$

where $\tilde{\omega}_i := \frac{2\omega_i}{\gamma_{\min}}$, $\tilde{t}_i := \frac{2t_i}{\gamma_{\min}}$ with the quadrature weights and points $\omega_i,\ t_i$ from Theorem 2.40. The corresponding error is

$$\begin{aligned} &\|X - \tilde{X}\|_{\mathrm{F}} \le \zeta_{\mathrm{Sylv}} \left\|E^{-1}FG^TC^{-1}\right\|_2 \\ &\text{with} \quad \zeta_{\mathrm{Sylv}} := \frac{\phi}{2\pi} \exp\left(\frac{1+2d\gamma_{\min}^{-1}}{\pi} - \pi\sqrt{k}\right) \oint_\Gamma \left\| \left(\frac{\tau\gamma_{\min}}{2} I_{nm} - \mathcal{A}\right)^{-1} \right\|_{\mathrm{F}} \mathrm{d}_\Gamma\tau \\ &\quad \text{and} \quad \mathcal{A} := I_n \otimes E^{-1}A - C^{-T}B^T \otimes I_m. \end{aligned} \tag{2.25}$$

Proof. The Sylvester equation (2.15) with $Q = FG^T$ can be rewritten to the equivalent linear system

$$\mathcal{A}\operatorname{vec}(X) = \operatorname{vec}\left(E^{-1}FG^TC^{-1}\right) = (C^{-T}G \otimes E^{-1}F)\operatorname{vec}(I_r) := \mathcal{Z}.$$

We multiply this linear system by $\frac{2}{\gamma_{\min}}$ since $\frac{2}{\gamma_{\min}}\mathcal{M}$ satisfies the conditions of Theorem 2.40. Applying the integral representation of Lemma 2.39 and the quadrature approximation of 2.40 yields

$$\operatorname{vec}(X) = -\frac{2}{\gamma_{\min}} \int_0^\infty \exp\left(\frac{2}{\gamma_{\min}}\mathcal{A}t\right)\mathrm{d}t\mathcal{Z} \approx -\sum_{i=-k}^{k} \tilde{\omega}_i \exp(\mathcal{A}\tilde{t}_i)\mathcal{Z} =: \operatorname{vec}\left(\tilde{X}\right).$$

Since $\exp(I_n \otimes M - N \otimes I_m) = \exp(-N) \otimes \exp(M)$ for $N \in \mathbb{R}^{n\times n},\ M \in \mathbb{R}^{m\times m}$ [132], this approximation can be rewritten as

$$\begin{aligned} \operatorname{vec}\left(\tilde{X}\right) &= -\sum_{i=-k}^{k} \tilde{\omega}_i \exp(-C^{-T}B^T\tilde{t}_i) \otimes \exp(E^{-1}A\tilde{t}_i)(C^{-T}G \otimes E^{-1}F)\operatorname{vec}(I_r) \\ &= -\sum_{i=-k}^{k} \tilde{\omega}_i \exp(-C^{-T}B^T\tilde{t}_i)C^{-T}G \otimes \exp(E^{-1}A\tilde{t}_i)E^{-1}F\operatorname{vec}(I_r) \end{aligned}$$

from which (2.24) follows by reversing the vectorization operator and applying Lemma 2.18a. The error expression is simply obtained by

$$\begin{aligned}\|X - \tilde{X}\|_{\mathrm{F}} &= \|\operatorname{vec}(X) - \operatorname{vec}\left(\tilde{X}\right)\|_F = \|(\mathcal{A}^{-1} - \tilde{\mathcal{A}}^{-1})\mathcal{Z}\|_F \\ &\leq \|(\mathcal{A}^{-1} - \tilde{\mathcal{A}}^{-1})\|_F \|E^{-1}FG^TC^{-1}\|_2\end{aligned}$$

and using (2.23) of Theorem 2.40. □

Remark 2.42:

The condition $\lambda - \mu \in \mathbb{C}_- \ \forall \lambda \in \Lambda(A,\ E),\ \mu \in \Lambda(B,\ C)$ can be relaxed by assuming that the spectra $\Lambda(A,\ E)$ and $\mu \in \Lambda(B,\ C)$ are separated by a line [113]. Then there exists a complex constant ν such that $\Lambda(\nu A,\ E)$ and $\mu \in \Lambda(\nu B,\ C)$ are separated by a vertical line parallel to the imaginary axis and, hence, $\hat{\lambda} - \hat{\mu} \in \mathbb{C}_- \ \forall \hat{\lambda} \in \Lambda(\nu A,\ E)$, $\hat{\mu} \in \Lambda(\nu B,\ C)$. One then applies the above theorem to the equivalent, but complex, GCASE $\nu A(\nu^{-1}X)C - E(\nu^{-1}X)(\nu B) = FG^T$. ◊

Remark 2.43:

In [20] an extension of the above result for certain symmetric, linear matrix equations of type (2.8) with $\ell > 2$ can be found. Applying a quadrature formula to the Dunford-Cauchy representation of the matrix exponentials in the solution formula (2.17) leads to more sophisticated results for Sylvester equations [113]. ◊

The error expression in Corollary 2.41 should not be understood as very accurate way to estimate the error of low-rank approximations. In practice, much smaller errors can be achieved by approximations with lower rank. The result of Corollary 2.41 is nevertheless interesting as it theoretically explains the often observed fast singular value decay of the solutions of Sylvester equations and the existence of low-rank approximations due to the almost exponentially decreasing character of the approximation error (2.25) which depends on $\exp(\sqrt{k})$.

Moreover, the constant ζ_{Sylv} indicates some quantities of the defining coefficient matrices which influence this decay: the rank r of the right hand side as well as the values $\gamma_{\min}$ and d. In particular, the ratio $\beta := d/\gamma_{\min}$ appears to be influential in (2.25). It determines a certain delay before the almost exponentially fast decrease of the approximation error takes place: only if k is large enough such that $\sqrt{k} > \frac{1+2\beta}{\pi^2}$, we have that $\exp\left(\frac{1+2\beta}{\pi} - \pi\sqrt{k}\right) < 1$. However, one should not conclude that r and β are the only important quantities and that the influence of r and β is directly visible. For instance, larger values of β might not necessarily lead to a slower singular value decay. The eigenvectors of $(A,\ E)$ and $(B,\ C)$ also effect (2.25), but their influence is somewhat hidden in the value of the integral and not easily read off.

For GCALEs, the low-rank approximation and error bound in Corollary 2.41 are simplified in a straightforward way by setting $C = -E^T$, $B = A^T$, and $G = F$. In that

case, the constant in the error expression (2.25) is

$$\begin{aligned} \zeta_{\text{Lyap}} &:= \frac{\phi}{\pi} \exp\left(\frac{1+2\tilde{\beta}}{\pi} - \pi\sqrt{k}\right) \oint_{\Gamma} \left\|(\tau\tilde{\gamma}_{\min} I_{n^2} - \mathcal{A})^{-1}\right\|_{\mathrm{F}} \mathrm{d}_{\Gamma}\tau \\ \text{with} \quad \tilde{\beta} &:= \tfrac{\max|\mathrm{Im}\,(\lambda)|}{\min|\mathrm{Re}\,(\lambda)|},\ \tilde{\gamma}_{\min} := \min|\mathrm{Re}\ (\lambda)|, \quad \lambda \in \Lambda(A,\ E). \end{aligned} \tag{2.26}$$

The scalar $\tilde{\beta}$ is closely related to the *opening angle* of $\Lambda(A,\ E) \subset \mathbb{C}_-$:

$$\psi(A,\ E) = \max|\arg\lambda| = \max \arctan\left|\tfrac{\mathrm{Im}\,(\lambda)}{\mathrm{Re}\,(\lambda)}\right|. \tag{2.27}$$

Hence, if there are eigenvalues close to the imaginary axis but with imaginary parts which dominate the real parts, we expect that this slows down the singular value decay of the GCALE solution. The opening angle of the spectrum $\psi(A,\ E)$ will also play a role in other topics later.

For GCALEs there are alternatives to this quadrature based approach for quantifying the singular value decay [184, 213, 4, 242, 222]. Since the sought solutions are positive semidefinite, one can equivalently look at the decay of the eigenvalues in an non-increasing order. As example, we mention a result from [4]. Consider a CALE (2.10) defined by a diagonalizable matrix A, $E = I_n$ and $Q = ff^T$, $f \in \mathbb{R}^n$. Then the solution X can be expressed as

$$X = W_f K W_f^H, \quad W_f := W \operatorname{diag}\left(W^{-1}f\right), \tag{2.28}$$

where W is the matrix containing the right eigenvectors of A as columns, i.e., $W^{-1}AW = \operatorname{diag}(\lambda_1, \ldots, \lambda_n)$, $\lambda_i \in \Lambda(A)$. The matrix K is a Hermitian positive (semi)definite Cauchy matrix $K = \left(\frac{-1}{\lambda_i + \overline{\lambda_j}}\right)_{i,j=1}^{n}$. It is proven in [4, Theorem 3.1] that there exist rank-k approximations of X satisfying

$$\begin{aligned} \|X - X_k\| &\leq \delta_{k+1}((n-k)\kappa(W)\|f\|)^2 \\ \text{with} \quad \delta_{k+1} &= \frac{1}{-2\,\mathrm{Re}\ (\lambda_{k+1})} \prod_{j=1}^{k} \left|\frac{\lambda_{k+1} - \lambda_j}{\overline{\lambda_{k+1}} + \lambda_j}\right|^2. \end{aligned} \tag{2.29}$$

The δ's are related to the Cholesky factorization of K [108] and can, under a certain ordering of the eigenvalues of A, be ordered in a non-increasing form $\delta_1 \geq \delta_2 \ldots \geq \delta_n \geq 0$. Furthermore, every factor in the product defining δ_{k+1} is smaller than one and, thus, the error decreases with increasing k. Moreover, the δ's often decay very fast towards zero and if A is not too far from normal, this decay mimics the singular value decay of X. Hence, (2.29) can, to some extent, also be used as explanation for the fast singular value decay of CALE solutions. In Section 7.3.1, we will use a similar approach to investigate the singular value decay of certain special CALEs related to model order reduction. A generalization of the above results to inhomogeneities $Q = FF^T$, $F = [f_1, \ldots, f_r] \in \mathbb{R}^{n\times r}$ is given by

$$\begin{aligned} X &= \sum_{i=1}^{r} W_{f_i} K W_{f_i}^H, \quad W_{f_i} := W \operatorname{diag}\left(W^{-1}f_i\right), \\ \|X - X_{rk}\| &\leq \delta_1 \eta r((n-k)^2\kappa(W)\|B\|)^2, \end{aligned} \tag{2.30}$$

for an approximation X_{rk} of rank rk and $\eta > \delta_{k+1}/\delta_1$, see [4, Theorem 3.2]. In [222], an improved error bound is proposed which also considers certain non-diagonalizable matrices. For GCALEs, similar relations can be found by using $W_f = W \operatorname{diag}(W^{-1}E^{-1}f)$ in (2.28). In [108], an expression analog to (2.28) can be found for GCASEs.

The expression in (2.28) reveals an interesting relation between f and the inverse eigenvector matrix which is not directly reflected in the prior approximation result (2.29). Since the matrix of left eigenvectors is $Y = W^{-H}$, the magnitudes of the entries $y_i^H f$, $i = 1, \ldots, n$, of the vector $\tilde{f} := \operatorname{diag}\left(Y^H f\right)$ influences W_f. This has, e.g., also been discussed in [222], where the authors conclude that, next to the eigenvalues of K, the vector $\tilde{f}$ has a significant impact on the eigenvalue decay rate of the solution. In an extreme case, where $(A,\ f)$ is not completely controllable, $y_i^H f = 0$ for some i and, thus, W_f will be singular which directly influences the exact rank of X. The argumentation for the case $r > 1$ is similar and can be deduced from (2.30). Due to the appearance of $\kappa(W)$ in (2.29), the above approximation result ceases to be useful for highly non-normal matrices as it is, e.g., investigated in [7]. More precise effects of the eigenvectors on the eigen- or singular value decay rates are, to the author's knowledge, not available and are subject to current research. Some discussions regarding the impact of non-normality on the eigenvalue decay can be found, e.g., in [202, 7].

It is important to point out that both the above approaches suggest by (2.24) and (2.30) that the rank of the approximation, to achieve a certain accuracy, increases with r. This, in turn, means that the singular value decay decreases as r increases. While this can indeed be observed in many cases, the conjecture is in general not true. In the upcoming short experiment, but also in Chapter 7.3, we will encounter situations where an inhomogeneity of higher rank does not at all lead to a deceleration of the singular value decay rate. Again, more thorough insights into the interaction of the eigenvectors and F are required to explain these effects.

Despite the various results for linear matrix equations, less research has been done regarding theoretical results for the singular, and eigenvalue decay as well as low-rank approximations of solutions of AREs. In [23], the authors address this issue for CAREs by using similar techniques based on Cauchy matrices as the ones for CALEs above.

We end this section with a small CALE example to illustrate some of the mentioned aspects of the low-rank phenomenon.

We use CALEs

$$A_i X_i + X_i A_i^T + Q_i, \quad Q_i = F_i F_i^T \tag{2.31}$$

defined by different matrices A_i and inhomogeneities $Q_i = F_i F_i^T$ which are summarized, together with the values β and r, in Table 2.1. The CALEs for five different pairings of A_i, F_i are solved by the MATLAB routine `lyap`. Figure 2.1 shows the scaled singular values $\sigma_j(X)/\sigma_1(X)$ which were computed using the `svd` command. Apparently, for all five test cases, the scaled singular values drop below the machine precision $\mathbf{u}_{\text{mach}}$ before j reaches n. With the same reasoning as in the example in the introduction, the values below $\mathbf{u}_{\text{mach}}$ should by no means be trusted and should not be considered as accurate singular values. The slower decay of these computed singular values once they fall below

Table 2.1.: Matrices defining the CALEs (2.31)

Matrix A	β	Factor F of inhomogeneity	r
$A_1 = \mathrm{diag}(-[1, 1, \ldots, 50, 50])$	0	$F_1 \in \mathbb{R}^n$ random	1
$A_2 = A_1 + I_{50} \otimes \left[\begin{smallmatrix} 0 & -10 \\ 10 & 0 \end{smallmatrix}\right]$	10	$F_2 = [F_1, c_2]$, $c_2 \in \mathbb{R}^{n\times 2}$ random	3
$A_3 = A_2 + \frac{9}{10} I_n$	100	$F_3 = [F_1,\ c_3 + [e_2, \ldots, e_{10}]]$, $c_3 \in \mathbb{R}^{n\times 9}$, $\|c_3\| \approx 10^{-7}$	10

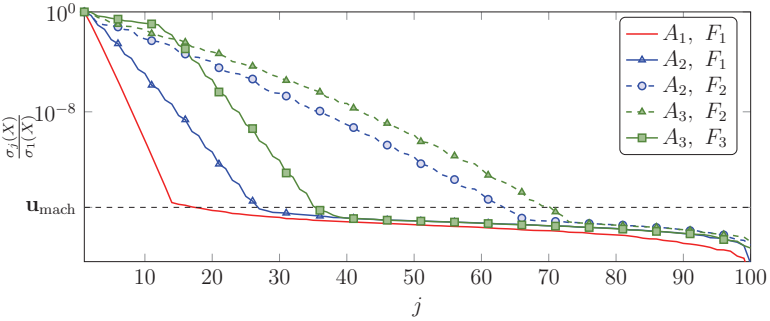


Figure 2.1.: Scaled singular values (σ_j/σ_1) of the solutions of the different CALEs.

$\mathbf{u}_{\mathrm{mach}}$ might possibly only be a numerical artifact caused by the difficulties to accurately compute quantities smaller or within the range of the machine precision. More robust singular value algorithms, e.g., [80], might reveal that even below $\mathbf{u}_{\mathrm{mach}}$ the singular values keep decaying at a similar rate as before. The plot in Figure 2.1 nevertheless shows a different fast singular value decay for the different settings. The fastest decay is achieved with the diagonal matrix A_1 and the inhomogeneity $F_1F_1^T$ of rank one. In comparison, using the non-symmetric matrix A_2 instead, yields a considerably slower singular value decay since the complex eigenvalues lead to $\tilde{\beta}(A_2) = 10 > 0 = \tilde{\beta}(A_1)$. Replacing $F_1F_1^T$ by $F_2F_2^T$ increases the rank of the inhomogeneity to $r = 3$ and clearly slows down the singular value decay further. Shifting A_2 closer to the imaginary axis gives A_3 with an increased $\tilde{\beta}_3 = 100$ which yields a further deceleration of the decay. So far, these observations are in line with the observations regarding the error expression (2.25) of Corollary 2.41. With the inhomogeneity $F_3F_3^T$ with $r = 10$ one would expect from (2.25) and (2.30) an even slower singular value decay. However, the decay is significantly faster than with $F_2F_2^T$ of rank three. This suggests that the solution of the CALE can be more easily approximated by a low-rank matrix than for the CALE defined by A_3, $F_2F_2^T$. An explanation for this is difficult to find with Corollary 2.41, but the expressions (2.28), (2.30) are more insightful. Apparently, the additional columns of F_3 are nearly orthogonal to some left eigenvectors of A_3. Hence, some of the corresponding W_{f_i} in (2.30) are nearly singular which obviously influences the singular values and, thus, also the numerical rank of the CALE solution.

2.3.4. Concise Overview of Full / Low-Rank Methods

Instead of giving an exhaustive summary of algorithms for solving matrix equations, we mention selected approaches, including the ones important for comparative studies in this thesis. Further methods, as well as more details, can be found, e.g., in the survey articles [210, 52]. Whenever we give lists of references in the following short summary, these are most likely not complete but selective. Therefore, these lists should always be understood as directions towards the cited literature as well as the references therein.

For small to medium scale linear matrix equations of length $\ell = 2$, methods based on (generalized) Schur, eigenvalue, or Hessenberg decompositions of the involved coefficient matrices can be applied, such as the popular Bartels-Stewart algorithm [14] for Sylvester and Hammerling's method [124] for Lyapunov equations, where the latter one directly computes a Cholesky factor of the solution. Other related methods can be found in, e.g., [214, 109, 90, 105, 68]. Another class of methods based on the matrix sign function iteration was investigated in, e.g., [133, 46, 17]. The ADI (alternating direction implicit) iteration is another iterative scheme that can be applied to solve CALEs and CASEs [229, 233]. This approach will be reviewed in the next chapter as its represents the backbone of the methods discussed in this thesis. For matrix equations with $\ell > 2$, some iterative methods which solve an equation with $\ell = 2$ in each iteration step by using one of the just mentioned methods, can be found in, e.g., [71, 20, 21]. The solution of algebraic Riccati equations can be built from certain invariant subspaces of the associated Hamiltonian (CAREs), symplectic (DAREs), or M-matrices (MNAREs). We refer to [150, 55] for overviews of these concepts. Algorithms which employ these properties by computing certain decompositions of the involved block matrices can be found in, for instance, [154, 155, 121, 55, 63]. The application of the sign function iteration for solving GCAREs and GDAREs is considered in [191, 106, 144, 143]. A generalization of the ADI iteration for CAREs, the quadratic ADI iteration (QADI), is proposed in [238]. Due to the nonlinear nature of AREs, Newton schemes present another well suited approach for their solution [145, 126, 122]. There, in each Newton step a linear matrix equation has to be solved. All of these methods involve computations with and manipulations of the original coefficient matrices, such as constructing the factorizations mentioned before. This yields a cubic complexity and quadratic storage requirements and, hence, they are not feasible for large and sparse matrices. Furthermore, they do not exploit the often present low numerical rank of the solution. There are, however, specialized methods for GCASEs when only one of the pairs $(A,\ E)$ or $(B,\ C)$ is large and sparse, but the other one is much smaller and dense [215, 30].

Algorithms for handling large matrix equations defined by sparse matrices are usually based on the assumption that matrix-vector products, as well as solutions of (shifted) linear systems with the coefficient matrices, can be efficiently computed in contrast to, e.g., Schur decompositions. As outlined before, one typically computes a low-rank approximation $\tilde{X}$ of the solution X, such that $\operatorname{rank}\{\tilde{X}\} = k \ll \min(n, m)$. This approach is, of course, only valid if the problem at hand actually admits such a low-rank solution, e.g., when $\operatorname{rank}\{FG^T\} = r \ll \min(n, m)$ and β is sufficiently small which we assume in the following. Typically, this low-rank solution is computed in a factorized form

$\tilde{X} = ZDY^T$ with $Z \in \mathbb{R}^{n\times k}$, $D \in \mathbb{R}^{k\times k}$, and $Y \in \mathbb{R}^{m\times k}$. For equations having positive (semi)definite solutions (GCALEs, GCAREs, as well as their discrete-time versions), this can be simplified to $\tilde{X} = ZZ^T$ or $\tilde{X} = ZDZ^T$ with $D = D^T \succeq 0$.

A large class of methods for this purpose is built upon projections onto low-dimensional subspaces. We briefly outline the main idea for CALEs. Given a subspace $\mathcal{U} = \operatorname{span}\{U\} \subset \mathbb{C}^n$ with $U \in \mathbb{C}^{n\times k}$, $U^HU = I_k$, $k \ll n$, the solution X is approximated by $\tilde{X} = UDU^H$. Imposing a Galerkin condition [196] onto the residual

$$\mathcal{L} = A(UDU^H) + (UDU^H)A^T + FF^T,$$

leads to $U^HAUD + DU^HA^HU + U^HFF^TU = 0$, i.e., D is the solution of a small, k-dimensional CALE which can be solved by the algorithms mentioned above. Usually, one produces sequences of subspaces of increasing dimensions in an iterative manner. Different algorithms mainly differ in the choice of the subspace $\mathcal{U}$. For (G)CALEs, standard Krylov subspaces $\mathcal{K}_k(A,\ F) = \operatorname{span}\left\{F, AF, \ldots, A^{k-1}F\right\}$ have been employed in [196, 137]. The extended Krylov subspace method (EKSM) [209, 218] uses extended Krylov subspaces $\mathcal{EK}_k(A,\ F) = \mathcal{K}_k(A,\ F) \cup \mathcal{K}_k(A^{-1},\ A^{-1}F)$ which leads to a significantly faster convergence compared to using $\mathcal{K}_k(A,\ F)$ alone. A further generalization is the rational Krylov subspace method (RKSM) [83, 82, 84] which works with subspaces $\mathcal{K}_k^{\text{rat}}(A,\ F,\ \xi) = d_{k-1}(A)^{-1}\mathcal{K}_k(A,\ F)$, where d_{k-1} is polynomial whose roots are given real or complex parameters $\xi = \{\xi_1, \ldots, \xi_k\}$. An illustration of RKSM is given in Algorithm A.1 in the appendix. Several modifications and generalizations of similar methods are available for GDALEs [203], GCASEs [89, 11, 12, 138, 134, 60], and AREs [127, 211, 167]. A related approach, based in some sense on $\mathcal{H}_2$ interpolation and using also rational Krylov subspaces, is proposed in [21].

The algorithms investigated in this thesis are low-rank versions of the aforementioned ADI iteration which, for GCALEs, were extensively studied in [183, 161, 18, 42, 201]. These methods can be carried over to GDALEs [27, 203] and GCASEs [162, 43, 32]. In the chapters to come, we will review and improve these methods regarding various computational aspects. There, we will also see that the ADI iteration can be understood as functional iteration, where the next iterate X_k is obtained via $X_k = f(X_{k-1})$ for $k \geq 1$. The Smith method [29, 165] and, in particular, its low-rank variations [183, 202] also fit into this framework. Solving the occurring linear matrix equations in Newton methods for GCAREs and GDAREs by the low-rank ADI iteration is considered in [42, 49, 201, 50, 27, 94, 10, 125]. In Chapter 6, we will discuss a similar method to deal with large-scale NAREs. Low-rank versions of the QADI iteration for GCAREs are investigated in [238, 201, 23, 24]. The Riccati-ADI iteration [170] is a further variant of the ADI iteration directly applicable to GCAREs.

There are several alternatives to low-rank Krylov and ADI type methods. For general linear matrix equations (2.8) with $\ell \geq 2$, one can apply iterative methods in a matrix-valued fashion. In fact, any available Krylov subspace method for linear systems [199, 225] can be used as in [59, 20]. It appears that this approach is particularly well suited for the situation $\ell > 2$ when an appropriate preconditioner is employed. For AREs, there exist further methods that iteratively exploit the relation of certain invariant subspaces

of the associated block matrices and the solution. This includes the subspace methods discussed in [211, 23] and doubling algorithms [70, 69, 56, 55].

2.4. Used Software, Hardware, and Test Examples

Throughout this thesis we will evaluate the considered algorithms by several numerical experiments which are all carried out in MATLAB 8.0.0.783 (R2012b) on a Intel® Xeon® CPU X5650 @ 2.67GHz with 48 GB RAM. This is one computing node of the linux cluster `otto`[1] at the *Max Planck Institute for Dynamics of Complex Technical Systems* in Magdeburg. It should be mentioned that the majority of numerical experiments could also be exercised on lesser hardware, e.g., smaller amounts of memory. This holds especially for the numerical methods, where the enhancements proposed in this thesis are included.

Here, we briefly introduce a number of test examples which are used several times in the upcoming numerical experiments. Often, these examples are related to a dynamical system of the form (2.5). A few further examples will be introduced in the upcoming sections on occasion for single use.

An Ocean Circulation Problem

A finite element model of an ocean circulation problem [227] yields a nonsymmetric Hurwitz matrix A with $n = 42,249$ which we use to define a CALE (2.10). If not stated otherwise, the factor F of the inhomogeneity $Q = FF^T$ is constructed by $r = 20$ columns with uniformly distributed random entries. We will abbreviate this example by *ocean*.

Scalable Finite Difference Discretizations of a Partial Differential Equation

An often used academic example for CALEs is obtained by using centered finite differences to discretize the partial differential equation

$$\dot{v} = -\Delta v - f_1 \frac{\partial v}{\partial \xi_1} - f_2 \frac{\partial v}{\partial \xi_2} - f_3 = 0$$

for $v = v(\xi_1,\ \xi_2)$ defined on $\Omega = (0,1)^2$ with homogeneous Dirichlet boundary conditions. Here, f_i, $i = 1, 2, 3$, are functions depending on $\xi_1,\ \xi_2$ and are often referred to as convection and reaction terms. Using n_0 equidistant grid points for each spatial dimension yields a dimension $n = n_0^2$ for A. The factor $F \in \mathbb{R}^{n\times r}$ of the inhomogeneity is usually taken as random matrix. The terms $f_1,\ f_2$ influence the symmetry of A. By varying f_i, n_0, and r, we will employ different versions of this example which is abbreviated by *FDM*. The particularly selected settings are mentioned at each occurrence in numerical experiments. A frequently used choice is $f_1 = 10^2\xi_1$, $f_2 = 10^3\xi_2$, and $f_3 = 0$ as in, e.g., [184].

[1] See `http://www.mpi-magdeburg.mpg.de/1012477/otto` for more information.

Cooling Process of Steel Rails

The steel profile cooling models are part of the Oberwolfach Model Reduction Benchmark Collection[2]. They represent spatial finite element discretizations of a heat transfer problem arising in the cooling of steel rail profiles [200, 47, 48]. Different grid sizes result in matrices $A \prec 0$, $E \succ 0$ of different dimensions which we use to define GCALEs (2.11). We use the cases with $n = 20,209$ and $n = 79,841$, which are abbreviated by *rail20k* and *rail79k*, respectively. The model constitutes a dynamical system (2.5) and also provides an input matrix B with $r = 7$ which serves as F. For the parts of this thesis devoted to model order reduction, the output matrix $C \in \mathbb{R}^{6\times n}$ in (2.5) is also used.

A Finite Element Discretization of a Convection-Diffusion Problem

The IFISS 3.2[3] package [207] provides the example T-CD3, where a time-dependent convection diffusion equation defined on $(0,1)^2$ is discretized by Q1 finite elements on a uniform grid. We use the default settings for this example, but different grid sizes to generate A, E: 64×64, 128×128, and 256×256 grids leading to the dimensions $n = 4,225$, $n = 16,641$, and $n = 66,049$. The corresponding abbreviations are *ifiss4k*, *ifiss16k* and *ifiss66k*. The matrix F is typically constructed by $r = 5$ random columns.

The Triple Chain Oscillator

In several applications, linear control systems of the form

$$\begin{aligned} M\ddot{q}(t) + D\dot{q}(t) + Kx(t) &= B_1 u(t), \\ y(t) &= C^{\mathrm{p}} q(t) + C^{\mathrm{v}} \dot{q}(t) \end{aligned} \tag{2.32}$$

with $M,\ D,\ K \in \mathbb{R}^{n\times n}$, $B_1 \in \mathbb{R}^{n\times r}$, and $C^{\mathrm{p}},\ C^{\mathrm{v}} \in \mathbb{C}^{p\times n}$, arise. The functions $q(t),\ u(t),\ y(t)$ are defined as in (2.5). Since (2.32) involves the second derivative w.r.t. time t, such systems are usually referred to as second order, linear, time-invariant, control systems. They are often dealt with by rewriting (2.32) formally into an equivalent generalized state space system (2.5) of first order with $E,\ A \in \mathbb{R}^{2n\times 2n}$, $F \in \mathbb{R}^{2n\times r}$, $L \in \mathbb{R}^{p\times 2n}$ and the augmented generalized state vector $x(t) = [q(t)^T, \dot{q}(t)^T]^T$. This is closely related to rewriting the associated quadratic matrix polynomial $\lambda^2 M + \lambda D + K$ into a linear pencil $\lambda E - A$ [220]. Hence, we will refer to this transformation as *linearization*. The two most standard linearizations are the *first companion linearization*

$$E_1 = \begin{bmatrix} N_1 & 0 \\ 0 & M \end{bmatrix},\ A_1 = \begin{bmatrix} 0 & N_1 \\ -K & -D \end{bmatrix},\ F_1 = \begin{bmatrix} 0 \\ B_1 \end{bmatrix},\ L_1 = \begin{bmatrix} C^{\mathrm{p}} & C^{\mathrm{v}} \end{bmatrix} \tag{2.33a}$$

and the *second companion linearization*

$$E_2 = \begin{bmatrix} D & M \\ N_2 & 0 \end{bmatrix},\ A_2 = \begin{bmatrix} -K & 0 \\ 0 & N_2 \end{bmatrix},\ F_2 = \begin{bmatrix} B_1 \\ 0 \end{bmatrix},\ L_2 = \begin{bmatrix} C^{\mathrm{p}} & C^{\mathrm{v}} \end{bmatrix}, \tag{2.33b}$$

[2] Available at `http://portal.uni-freiburg.de/imteksimulation/downloads/benchmark` under the ID=38881.

[3] See `http://www.maths.manchester.ac.uk/~djs/ifiss/`.

where N_1, N_2 are arbitrary nonsingular $n \times n$ matrices. Common choices for those are $N_1 = I_n$ or $N_1 = -K$ in (2.33a) and $N_2 = I_n$ or $N_2 = M$ in (2.33b), respectively.

The scalable triple chain oscillator [223], abbreviated by *chain*, describes three coupled chains of masses interlinked with springs and dampers. Its equations of motion can be formulated in the form (2.32). Using $n_0 = 7,000$ masses yields matrices $M \succ 0$ and $K \succ 0$ of dimension $n = 3n_0 + 1 = 21,001$. The matrix $D \succ 0$ is modeled as $D = 0.02M + 0.5K$ with $\nu = 5$ added to the first, n-th, and $2n + 1$-th diagonal entry, and B_1 is a random matrix of dimension $n \times 5$.

The Brazilian Interconnected Power System Examples

The Brazilian interconnected power system models[4] provide a number of differential-algebraic systems (2.5) defined by the block structured matrices

$$E = \begin{bmatrix} E_{11} & 0 \\ 0 & 0 \end{bmatrix}, \; A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \in \mathbb{R}^{n \times n}, \; F = \begin{bmatrix} F_1 \\ F_2 \end{bmatrix} \in \mathbb{R}^{n \times r} \tag{2.34}$$

with $E_{11} \in \mathbb{R}^{n_f \times n_f}$ and $A_{22} \in \mathbb{R}^{(n-n_f)\times(n-n_f)}$ nonsingular. All the other subordinate block matrices have appropriate dimensions. The number $n_f < n$ refers to the number of finite eigenvalues in $\Lambda(A, \; E)$, which are supposed to be located in $\mathbb{C}_-$. The special structure of E implies that the pair $(A, \; E)$ is of index one [148]. Since the inverse of E does not exist, the formulation of an associated GCALE is not as straightforward as before. In [98] a GCALE associated to (2.34) is defined by

$$\begin{aligned} &\hat{A}\hat{P}E_{11} + E_{11}\hat{P}\hat{A}^T = -\hat{F}\hat{F}^T \\ \text{with} \quad &\hat{A} := A_{11} - A_{12}A_{22}^{-1}A_{21} \in \mathbb{R}^{n_f \times n_f}, \; \hat{F} := F_1 - A_{12}A_{22}^{-1}F_2 \in \mathbb{R}^{n_f \times r} \end{aligned} \tag{2.35}$$

which we employ for our studies. By the abbreviation *bips* we refer to the system bips07_3078 with $n = 21,128$, $n_f = 3,078$, $E_{11} = I_{n_f}$, and $r = 4$. As in [98, Section V.A] the shifted matrix $A - 0.05E$ is used since the original pair $(A, \; E)$ has finite eigenvalues very close to the imaginary axis.

[4] Available at `http://sites.google.com/site/rommes/software`.

CHAPTER 3

THE LOW-RANK ALTERNATING DIRECTIONS IMPLICIT ITERATION FOR LYAPUNOV AND SYLVESTER EQUATIONS

Contents

This chapter introduces the most important algorithm of this thesis, the low-rank alternating direction implicit (LR-ADI) iteration for computing low-rank solution factors of large-scale Lyapunov and Sylvester equations. A short summary of the historical origin of ADI methods is given in the next section. The subsequent two sections treat the application of the ADI iteration for Lyapunov and Sylvester equations separately. The low-rank ADI iteration for standard and generalized Lyapunov equations is derived in Subsection 3.2.1 which is followed by a brief overview of the shift parameter problem

and suitable termination criteria in Subsections 3.2.2 and 3.2.3, respectively. In Subsection 3.2.4, we investigate the structure of the Lyapunov residual in the ADI iteration and develop a novel formulation of the low-rank ADI iteration, where the residual norm can be computed much more efficiently compared to other approaches. Some specially tailored versions of this reformulated ADI iteration for certain structured problems are given in Subsection 3.2.5, and 3.2.6 presents some numerical experiments. Afterwards, similar investigations are carried out for generalized Sylvester equations in Section 3.3, where also special cases, e.g., discrete-time Lyapunov equations, are considered. After numerical examples illustrating the performance of the proposed methods, Section 3.4 summarizes and gives some further research perspectives.

3.1. The Origin of the Alternating Directions Implicit Iteration

The original ADI iteration was developed in [181] for solving linear systems

$$Au = b$$

with $A \in \mathbb{R}^{n\times n}$ symmetric, positive definite, arising in the numerical solution of elliptic and parabolic differential equations. Assume A can be expressed as $A = H + V$ with symmetric, positive definite matrices $H,\ V \in \mathbb{R}^{n\times n}$, such that the above linear system can be equivalently stated as

$$Hu + Vu = b.$$

For instance, if A represents a centered finite difference discretization of a partial differential equation in two dimensions, then H and V can be chosen as centered finite difference discretizations with respect to the x and y direction, respectively. The ADI iteration is for $j = 1, 2, \ldots$ defined in terms of double steps

$$\begin{aligned}(H + \alpha_j I_n)u_{j-\frac{1}{2}} &= (\alpha_j I_n - V)u_{j-1} + b,\\ (V + \alpha_j I_n)u_j &= (\alpha_j I_n - H)u_{j-\frac{1}{2}} + b,\end{aligned} \tag{3.1}$$

where $\alpha_j \in \mathbb{R}_+$ are appropriately chosen parameters. Provided that H and V commute ($HV = VH$), there exist shift parameters such that the ADI iteration above converges with a superlinear convergence rate [15].

3.2. The Low-Rank ADI Iteration for Large-Scale Continuous-Time Lyapunov Equations

3.2.1. The Application of the ADI Iteration to Lyapunov Equations

The standard CALE (2.10)

$$AX + XA^T = -FF^T, \quad F \in \mathbb{R}^{n\times r} \tag{3.2}$$

with a c-stable A represents an ADI model problem since the operator $L : X \mapsto AX + XA^T$ is a sum of the commuting operators $L_L : X \mapsto AX$ and $L_R : X \mapsto XA^T$. The original approach to derive an ADI scheme for (3.2) is to formally apply the iteration (3.1) to $L_L,\ L_R$ which gives

$$\begin{aligned}(A + \alpha_j I_n)X_{j-\frac{1}{2}} &= -FF^T - X_{j-1}(A^H - \alpha_j I_n),\\ (A + \alpha_j I_n)X_j^H &= -FF^T - X_{j-\frac{1}{2}}^H(A^H - \alpha_j I_n),\end{aligned} \tag{3.3}$$

where the shift parameters $\alpha_j \in \mathbb{C}_-$ are allowed to be complex now, and the iteration is started with an initial guess $X_0 = X_0^T \in \mathbb{R}^{n\times n}$. Rewriting (3.3) into a single step yields

$$\begin{aligned}X_j &= (A + \alpha_j I_n)^{-1}(A - \overline{\alpha_j} I_n)X_{j-1}(A - \overline{\alpha_j} I_n)^H(A + \alpha_j I_n)^{-H}\\ &\quad - 2\,\mathrm{Re}\,(\alpha_j)\,(A + \alpha_j I_n)^{-1}FF^T(A + \alpha_j I_n)^{-H}\\ &= \mathcal{C}(A, \alpha_j)X_{j-1}\mathcal{C}(A, \alpha_j)^H + \mathcal{Q}(\alpha_j)\end{aligned} \tag{3.4}$$

with the Cayley transformation (see Definition 2.15)

$$\mathcal{C}(A, \alpha) := (A + \alpha I_n)^{-1}(A - \overline{\alpha} I_n) \quad \text{and} \tag{3.5a}$$

$$\mathcal{Q}(\alpha) := -2\,\mathrm{Re}\,(\alpha)\,(A + \alpha I_n)^{-1}FF^T(A + \alpha I_n)^{-H}. \tag{3.5b}$$

We will refer to the iterative scheme (3.4) as ADI iteration for CALEs. There, the X_j are always Hermitian matrices although the iterates $X_{j-\frac{1}{2}}$ in the double-step iteration (3.3) are not. The restriction $\alpha_j \in \mathbb{C}_-$ leads to $\rho(\mathcal{C}(A, \alpha_j)) < 1$ by Proposition 2.16c.

Before we continue, we show an alternative insightful way to derive the ADI iteration (3.4) on the ground of the following important lemma.

Lemma 3.1 ([212, 183, 119]):

For all $\alpha \notin \Lambda(A)$ the CALE (3.2) is equivalent to the DALE (Definition 2.26)

$$X = \mathcal{C}(A, \alpha)X\mathcal{C}(A, \alpha)^H + \mathcal{Q}(\alpha) \tag{3.6}$$

with $\mathcal{C}(A, \alpha)$ and $\mathcal{Q}(\alpha)$ as in (3.5). ◊

The equivalent DALE (3.6) motivates the functional iteration

$$X_j = \mathcal{C}(A, \alpha)X_{j-1}\mathcal{C}(A, \alpha)^H + \mathcal{Q}(\alpha) \tag{3.7}$$

for $j = 1, 2, \ldots$ which is essentially the Smith iteration [212]. Varying also the shifts α, i.e., a different α_j is used in (3.7) for each j, immediately yields the ADI iteration (3.4). As discussed in [119], the iteration (3.7) (with or without varying shifts) can also be interpreted as transforming a continuous-time dynamical system (2.5) into a discrete-time dynamical system (2.6).

The Low-Rank ADI Iteration The ADI iteration (3.4) operates on $n \times n$ matrices and does not exploit the low-rank structure of the right hand side FF^T in (3.2). Hence, it is not suitable for computing low-rank solution factors of large-scale equations. For deriving a low-rank formulation of (3.4), the following lemma will be useful.

Lemma 3.2:

For all $M \in \mathbb{C}^{n\times n}$ and $\alpha,\ \beta \in \mathbb{C}$, the matrices $(M \pm \alpha I_n)^{\pm 1}$ and $(M \pm \beta I_n)^{\pm 1}$ commute, provided the inverses exist. ◊

If we introduce the low-rank factorization $X_j = Z_j Z_j^H$ and set $Z_0 = 0$, then (3.4) accumulates the low-rank factor Z_j via

$$Z_1 = \sqrt{-2\,\mathrm{Re}\ (\alpha_1)}(A + \alpha_1 I_n)^{-1} F, \tag{3.8}$$

$$Z_j = \left[\sqrt{-2\,\mathrm{Re}\ (\alpha_j)}(A + \alpha_j I_n)^{-1} F,\quad \mathcal{C}(A, \alpha_j) Z_{j-1}\right], \tag{3.9}$$

where we exploited the symmetric structure of $\mathcal{Q}(\alpha_j)$. In this way, rj columns have to be processed at iteration j, i.e., j shifted linear system with r right hand sides each have to be solved, such that the iteration gets increasingly expensive. Introducing the notations

$$\gamma_j := \sqrt{-2\,\mathrm{Re}\ (\alpha_j)},\quad \mathcal{F}_j := (A + \alpha_j I_n)^{-1} F,$$
$$\mathcal{C}_j := \mathcal{C}(A, \alpha_j),\quad \text{and}\quad \mathcal{C}_{i,j} := \mathcal{C}(A, \alpha_i, \alpha_j) = (A - \overline{\alpha_i} I_n)(A + \alpha_j I_n)^{-1},$$

the low-rank factor Z_j in the above scheme can be written as

$$Z_j = \left[\gamma_j \mathcal{F}_j,\ \gamma_{j-1}\mathcal{C}_j\mathcal{F}_{j-1},\ \gamma_{j-2}\mathcal{C}_j\mathcal{C}_{j-1}\mathcal{F}_{j-2}, \ldots, \gamma_1 \mathcal{C}_j \cdots \mathcal{C}_1 \mathcal{F}_1\right].$$

Now by Lemma 3.2 one observes that $\mathcal{C}_i\mathcal{F}_j = \mathcal{C}_{i,j}\mathcal{F}_i$ and $\mathcal{C}_i\mathcal{C}_j = \mathcal{C}_{i,j}\mathcal{C}_{j,i}\ \forall i, j$, such that

$$Z_j = \left[\gamma_j \mathcal{F}_j,\ \gamma_{j-1}\mathcal{C}_{j-1,j}\mathcal{F}_j,\ \gamma_{j-2}\mathcal{C}_{j-2,j-1}\mathcal{C}_{j-1,j}\mathcal{F}_j, \ldots, \gamma_1 \mathcal{C}_{1,2} \cdots \mathcal{C}_{j-1,j}\mathcal{F}_j\right].$$

Thus, Z_j is for $j > 1$ augmented by r columns which are constructed from applying a Cayley transform to the previous r columns processed at the iteration step $j - 1$. Reversing the order of the shift parameters yields the low-rank ADI iteration

$$V_1 = (A + \alpha_1 I_n)^{-1} F,\quad Z_1 = \sqrt{-2\,\mathrm{Re}\ (\alpha_1)} V_1 \tag{3.10a}$$

$$V_j = \mathcal{C}_{j-1,j} V_{j-1} = V_{j-1} - (\alpha_j + \overline{\alpha_{j-1}})(A + \alpha_j I_n)^{-1} V_{j-1},\quad j > 1, \tag{3.10b}$$

$$Z_j = \left[Z_{j-1},\ \sqrt{-2\,\mathrm{Re}\ (\alpha_j)} V_j\right], \tag{3.10c}$$

where we used Proposition 2.16a to rewrite $\mathcal{C}_{j-1,j}$. In this form, only r columns need to be processed in each iteration step and the main computational effort comes from the solution of the associated shifted linear system with r right hand sides. From now on, we assume that we are able to solve the linear systems efficiently by either sparse direct [87, 74] or iterative solvers [199, 225], and that the obtained numerical solutions are approximate enough such that any errors can be neglected. The case of inexact

linear system solves is considered, e.g., in [202, 219]. As the low-rank solution factor Z is augmented by r columns in each iteration step, the approximate solution is given as sum of dyadic products of these columns:

$$X_j = Z_j Z_j^H = -2 \sum_{i=1}^{j} \operatorname{Re}(\alpha_i)\, V_i V_i^H. \tag{3.11}$$

Dealing with Generalized CALEs. Generalized, continuous time Lyapunov equations

$$AXE^T + EXA^T = -FF^T \tag{3.12}$$

with nonsingular E can be transformed to standard CALEs

$$\hat{A}X + X\hat{A}^T = -\hat{F}\hat{F}^T \quad \text{with} \quad \hat{A} = E^{-1}A, \quad \hat{F} = E^{-1}F$$

and the iteration (3.10) can be applied to $\hat{A}$, $\hat{F}$ in a straightforward way. From a numerical point of view, this is not wise since forming $\hat{A}$ might be too expensive in a large-scale setting. Moreover, it can possibly become ill-conditioned and dense. A more sound way to deal with E is using

$$(\hat{A} + \alpha I_n)^{-1} = (A + \alpha E)^{-1} E$$

in the linear systems in (3.10). This amounts to work with generalized Cayley transformations (Definition 2.15b) in the above derivation of the ADI iteration which yields, again by using Proposition 2.16a,

$$\begin{aligned} V_1 &= (A + \alpha_1 E)^{-1} F, \\ V_j &= V_{j-1} - (\alpha_j + \overline{\alpha_{j-1}})(A + \alpha_j E)^{-1}(EV_{j-1}), \quad j > 1, \end{aligned}$$

see, e.g., [18, 201]. Compared to (3.10), the identity I_n in the shifted linear systems is replaced by E and the right hand side for $j > 1$ is EV_{j-1}. The resulting low-rank ADI iteration for GCALEs (G-LR-ADI) is summarized in Algorithm 3.1. For standard CALEs (3.2), we use the abbreviation LR-ADI for Algorithm 3.1.

Remark 3.3:

Due to the Cholesky-type structure $X \approx ZZ^H$ of the computed approximate solution, Algorithm 3.1 is also frequently called (generalized) low-rank Cholesky-factor ADI iteration and abbreviated by (G-)LRCF-ADI. In this thesis we refrain from using the term Cholesky-factor since the computed low-rank factors Z are not triangular and because it shortens the used abbreviations for the algorithm to come. Also note that in several instances, the (G-)LR-ADI iteration is formulated using scaled iterates of the form $\sqrt{\operatorname{Re}(\alpha_j) / \operatorname{Re}(\alpha_{j-1})}V_j$. ◊

In Algorithm 3.1, the G-LR-ADI iteration is written using complex arithmetic operations because some of the shift parameters can be complex numbers. Hence, the computed low-rank factors Z are complex matrices in this case, although the intrinsic GCALE is defined by real matrices. Chapter 4 is entirely devoted to handle this undesirable property.

Algorithm 3.1: G-LR-ADI iteration for GCALEs [18, 201]

Input : Matrices $A,\ E,\ F$ defining (3.12) and shift parameters $\{\alpha_1, \ldots, \alpha_{j_{\max}}\} \subset \mathbb{C}_-$.

Output: $Z \in \mathbb{C}^{n \times r j_{\max}}$ such that $ZZ^H \approx X$.

$Z_0 = [\,]$.
for $j = 1, \ldots, j_{\max}$ **do**
  if $j = 1$ **then**
    Solve $(A + \alpha_1 E)V_1 = F$ for V_1.
  else
    Solve $(A + \alpha_j E)\hat{V} = EV_{j-1}$ for $\hat{V}$.
    $V_j = V_{j-1} - (\alpha_j + \overline{\alpha_{j-1}})\hat{V}$.
  $Z_j = \left[Z_{j-1},\ \sqrt{-2\,\mathrm{Re}\,(\alpha_j)}V_j\right]$.

3.2.2. Error Reduction and Shift Parameters

The shift parameters in the iteration schemes (3.3), (3.4) of the ADI iteration and, thus, also in the low-rank version (3.10) and Algorithm 3.1 are crucial for a fast convergence. Often, these shifts are constructed such that they reduce the error $X_j - X$. The next lemma quantifies this error, where we stick to the case $E = I_n$ for simplification and due to the equivalence discussed in the previous subsection.

Lemma 3.4 (Error of the ADI iteration [202, 94, 135]):

The error at iteration step j of (3.4) and of the equivalent reformulations (3.3), (3.10), is given by

$$X_j - X = \left[\prod_{i=1}^{j} \mathcal{C}_i\right](X_0 - X)\left[\prod_{i=1}^{j} \mathcal{C}_i^H\right]. \tag{3.13}$$

Proof. Combining Lemma 3.1 with (3.4) immediately gives the result. □

This lemma yields $\|X_j - X\| \leq \left\|\prod_{i=1}^{j} \mathcal{C}_i\right\|^2 \|X_0 - X\|$. Assuming that A is diagonalizable, it can be factorized as $A = Q\Lambda Q^{-1}$ with $\Lambda = \mathrm{diag}\,(\lambda_1, \ldots, \lambda_n)$, and Q contains the right eigenvectors of A. Then

$$\begin{aligned}\left\|\prod_{i=1}^{j} \mathcal{C}_i\right\| &= \left\|Q\left(\prod_{i=1}^{j} \mathcal{C}(\Lambda,\ \alpha_i)\right)Q^{-1}\right\| \leq \kappa(Q)\left\|\left(\prod_{i=1}^{j} \mathcal{C}(\Lambda,\ \alpha_i)\right)\right\| \\ &= \kappa(Q)\prod_{i=1}^{j} \rho\left(\mathcal{C}(\Lambda,\ \alpha_i)\right) = \kappa(Q)\left\|\mathrm{diag}\left(\prod_{i=1}^{j} \tfrac{\lambda_1 - \overline{\alpha_i}}{\lambda_1 + \alpha_i}, \ldots, \prod_{i=1}^{j} \tfrac{\lambda_n - \overline{\alpha_i}}{\lambda_n + \alpha_i}\right)\right\|,\end{aligned}$$

see, e.g., [160, 161, 119]. A few remarks concerning the non-diagonalizable case can be found in [119]. By Proposition 2.16c, if $\alpha_i \in \mathbb{C}_-,\ \forall i \geq 1$ it holds $\rho_i := \rho\left(\mathcal{C}(\Lambda,\ \alpha_i)\right) < 1$

and, hence, $r_j := \prod_{i=1}^{j} \rho_i < 1$. Moreover, $r_j = \rho_j r_{j-1} < r_{j-1}$ indicating that the sequence of the spectral radii r_j is monotonically decreasing and, in the limit, will approach the value zero. Hence, for every prescribed tolerance $\tau > 0$, there is a $j(\tau) \in \mathbb{N}$ such that $\|X_{j(\tau)} - X\| < \tau$. Now consider the resulting error bound

$$\|X_j - X\| \leq \kappa(Q)^2 r_j^2 \|X_0 - X\| =: \hat{r}_j.$$

If there exists $0 < \hat{\rho} < 1$ such that $\rho_j < \hat{\rho}$, $\forall j \geq 1$, the sequence $\hat{r}_j$ converges by [142, Definition 4.1.1.] q-linearly to zero and, consequently, the iterates X_j converge r-linearly towards X.

From the above bound we see that, for non-normal A, the speed at which the error is reduced is slowed down because $\kappa(Q) > 1$. Further investigations concerning the effects of non-normality can be found in [202, 7]. One often used approach to achieve a fast reduction of the error is to make the spectral radii r_j as small as possible which yields the min-max problem

$$\{\alpha_1^*, \ldots, \alpha_j^*\} = \underset{\alpha_1,\ldots,\alpha_j \subset \mathbb{C}_-}{\operatorname{argmin}} \left(\max_{1 \leq \ell \leq n} \left| \prod_{i=1}^{j} \frac{\lambda_\ell - \overline{\alpha_i}}{\lambda_\ell + \alpha_i} \right| \right), \ \lambda_\ell \in \Lambda(A), \tag{3.14}$$

where $\Lambda(A,\ E)$ is used for GCALEs. This rational optimization problem is also known as ADI shift parameter problem [232, 233]. We postpone further details regarding (3.14) until Chapter 5 which is entirely concerned with strategies for computing shift parameters. There, we also give information on approaches for solving (3.14) analytically or heuristically. For now we only mention that one heuristic approach [183] consists of replacing $\Lambda(A,\ E)$ by a small number of approximate, stable eigenvalues of $(A,\ E)$ and solve (3.14) in some sense approximately, resulting in a small number $J \ll n$ of shift parameters. This is especially helpful for the large-scale case in this thesis since the entire spectrum is unlikely to be available or efficiently computable in this situation. Typically, these J shifts are used in a cyclical manner if the number of iteration steps exceeds J. In this case, bounding the spectral radii is trivial because $\hat{\rho} := \max_{1 \leq \ell \leq J}(\rho_\ell) < 1$. The approximate eigenvalues of $(A,\ E)$ in the heuristic approach, but also in several other shift generation strategies [202, 39], are typically taken as Ritz values, i.e., the eigenvalues of $(Q^TAQ,\ Q^TEQ)$ for $Q \in \mathbb{R}^{n \times k}$, $Q^TQ = I_k$. For obtaining $\hat{\rho} < 1$ it has to hold $\Lambda(Q^TAQ,\ Q^TEQ) \subset \mathbb{C}_-$. This is, for instance, the case when $E = I_n$ and it holds for the field of values [132] of A: $\mathcal{W}(A) := \{z^HAz,\ 0 \neq z \in \mathbb{C}^n,\ \|z\| = 1\} \subset \mathbb{C}_-$. This condition holds not generically, e.g., when $A + A^T$ is not negative definite [132]. In that situation, some additional precautions are required to obtain $\alpha_i \in \mathbb{C}_-$ and, therefore, $\hat{\rho} < 1$. These techniques, as well as alternatives to the heuristic shift selection approach mentioned above, will also be subject of Chapter 5.

3.2.3. Stopping Criteria

One way to stop Algorithm 3.1 is when the change in the approximate solution is small, i.e., when $\|X_j - X_{j-1}\| < \tau$, which can be computed efficiently in the spectral and

Frobenius norms via

$$\|X_j - X_{j-1}\| = \|Z_j Z_j^H - Z_{j-1} Z_{j-1}^H\| = -2\,\mathrm{Re}\,(\alpha_j)\,\|V_j V_j^H\| = -2\,\mathrm{Re}\,(\alpha_j)\,\|V_j\|^2,$$

see [161]. A relative change can also be used, e.g., by scaling with $\|X_j\|$. However, the computation of the norm of the approximate solution might be expensive in large dimensions. Taking the computed symmetric low-rank structure $Z_j Z_j^H$ into account, using a power iteration or Lanczos process [111] can be employed to reveal the spectral norm since $\|M\|_2 = |\lambda_{\max}(M)| = \rho(M)$ for any symmetric matrix M. A similar stopping criterion is based on the relative change in the low-rank factors [42, 184] measured in the Frobenius norm, i.e., to stop the ADI iteration if

$$\|V_j\|_F / \|Z_j\|_F \leq \tau.$$

It is not necessary to compute $\|Z_j\|_F$ every time, because $\|Z_j\|_F^2 = \|Z_{j-1}\|_F^2 + \|V_j\|_F^2$, only the Frobenius norm of V_j has to be computed in each iteration step. A disadvantage of both approaches is that the used (relative) norms do not necessarily decrease as the LR-ADI iteration progresses and one often observes a quiet irregular behavior. Hence, we do not employ these approaches in the remainder.

Instead we focus on terminating Algorithm 3.1 based on the norm of the (generalized) Lyapunov residual matrix

$$\mathcal{L}_j := AX_jE^T + EX_jA^T + FF^T. \tag{3.15}$$

A popular stopping condition is

$$\|\mathcal{L}_j\| / \phi_j < \tau,$$

where ϕ_j is a suitable scaling constant. Typical choices are $\phi_j \equiv \phi = \|FF^T\|$ and a backward error related scaling $\phi_j = \|FF^T\| + 2\|A\|\|E\|\|X_j\|$ which is more expensive to evaluate. We will usually employ the first choice. For large-scale Lyapunov equations, using the Lyapunov residual for terminating the ADI iteration is difficult since $\mathcal{L}_j$ is a large and dense matrix, such that even constructing and storing it is infeasible and computing, e.g., the spectral or Frobenius norm is expensive. We mention briefly two approaches for this purpose.

Computing the Lyapunov Residual Norm One approach to compute $\|\mathcal{L}_j\|_F$ more efficiently was proposed in [182, 183] and uses the decomposition

$$\mathcal{L}_j = H_j D_j H_j^H, \quad H_j := [AZ_j,\ EZ_j,\ F], \quad D_j := \begin{bmatrix} 0 & I_{rj} & 0 \\ I_{rj} & 0 & 0 \\ 0 & 0 & I_r \end{bmatrix}. \tag{3.16a}$$

Using the thin QR decomposition $Q_j R_j = H_j$, $Q_j \in \mathbb{C}^{n\times(2j+1)r}$, $R_j \in \mathbb{C}^{(2j+1)r\times(2j+1)r}$, the spectral and Frobenius norms of the residual can by computed by

$$\|\mathcal{L}_j\| = \|R_j D_j R_j^H\|. \tag{3.16b}$$

To reduce the computation costs, the QR decomposition can be updated incrementally in each step since H_j is updated by $2r$ columns. As j increases, this approach can nevertheless become easily more expensive than the remaining computation in each step of the G-LR-ADI iteration.

Alternatively, since $\mathcal{L}_j$ is also a symmetric matrix and $\|\mathcal{L}_j\|_2$ coincides with the spectral radius of $\mathcal{L}_j$, one could use, by exploiting the low-rank structure $X_j = Z_j Z_j^T$ provided by the G-LR-ADI iteration, a power iteration or a Lanczos process (see, e.g., [111]) to retrieve this largest eigenvalue. This would essentially require only matrix vector products with A, E, G, Z and their transposes. Unless the power iteration or Lanczos process converge in very few steps, this can still lead to a high portion of computational effort in the G-LR-ADI iteration. Especially the power iteration tends to converge increasingly slow when the residual gets smaller. Since the column dimension of the low-rank solutions factor increases as the G-LR-ADI iteration proceeds, the computational costs for applying a power iteration or a Lanczos process will also increase.

In the next subsection we investigate the Lyapunov residual (3.15) in more depth. A novel result on the low-rank structure of (3.15) will be derived which not only enables a cheap evaluation of $\|\mathcal{L}_j\|$, but also reveals a new way to reformulate Algorithm 3.1.

3.2.4. The Low-Rank Structure of the Residual Matrix and a Reformulated Iteration

The next theorem gives a novel result on the structure of $\mathcal{L}_j$ within the G-LR-ADI iteration and constitutes the main contribution of this section.

Theorem 3.5 (Low-rank structure of $\mathcal{L}_j$ [37]):

The residual at iteration step j of the G-LR-ADI iteration is of rank at most r and given by

$$\mathcal{L}_j = AZ_jZ_j^HE^T + EZ_jZ_j^HA^T + FF^T = W_jW_j^H, \tag{3.17}$$

where $W_j \in \mathbb{C}^{n\times r}$ is given by

$$\begin{aligned} W_j &:= (A - \overline{\alpha_j}E)V_j && \text{(3.18a)}\\ &= W_{j-1} - 2\,\mathrm{Re}\,(\alpha_j)\,EV_j && \text{(3.18b)}\\ &= W_0 + EZ_j\Gamma_j(\mathbf{1}_j \otimes I_r) && \text{(3.18c)} \end{aligned}$$

with $W_0 := F$, $\Gamma_j := \mathrm{diag}\,(\gamma_1, \ldots, \gamma_j) \otimes I_r$, $\gamma_i := \sqrt{-2\,\mathrm{Re}\,(\alpha_i)}$ for $i = 1, \ldots, j$, and $\mathbf{1}_h = [1, \ldots, 1]^T \in \mathbb{R}^h$. If $\alpha_j \notin \Lambda(A,\ E)$ for all j, then the rank is exactly r. Moreover, the iterates V_j in Algorithm 3.1 can be equivalently constructed by

$$V_j = (A + \alpha_j E)^{-1}W_{j-1}, \quad j \geq 1. \tag{3.19}$$

Proof. To ease the representation, we consider the application of the LR-ADI iteration to the standard CALE

$$\hat{A}X + X\hat{A}^T + \hat{F}\hat{F}^T = 0, \quad \hat{A} := E^{-1}A, \quad \hat{G} := E^{-1}F, \tag{3.20}$$

which is equivalent to applying the G-LR-ADI iteration to the GCALE (3.12). Using Lemma 3.4:

$$X_j - X = \left[\prod_{i=1}^{j} \mathcal{C}(\hat{A}, \alpha_i)\right] (X_0 - X) \left[\prod_{i=1}^{j} \mathcal{C}(\hat{A}, \alpha_j)^H\right]$$

reveals, by employing the Lemmas 3.1 and 3.2, as in [94, Lemma 5.3], [135, Lemma 3.5.2], that the residual at iteration step j of Algorithm 3.1 can be written as

$$\begin{aligned}\hat{\mathcal{L}}_j &= \hat{A}X_j + X_j\hat{A}^T + \hat{F}\hat{F}^T = \hat{A}(X_j - X) + (X_j - X)\hat{A}^T \\ &= \left[\prod_{i=1}^{j} \mathcal{C}(\hat{A}, \alpha_i)\right] \hat{\mathcal{L}}_0 \left[\prod_{i=1}^{j} \mathcal{C}(\hat{A}, \alpha_j)^H\right].\end{aligned}$$

Now since $X_0 = 0$ in our setting, we have $\hat{\mathcal{L}}_0 = \hat{F}\hat{F}^T$ such that the above relation immediately yields

$$\hat{\mathcal{L}}_j = \hat{W}_j\hat{W}_j^H \quad \text{with} \quad \hat{W}_j := \prod_{i=1}^{j} \mathcal{C}(\hat{A}, \alpha_i)\hat{F}. \tag{3.21}$$

This already shows that $\operatorname{rank}(\hat{\mathcal{L}}_j) = r$ if $\alpha_j \notin \Lambda(\hat{A})$, $\forall j$. If any shift α_j is an eigenvalue of $\hat{A}$, then the inverse of $\hat{A}+\alpha_j I_n$ still exists, but the matrix $\hat{A}-\overline{\alpha_j}I_n$ has a rank deficiency and, thus, the rank of $\mathcal{L}_j$ may drop below r. The increment V_j can be expressed as

$$\begin{aligned}V_j &= (\hat{A} - \overline{\alpha_{j-1}}I_n)(\hat{A} + \alpha_j I_n)^{-1}V_{j-1} \\ &= (\hat{A} - \overline{\alpha_{j-1}}I_n)(\hat{A} + \alpha_j I_n)^{-1}(\hat{A} - \overline{\alpha_{j-2}}I_n)(\hat{A} + \alpha_{j-1}I_n)^{-1}V_{j-2} \\ &= (\hat{A} + \alpha_j I_n)^{-1}(\hat{A} - \overline{\alpha_{j-1}}I_n)(\hat{A} + \alpha_{j-1}I_n)^{-1}(\hat{A} - \overline{\alpha_{j-2}}I_n)V_{j-2} \\ &= \ldots = (\hat{A} + \alpha_j I_n)^{-1}\prod_{i=1}^{j-1} \mathcal{C}(\hat{A}, \alpha_i)\hat{F},\end{aligned} \tag{3.22}$$

where we used Lemma 3.2 again. Comparing (3.21) and (3.22) yields $\hat{W}_j = (\hat{A}-\overline{\alpha_j}I_n)V_j$ which is (3.18a). The relations (3.18b), (3.19) are then established by

$$\begin{aligned}V_j &= (\hat{A} + \alpha_j I_n)^{-1}(\hat{A} - \overline{\alpha_{j-1}}I_n)V_{j-1} = (\hat{A} + \alpha_j I_n)^{-1}\hat{W}_{j-1}, \\ \hat{W}_j &= (\hat{A} - \overline{\alpha_j}I_n)V_j = (\hat{A} - \overline{\alpha_j}I_n)(\hat{A} + \alpha_j I_n)^{-1}\hat{W}_{j-1} \\ &= \left(I_n - (\alpha_j + \overline{\alpha_j})(\hat{A} + \alpha_j I_n)^{-1}\right)\hat{W}_{j-1} = \hat{W}_{j-1} - 2\operatorname{Re}(\alpha_j)\, V_j\end{aligned}$$

which holds also for $j = 1$ if we define $\hat{W}_0 := \hat{F}$.

The results (3.18a), (3.18b), (3.19) for GCALEs are obtained via

$$\begin{aligned}\mathcal{L}_j &= E\hat{\mathcal{L}}_j E^T = E\hat{W}_j\hat{W}_j^H E^T = W_j W_j^H, \\ W_j &:= E\hat{W}_j = (A - \overline{\alpha_j}E)V_j = W_{j-1} - 2\operatorname{Re}(\alpha_j)\, EV_j, \\ V_j &= (\hat{A} + \alpha_j I_n)^{-1}\hat{W}_{j-1} = (A + \alpha_j E)^{-1}E\hat{W}_j = (A + \alpha_j E)^{-1}W_j,\end{aligned}$$

Algorithm 3.2: Reformulated G-LR-ADI iteration

Input : Matrices $A,\ E,\ F$ defining (3.12), shift parameters $\{\alpha_1, \ldots, \alpha_{j_{\max}}\} \subset \mathbb{C}_-$, and tolerance $0 < \tau \ll 1$.
Output: $Z \in \mathbb{C}^{n \times rj}$ such that $ZZ^H \approx X$.

$Z_0 = [],\ W_0 = F,\ j = 1.$
while $\|W_{j-1}^H W_{j-1}\| \geq \tau \|F^T F\|$ **do**
  Solve $(A + \alpha_j E)V_j = W_{j-1}$ for V_j.
  $W_j = W_{j-1} - 2\,\mathrm{Re}\,(\alpha_j)\,EV_j$.
  $Z_j = [Z_{j-1},\ \sqrt{-2\,\mathrm{Re}\,(\alpha_j)}V_j]$.
  $j = j + 1$.

and $W_0 := F$. The expression (3.18c) simply follows from subsequently using (3.18b) and the definition of the low-rank solution factor Z_j in Algorithm 3.1. □

Remark 3.6:

Note that the relations in (3.18b), (3.18c) were (in a slightly altered notation) independently derived in a different way in [235, Theorem 5.1., Corollary 5.1.], where the authors used the relation of the LR-ADI iteration with rational Krylov subspaces [160, 161, 96, 21]. ◊

This novel result has the following impacts. At first it enables the computation of the spectral or Frobenius norm of the Lyapunov residual via $\|\mathcal{L}_j\| = \|W_j W_j^H\| = \|W_j^H W_j\|$ which is much cheaper than the other approaches mentioned above. For the spectral norm it also holds $\|\mathcal{L}_j\|_2 = \|W_j\|_2^2$. The only requirement is the computation of the spectral norm of a small $r \times r$ or a thin rectangular $n \times r$ matrix. This is significantly cheaper than the approaches mentioned in Subsection 3.2.3 as we will also see in the numerical example in 3.2.6. Moreover, the result is exact (in finite arithmetics) compared to the approximate results obtained with power iteration and Lanczos process. Another effect of the Theorem 3.5 is that combining (3.18) and (3.19) reveals a novel but mathematically equivalent formulation of the G-LR-ADI iteration, where the low-rank factors W_j of $\mathcal{L}_j$ are an integral part of the iteration, see also [234, Corollary 5.15]. Due to the initialization $W_0 = F$, no distinction between the first and all other iteration steps $j \geq 2$, as in Algorithm 3.1, is required. The complete reformulated G-LR-ADI iteration is given in Algorithm 3.2, where we used (3.18b) and already included a termination criterion based on the scaled residual norm.

Remark 3.7:

We point out that the relations developed in Theorem 3.5 only hold if the linear systems (3.19) are solved exactly as we assumed from the beginning on. If they are solved inexactly, e.g., by applying an iterative solver until the norm of the linear system's residual falls below a certain tolerance, Algorithm 3.2 can still be applied. However, in that case the W_j are not anymore the low-rank factors of $\mathcal{L}_j$ such that $\|W_j^H W_j\|$ is not equal to $\|\mathcal{L}_j\|$. If the tolerance regarding the linear system is chosen

too large, $\|W_j^H W_j\|$ appears to underestimate $\|\mathcal{L}_j\|$ and the Lyapunov residual norm should be estimated in a different way, e.g., via the Lanczos approach mentioned above. Choosing a small enough tolerance gives a negligible discrepancy between $\|W_j^H W_j\|$ and $\|\mathcal{L}_j\|$. Deeper investigations regarding the ADI iteration with inexact linear solves can be found in [202, 219]. ◊

This new formulation also reveals another insightful point of view of the G-LR-ADI iteration. In each step Algorithm 3.2 implicitly solves a GCALE defined by A, E and the previous residual matrix $W_{j-1}W_{j-1}^H$.

Corollary 3.8:

Consider the GCALE (3.12) and the shift parameters $\{\alpha_1, \ldots, \alpha_{\ell+k}\} \subset \mathbb{C}_-$. Let Z_k be the low-rank solution factor after k iterations of the G-LR-ADI iteration (Algorithm 3.1 or 3.2). Moreover, let $\tilde{Z}_\ell^{(k)}$ denote the low-rank factor after ℓ iteration steps applied to the GCALE

$$AXE^T + EXA^T + W_k W_k^H = 0 \tag{3.23}$$

and the shift parameters $\tilde{\alpha}_i := \alpha_{k+i}$ for $i = 1, \ldots, \ell$. Then for $k + \ell$ iteration steps of Algorithm 3.2 it holds $Z_{k+\ell} = [Z_k, \quad \tilde{Z}_\ell^{(k)}]$. ◊

Proof. The cases $k = 0$ and $\ell = 0$ are trivial to proof. For $k \geq 1$ we have

$$\begin{aligned} V_k &= (A + \alpha_k E)^{-1} W_k, \\ W_k &= F + E Z_k \Gamma_k (\mathbf{1}_k \otimes I_r), \\ Z_k &= [\gamma_1 V_1, \ldots, \gamma_k V_k], \quad \Gamma_k = \operatorname{diag}(\gamma_1, \ldots, \gamma_k) \otimes I_r \end{aligned}$$

with $\gamma_i := \sqrt{-2\operatorname{Re}(\alpha_k)}$, $i = 1, \ldots, k$. Let $\tilde{V}_\ell^{(k)}$, $\tilde{W}_\ell^{(k)}$ denote the iterates and residual factors after $\ell \geq 1$ steps of the G-LR-ADI iteration applied to the GCALE (3.23) and the shifts $\tilde{\alpha}_i$, $i = 1, \ldots, \ell$. By Theorem 3.5 it holds

$$\begin{aligned} \tilde{V}_\ell^{(k)} &= (A + \tilde{\alpha}_\ell E)^{-1} \tilde{W}_{\ell-1}^{(k)} = (A + \alpha_{k+\ell} E)^{-1} (A - \overline{\alpha_{k+\ell-1}} E) \tilde{V}_{\ell-1}^{(k)} \\ &= \ldots = (A + \alpha_{k+\ell} E)^{-1} (A - \overline{\alpha_{k+\ell-1}} E) \cdots (A + \alpha_{k+1} E)^{-1} \tilde{W}_0^{(k)} \\ &= (A + \alpha_{k+\ell} E)^{-1} (A - \overline{\alpha_{k+\ell-1}} E) \cdots (A + \alpha_{k+1} E)^{-1} W_k = V_{k+\ell}. \end{aligned}$$

Hence,

$$\tilde{Z}_\ell^{(k)} = [\tilde{\gamma}_1 \tilde{V}_1^{(k)}, \ldots, \tilde{\gamma}_k \tilde{V}_\ell^{(k)}] = [\gamma_{k+1} V_{k+1}, \ldots, \gamma_{k+\ell} V_{k+\ell}]$$

and the result is established. Also note that

$$\begin{aligned} \tilde{W}_\ell^{(k)} &= \tilde{W}_0^{(k)} + E \tilde{Z}_\ell^{(k)} \tilde{\Gamma}_\ell^{(k)} (\mathbf{1}_\ell \otimes I_r) = W_k + E \tilde{Z}_\ell^{(k)} \tilde{\Gamma}_\ell^{(k)} (\mathbf{1}_\ell \otimes I_r) \\ &= F + E \left[Z_k, \tilde{Z}_\ell^{(k)} \right] \operatorname{diag}\left(\Gamma_k, \ \tilde{\Gamma}_\ell^{(k)} \right) (\mathbf{1}_{\ell+k} \otimes I_r) \\ &= F + E \left[\gamma_1 V_1, \ldots, \gamma_k V_k, \gamma_{k+1} V_{k+1}, \ldots, \gamma_{k+\ell} V_{k+\ell} \right] \Gamma_{k+\ell} (\mathbf{1}_{\ell+k} \otimes I_r) \\ &= F + E Z_{k+\ell} \Gamma_{k+\ell} (\mathbf{1}_{\ell+k} \otimes I_r) = W_{k+\ell}. \end{aligned}$$

□

With the help of the relations (3.18), the low-rank solution factors generated by the G-LR-ADI iteration can also be expressed as the solutions of certain Sylvester equations. Similar results regarding the original G-LR-ADI iteration (3.10) can be found in [160, 161].

Corollary 3.9 (Adaptation of [235, Lemma 3.1], [234, Lemma 5.12]):

For the GCALE (3.12), shift parameters $\{\alpha_1, \ldots, \alpha_j\} \subset \mathbb{C}_-$, and $\gamma_i = \sqrt{-2\operatorname{Re}(\alpha_i)}$, $i = 1, \ldots, j$, the low-rank solution factor Z_j after j steps of the G-LR-ADI iteration (Algorithm 3.2) satisfies the Sylvester equations

$$AZ_j + EZ_jB_{\text{ADI}} = FG_{\text{ADI}}^T, \tag{3.24a}$$

$$AZ_j - EZ_jB_{\text{ADI}}^T = W_jG_{\text{ADI}}^T \tag{3.24b}$$

with

$$B_{\text{ADI}} := \begin{bmatrix} \alpha_1 & -\gamma_1\gamma_2 & \cdots & -\gamma_1\gamma_j \\ & \ddots & \ddots & \vdots \\ & & \ddots & -\gamma_{j-1}\gamma_j \\ & & & \alpha_j \end{bmatrix} \otimes I_r \in \mathbb{C}^{jr\times jr}, \quad G_{\text{ADI}} := \Gamma_j(\mathbf{1}_j \otimes I_r). \tag{3.24c}$$

Note that only a real transposition is applied to the possibly complex matrix B_{ADI} in (3.24b).

Proof. The GCASE (3.24a) is exactly the one in [235, Lemma 3.1], [234, Lemma 5.12] adapted to our notation. The basic idea of the proof is to consider and merge (3.19), (3.18b) for $i = 1, \ldots, j$:

$$\begin{aligned} AV_i &= W_{i-1} - \alpha_iEV_i = W_0 - \alpha_iEV_i - 2E\sum_{k=1}^{i-1} V_k \operatorname{Re}(\alpha_k) \\ &= W_0 - E[V_1, \ldots, V_i][2\operatorname{Re}(\alpha_1)\, I_r, \ldots, 2\operatorname{Re}(\alpha_{i-1})\, I_r, \alpha_iI_r]^T \end{aligned}$$

such that

$$A[V_1, \ldots, V_j] = W_0(\mathbf{1}_r^T \otimes I_r) - E[V_1, \ldots, V_j]\left(\begin{bmatrix} \alpha_1 & 2\operatorname{Re}(\alpha_1) & \cdots & 2\operatorname{Re}(\alpha_1) \\ & \ddots & \ddots & \vdots \\ & & \ddots & 2\operatorname{Re}(\alpha_{j-1}) \\ & & & \alpha_j \end{bmatrix} \otimes I_r\right).$$

Incorporating $Z_j = [V_1, \ldots, V_j]\Gamma_j$ into this formula yields (3.24a) after some basic manipulations.

For (3.24b) it is by iteratively using (3.18b) easy to see that

$$W_{j-i} = W_j + 2E\sum_{k=0}^{i-1} V_{j-k}\operatorname{Re}(\alpha_{j-k}), \quad i = 1, \ldots, j-1.$$

Hence, it holds for $i = 1, \ldots, j-1$

$$AV_{j-i} = W_j + E[V_{j-i}, \ldots, V_j][-\alpha_{j-i}I_r, 2\operatorname{Re}(\alpha_{j-i})\, I_r, \ldots, 2\operatorname{Re}(\alpha_j)\, I_r]^T$$

such that with $AV_j = W_j - \alpha_j E V_j$,

$$A[V_1,\dots,V_j] = W_j(\mathbf{1}_r^T \otimes I_r) + E[V_1,\dots,V_j]\left(\begin{bmatrix} -\alpha_1 & & & \\ 2\operatorname{Re}(\alpha_2) & -\alpha_2 & & \\ \vdots & \ddots & \ddots & \\ 2\operatorname{Re}(\alpha_j) & \cdots & 2\operatorname{Re}(\alpha_j) & -\alpha_j \end{bmatrix} \otimes I_r\right).$$

The final result (3.24b) is again obtained by inserting $Z_j = [V_1,\dots,V_j]\Gamma_j$. □

Representing the low-rank solution factor Z_j via (3.24) will be useful in the upcoming Chapters 5 and 6. It is known that the G-LR-ADI iteration can be connected to rational Krylov subspace methods [160, 161, 82, 96, 235, 237, 234]. In this context the relations (3.24) do not appear surprising as they represent generalizations of some results from [193, 194]. The authors in [235, 237, 234] use (3.24a) to investigate the connection between the G-LR-ADI iteration and $\mathcal{H}_2$ model order reduction, see also [96, 21]

3.2.5. Structure Exploiting Versions of the G-LR-ADI Iteration for Special GCALEs

The Second Order LR-ADI Iteration

Second order systems (2.32) are often formally rewritten as equivalent generalized state space systems (2.5) of first order with matrices $E,\ A \in \mathbb{R}^{2n\times 2n}$, $F \in \mathbb{R}^{2n\times r}$, $L \in \mathbb{R}^{p\times 2n}$ defined by, e.g., (2.33) (cf. Section 2.4).

In order to compute and approximate solutions to the GCALEs defined by $E,\ A,\ F,\ L$ it is not recommended to explicitly form these augmented matrices. It is possible to exploit the structure given by (2.33) efficiently by suitably rewriting the occurring matrix vector products and shifted linear systems. For the G-LR-ADI iteration in its original form (3.12) this has been done in [201, 51, 178, 37] which leads to the so called second order LR-ADI iteration (SO-LR-ADI). Here we briefly sketch the adaption of this approach to the reformulated G-LR-ADI iteration in Algorithm 3.2, where we restrict ourselves to the second companion linearization (2.33b) with $N_2 = M$, i.e.,

$$E_2 = \begin{bmatrix} D & M \\ M & 0 \end{bmatrix},\ A_2 = \begin{bmatrix} -K & 0 \\ 0 & M \end{bmatrix},\ F_2 = \begin{bmatrix} B_1 \\ 0 \end{bmatrix},\ L_2 = \begin{bmatrix} C^{\mathrm{p}} & C^{\mathrm{v}} \end{bmatrix}.$$

The key ingredient is to partition the $2n \times r$ arrays V_j and W_j into blocks of dimension $n \times r$

$$V_j = \begin{bmatrix} V_j^{(\mathrm{p})} \\ V_j^{(\mathrm{v})} \end{bmatrix}, \quad W_j = \begin{bmatrix} W_j^{(\mathrm{p})} \\ W_j^{(\mathrm{v})} \end{bmatrix}.$$

Then, the linear systems $(A_2 + \alpha_j E_2)V_j = W_{j-1}$ are equivalent to

$$(\alpha_j^2 M - \alpha_j D + K)V_j^{(\mathrm{p})} = -W_{j-1}^{(\mathrm{p})} + \alpha_j W_{j-1}^{(\mathrm{v})}, \tag{3.25a}$$

$$V_j^{(\mathrm{v})} = -\alpha_j V_j^{(\mathrm{p})} + M^{-1} W_{j-1}^{(\mathrm{v})}, \tag{3.25b}$$

Algorithm 3.3: Reformulated SO-LR-ADI iteration

Input : Matrices $M,\ D,\ K$ and F defining (2.33b), shift parameters $\{\alpha_1, \ldots, \alpha_{j_{\max}}\} \subset \mathbb{C}_-$, and tolerance $0 < \tau \ll 1$.

Output: $Z \in \mathbb{C}^{2n \times rj}$ such that $ZZ^H \approx X$.

1 $Z_0 = [\,],\ \begin{bmatrix} W_0^{(\mathrm{p})} \\ W_0^{(\mathrm{v})} \end{bmatrix} = F,\ j = 1.$

2 **while** $\|W_{j-1}^H W_{j-1}\| \geq \tau \|F^T F\|$ **do**

3 Solve $(\alpha_j^2 M - \alpha_j D + K)V_j^{(\mathrm{p})} = -W_{j-1}^{(\mathrm{p})} + \alpha_j W_{j-1}^{(\mathrm{v})}$ for $V_j^{(\mathrm{p})}$.

4 **if** $j = 1$ **then**

5 $V_j^{(\mathrm{v})} = -\alpha_j V_j^{(\mathrm{p})}$.

6 **if** $W_0^{(v)} \neq 0$ **then** $V_j^{(\mathrm{v})} = V_j^{(\mathrm{v})} + M^{-1} W_0^{(\mathrm{v})}$.

7

8 **else**

9 $V_j^{(\mathrm{v})} = -\alpha_j V_j^{(\mathrm{p})} + V_{j-1}^{(\mathrm{v})} - \overline{\alpha_{j-1}} V_{j-1}^{(\mathrm{p})}$.

10 $W_j^{(\mathrm{p})} = W_{j-1}^{(\mathrm{p})} - 2\,\mathrm{Re}\,(\alpha_j)\,(DV_j^{(\mathrm{p})} + MV_j^{(\mathrm{v})})$.

11 $W_j^{(\mathrm{v})} = W_{j-1}^{(\mathrm{v})} - 2\,\mathrm{Re}\,(\alpha_j)\,MV_j^{(\mathrm{v})}$.

12 $Z_j = \left[Z_{j-1},\ \sqrt{-2\,\mathrm{Re}\,(\alpha_j)} \begin{bmatrix} V_j^{(\mathrm{p})} \\ V_j^{(\mathrm{v})} \end{bmatrix} \right]$.

13 $j = j + 1$.

and, likewise, the blocks for the residual factor are given by

$$W_j^{(\mathrm{p})} = W_{j-1}^{(\mathrm{p})} - 2\,\mathrm{Re}\,(\alpha_j)\,(DV_j^{(\mathrm{p})} + MV_j^{(\mathrm{v})}), \tag{3.25c}$$

$$W_j^{(\mathrm{v})} = W_{j-1}^{(\mathrm{v})} - 2\,\mathrm{Re}\,(\alpha_j)\,MV_j^{(\mathrm{v})}. \tag{3.25d}$$

Multiplying (3.25d) by M^{-1} yields with (3.25b)

$$M^{-1}W_j^{(\mathrm{v})} = M^{-1}W_{j-1}^{(\mathrm{v})} - 2\,\mathrm{Re}\,(\alpha_j)\,V_j^{(\mathrm{v})} = V_j^{(\mathrm{v})} + (\alpha_j - 2\,\mathrm{Re}\,(\alpha_j))V_j^{(\mathrm{p})}$$

and, thus, it holds for (3.25b) and $j > 1$

$$V_j^{(\mathrm{v})} = -\alpha_j V_j^{(\mathrm{p})} + V_{j-1}^{(\mathrm{v})} + (\alpha_{j-1} - 2\,\mathrm{Re}\,(\alpha_{j-1}))V_{j-1}^{(\mathrm{p})}.$$

This way the required additional linear solve with M in (3.25b) can be circumvented for $j > 1$. For iteration step $j = 1$ it is only required once if $W_0^{(\mathrm{v})} \neq 0$. Otherwise, (3.25b) simplifies to $V_1^{(\mathrm{v})} = -\alpha_1 V_1^{(\mathrm{p})}$. For this particular choice of the linearization, the resulting reformulated SO-LR-ADI iteration is illustrated in Algorithm 3.3.

It is worth mentioning that Algorithm 3.3 can be easily modified to handle the first companion linearization (2.33a) and we outline only the essential changes: In the Lines 3 – 9 one simply has to interchange $V_j^{(\mathrm{p})}$ with $V_j^{(\mathrm{v})}$. If $N_2 = K$ in (2.33a), M in Line 6 has to be replaced by K. The blocks of the residual factors in the Lines 10,11 are simply deduced from the structure of E_1, see [37].

The SLRCF-ADI Iteration for Index One DAEs

Another special case we will frequently use in the upcoming numerical examples is given by the dynamical system (2.5) with the structure (2.34). One can associate a generalized Lyapunov equation defined by $\hat{A} := A_{11} - A_{12}A_{22}^{-1}A_{21}$ and $\hat{F} := F_1 - A_{12}A_{22}^{-1}F_2$ to (2.34) (cf. [98]).

Solving this generalized Lyapunov equation with G-LR-ADI is of course possible, but the matrices $\hat{A} + \alpha_j E_{11}$ in the occurring linear systems will in general be dense which prohibits the use of sparse solvers. In [98], the Sparse LRCF-ADI (SLRCF-ADI) iteration, a specially tailored G-LR-ADI iteration, is proposed which solves (2.35) without forming the matrices $\hat{A}$, $\hat{F}$ explicitly. The prefix SPARSE in SLRCF-ADI refers to this sparsity preserving implementation. The key ingredient exploited in [98] is the observation that the solution of the dense linear system $(\hat{A} + \alpha_j E_{11})V_j = W_{j-1}$ of size n_f can be equivalently and more efficiently be obtained from the sparse linear system

$$\begin{bmatrix} A_{11} + \alpha_j E_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \begin{bmatrix} V_j \\ \Psi \end{bmatrix} = \begin{bmatrix} W_{j-1} \\ 0 \end{bmatrix}, \tag{3.26}$$

of size n, where $\Psi \in \mathbb{C}^{n-n_f \times r}$ is an auxiliary variable. It can be shown that the right hand side in the first iteration can be set to $[F_1^T, F_2^T]^T$. The residual factors are given by $W_j = W_{j-1} - 2\,\mathrm{Re}\,(\alpha_j)\,E_{11}V_j$, where for W_0, the matrix $\hat{F}$ has to be computed once at the beginning of the iteration which requires one solve with the $(n - n_f) \times (n - n_f)$ matrix A_{22}, or, alternatively, the solution of the larger system

$$\begin{bmatrix} I_{n_f} & A_{12} \\ 0 & A_{22} \end{bmatrix} \begin{bmatrix} \hat{F} \\ \Psi \end{bmatrix} = \begin{bmatrix} F_1 \\ F_2 \end{bmatrix}.$$

There are also LR-ADI approaches for handling DAE systems of higher indices [217, 171]. For instance, in [10] the authors work with an approach regarding the index-2 case arising in optimal control of the (Navier)-Stokes equation. In [31], a combination of the SO-LR-ADI and SLRCF-ADI iterations for second-order systems of index one is investigated.

3.2.6. Numerical Examples

Here, we briefly use the *ocean* and *rail79k* examples to investigate the novel approach for computing the Lyapunov residual norm in the G-LR-ADI iteration. For both examples, we use a fixed number J of heuristic shift parameters [184] which are obtained from k_+ and k_- Ritz and inverse Ritz values, respectively, which are computed by two Arnoldi processes (w.r.t. A and A^{-1}) using $F\mathbf{1}_r$ as starting vector. We employ the scaled residual norm $\varepsilon_j := \|\mathcal{L}_j\|_2 / \|F^T F\|_2$ as stopping criterion with a tolerance τ. For comparison we employ the three previously mentioned different approaches to compute the spectral norm of $\mathcal{L}_j$. At first, an incrementally updated QR factorizations as in (3.16) and, secondly, a Lanczos process without reorthogonalization applied to $\mathcal{L}_j$ to approximate its largest eigenvalue. The Lanczos process is started with the initial vector $\mathbf{1}_n$ and is terminated after 5 steps or when the relative change of the approximate eigenvalue

Table 3.1.: Results obtained with different strategies for computing the Lyapunov residual norm within the G-LR-ADI iteration.

	Parameters			Results				
Example	k_+, k_-, J	τ	j_{iter}	$\varepsilon_{j_{\text{iter}}}$	t_{ADI}	t_W	t_{QR}	t_{Lan}
ocean	40, 40, 30	10^{-10}	69	$7.5{\cdot}10^{-11}$	117.4	6.2	467.4	85.2
rail79k	40, 40, 20	10^{-10}	54	$7.0{\cdot}10^{-11}$	53.4	0.7	35.6	7.7

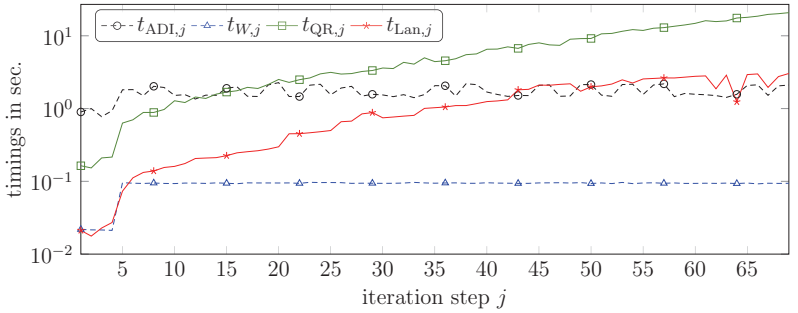


Figure 3.1.: Computation times of the G-LR-ADI iteration itself and of the different variants of computing $\|\mathcal{L}_j\|$ against the iteration step j for the *ocean* example.

is smaller than 10^{-2}. The last approach uses the novel low-rank factorization of $\mathcal{L}_j$ established in Theorem 3.5 with the cheap expression (3.18a) for the residual factor W_j, which allows to determine the spectral norm via $\|\mathcal{L}_j\| = \|W_j^H W_j\|$. We measure and compare the times spend in seconds for computing $\|\mathcal{L}_j\|$ via each of these approaches as well as the time spend in the remaining computations of the G-LR-ADI iteration.

The used parameters to set up the shift parameter generation and the LR-ADI iteration, as well as the results, are summarized in Table 3.1. It is evident that using the low-rank factors of the Lyapunov residual leads to a negligible amount of time spent for computing $\|\mathcal{L}_j\|$. The approaches using a Lanczos process or updated QR factorizations require considerably more computation time, where the time t_{QR} of the QR approach can even surpass the remaining computation time t_{ADI} of the iteration. This can also be seen in Figure 3.1 showing the progress of the computation times as the LR-ADI iteration proceeds for the *ocean* example. The timings t_{ADI} and t_W remain at an approximately constant level whereas t_{QR} and t_{Lan} clearly increase with t_{QR} becoming greater than t_{ADI} at some point of the iteration. To conclude, the new approach to compute the Lyapunov residual norm via the low-rank factors W_j substantially outperforms the existing approaches. We remark that this is only the case when the linear systems are solved exactly or to a high accuracy because otherwise $\|W_j^H W_j\|$ might not be a good indicator

for $\|\mathcal{L}_j\|$. In that case, the Lanczos approach appears to be an reasonable alternative. In the next section we continue with similar investigations regarding Sylvester equations.

3.3. The Low-rank ADI Iteration for Sylvester Equations

We now consider the numerical solution of Sylvester equations of the form

$$AX - XB = FG^T \tag{3.27}$$

with $A \in \mathbb{R}^{n\times n},\ B \in \mathbb{R}^{m\times m},\ F \in \mathbb{R}^{n\times r},\ G \in \mathbb{R}^{m\times r}$, and the sought solution $X \in \mathbb{R}^{n\times m}$. We assume that the spectra $\Lambda(A)$ and $\Lambda(B)$ are disjoint which ensures the existence of a unique solution of (3.27), see e.g., [151, 132]. We also assume that A, B are large, sparse matrices. In what follows we review the ADI iteration for (3.27) and its low-rank version in the next subsection. There, we also briefly discuss the shift parameter problem, stopping criteria, and the handling of generalized Sylvester equations. After that, we generalize the results for GCALEs given in Section 3.2.4. This includes new properties of the Sylvester residual and a reformulated low-rank algorithm.

3.3.1. Derivation, Shift Parameters and Stopping Criteria

The ADI scheme for (3.27) with two sets of shift parameters $\{\alpha_1, \dots, \alpha_{j_{\max}}\}$ and $\{\beta_1, \dots, \beta_{j_{\max}}\}$ is given by

$$\begin{aligned}(A - \beta_j I_n)X_{j-\frac{1}{2}} &= X_{j-1}(B - \beta_j I_m) + FG^T,\\ X_j(B - \alpha_j I_m) &= (A - \alpha_j I_n)X_{j-\frac{1}{2}} - FG^T,\end{aligned} \tag{3.28}$$

see [229]. By assuming $\beta_j \notin \Lambda(A),\ \alpha_j \notin \Lambda(B)\ \forall j$ the above two half steps can be written into one single step

$$\begin{aligned}X_j =&(A - \alpha_j I_n)(A - \beta_j I_n)^{-1}X_{j-1}(B - \beta_j I_m)(B - \alpha_j I_m)^{-1}\\ &+ (\beta_j - \alpha_j)(A - \beta_j I_n)^{-1}FG^T(B - \alpha_j I_M)^{-1}\end{aligned} \tag{3.29}$$

$$=\mathcal{C}(A,\ -\beta_j,\ \alpha_j)X_{j-1}\mathcal{C}(B,\ -\alpha_j,\ \beta_j) + T(\alpha_j, \beta_j), \tag{3.30}$$

where

$$T(\alpha, \beta) := (\beta - \alpha)(A - \beta I_n)^{-1}FG^T(B - \alpha I_m)^{-1} \tag{3.31}$$

and $\mathcal{C}$ denotes Cayley transformations (Definition 2.15). Alternatively, there is the following connection [212] between continuous and discrete-time Sylvester equations.

Lemma 3.10:

For every $\beta \notin \Lambda(A),\ \alpha \notin \Lambda(B),\ \alpha \neq \beta$ the CASE (3.27) is equivalent to the DASE (Definition 2.31)

$$X = \mathcal{C}(A, -\beta, \alpha)X\mathcal{C}(B, -\alpha, \beta) + T(\alpha, \beta). \tag{3.32}$$

Proof. The proof can be carried out along the lines of [135, Lemma 3.1.1]. □

Similarly to the CALE case, this represents another motivation for the iteration (3.29). As detailed in [162, 43], similar techniques as in Section 3.2 can be employed to derive a low-rank version of the above ADI iteration for (3.27). Setting $X_0 = 0$, exploiting the structure of the iterates given by the low-rank right hand side FG^T, and reordering the shifts, leads to the low-rank ADI iteration for (3.27) which is in the remainder referred to as factored ADI (fADI) iteration [19], [43, Algorithm 1], [162, Algorithm 2.1]. It follows the iterative scheme

$$V_1 = (A - \beta_1 I_n)^{-1} F, \quad S_1 = (B - \alpha_1 I_m)^{-H} G, \tag{3.33a}$$

$$V_j = V_{j-1} + (\beta_j - \alpha_{j-1})(A - \beta_j I_n)^{-1} V_{j-1}, \tag{3.33b}$$

$$S_j = S_{j-1} + \overline{(\alpha_j - \beta_{j-1})}(B - \alpha_j I_m)^{-H} S_{j-1} \tag{3.33c}$$

for $j > 1$. After $j_{\max}$ iteration steps of (3.33a)-(3.33c), low-rank solution factors $Z_{j_{\max}} \in \mathbb{C}^{n \times r j_{\max}}$, $Y_{j_{\max}} \in \mathbb{C}^{m \times r j_{\max}}$, $\Gamma_{j_{\max}} \in \mathbb{C}^{r j_{\max} \times r j_{\max}}$ are computed such that $Z_{j_{\max}} \Gamma_{j_{\max}} Y_{j_{\max}}^H \approx X$. In each iteration step, these low-rank solution factors are constructed via

$$Z_j = [Z_{j-1},\ V_j], \quad Y_j = [Y_{j-1},\ S_j], \quad \Gamma_j = \operatorname{diag}(\Gamma_{j-1}, (\beta_j - \alpha_j) I_r) \tag{3.33d}$$

i.e., r new columns are added to the low-rank factors Y_{j-1}, Z_{j-1}, and Γ_{j-1} is augmented by an $r \times r$ diagonal matrix. In terms of fADI iterates, the solution is constructed as

$$X_{j_{\max}} = \sum_{j=1}^{j_{\max}} (\beta_j - \alpha_j) V_j S_j^H.$$

which is for efficiency reasons never built explicitly. At the start, Z_0, Y_0, and Γ_0 are initialized as empty arrays. Note that for Lyapunov equations ($B = -A^T$, $G = -F$, $\beta_j = -\overline{\alpha_j}$) the above scheme gives the low-rank ADI (LR-ADI) iteration in Algorithm 3.1 [183, 161, 201, 42]. The main computational tasks are the solutions of the linear systems of equations with the shifted A, B matrices. As before we assume that we are able to employ sparse-direct or iterative Krylov subspace methods. Moreover, it should hold that $r \ll n$ since the column dimension r of F, G determines the number of right hand sides in the linear systems. A small value of r is by Theorem 2.41 also crucial for the existence of a low-rank solution of the Sylvester equation. The shift parameters $\{\alpha_j\}$, $\{\beta_j\}$ steer the convergence of the iteration and are discussed briefly in Subsection 3.3.1.

Generalized Sylvester Equations Generalized Sylvester equations

$$AXC - EXB = FG^T \tag{3.34}$$

with nonsingular $E \in \mathbb{R}^{n \times n}$ and $C \in \mathbb{R}^{m \times m}$ can also be dealt with by the fADI iteration (3.33). As for GCALEs, the key is to formally consider the equivalent standard Sylvester equation $\hat{A}C - X\hat{B} = \hat{F}\hat{G}^T$ with $\hat{A} := E^{-1}A$, $\hat{B} = BC^{-1}$, $\hat{F} = E^{-1}F$, and $\hat{G} = C^{-T}G$. Applying the same manipulations as those leading to the G-LR-ADI iteration

(Algorithm 3.1) for GCALEs, one finds that (3.33) for the equivalent standard Sylvester equation can be written as

$$V_1 = (A - \beta_1 E)^{-1} F, \quad S_1 = (B - \alpha_1 C)^{-H} G, \tag{3.35a}$$

$$V_j = V_{j-1} + (\beta_j - \alpha_{j-1})(A - \beta_j E)^{-1}(E V_{j-1}), \tag{3.35b}$$

$$S_j = S_{j-1} + \overline{(\alpha_j - \beta_{j-1})}(B - \alpha_j C)^{-H}(C^T S_{j-1}) \tag{3.35c}$$

which is from now on referred to as generalized fADI (G-fADI) iteration. For some of the following theoretical investigations using the equivalent standard Sylvester equations simplifies the occurring derivations.

Remark 3.11:

In [158] a generalized ADI scheme for (3.27) is introduced. There, "generalized" does not refer to generalized Sylvester equations (3.34). Instead, it reflects the case when the iteration (3.28) is modified such that different numbers of incremental steps are carried out w.r.t. A and B. Here, we do not follow this approach and restrict ourselves to the case when single steps w.r.t. A and B are employed as in (3.28), (3.29). ◊

The Error of the fADI Iteration and Shift Parameters The following lemma is helpful for providing insights into the ADI iteration (3.29) and represent a straightforward generalization of Lemma 3.1.

Lemma 3.12 (Generalization of Lemma 3.4, [202]):

The error after j iteration steps of the Sylvester ADI scheme (3.29) can be expressed as

$$X_j - X = \left[\prod_{i=1}^{j} \mathcal{C}(A, -\beta_i, \alpha_i)\right](X_0 - X)\left[\prod_{i=1}^{j} \mathcal{C}(B, -\alpha_i, \beta_i)\right]. \tag{3.36}$$

Proof. Combining Lemma 3.32 with (3.29) and using Lemma 3.2 immediately yields

$$X_j - X = \mathcal{C}(A, -\beta_j, \alpha_j)(X_{j-1} - X)\mathcal{C}(B, -\alpha_j, \beta_j)$$

and (3.36) follows from a repeated application of this identity. □

By taking norms in the error expression (3.36) we obtain

$$\frac{\|X_j - X\|}{\|X_0 - X\|} \leq \prod_{i=1}^{j} \|\mathcal{C}(A, -\beta_i, \alpha_i)\| \|\mathcal{C}(B, -\alpha_i, \beta_i)\|. \tag{3.37}$$

By similar steps as in Subsection 3.2.2 for the ADI iteration for GCALEs, this leads to the two-variable ADI parameter problem [230]

$$\min_{\substack{\alpha_i \in \mathbb{C} \\ \beta_i \in \mathbb{C}}} \max_{\substack{\lambda \in \Lambda(A) \\ \mu \in \Lambda(B)}} \prod_{i=1}^{j} \left| \frac{(\lambda - \alpha_i)(\mu - \beta_i)}{(\lambda - \beta_i)(\mu - \alpha_i)} \right| \tag{3.38}$$

for finding shifts that minimize the right hand side in (3.37). For generalized Sylvester equations (3.34), one has to use $\Lambda(A,\ E)$ and $\Lambda(B,\ C)$ in (3.38). Some approaches for solving (3.38) analytically, approximately, or heuristically can be found in [230, 233, 202, 162, 43]. For large-scale problems the often used approach [162, 43] is, similar to the Lyapunov case, to replace both spectra by small numbers $k_+^A,\ k_+^B$ and $k_-^A,\ k_-^B$ of Ritz and, respectively, inverse Ritz values w.r.t. $A,\ E$ and $B,\ C$. We postpone further discussions regarding shift parameters for (G-)fADI until Chapter 5 which is solely concerned with this issue.

Stopping Criteria The G-fADI iteration (3.35) can, similar to the G-LR-ADI iteration, be stopped when the residual

$$\mathcal{S}_j = AX_jC - EX_jB - FG^T \in \mathbb{C}^{n\times m}$$

is small enough, where one usually employs a condition like

$$\|\mathcal{S}_j\| \leq \tau\phi_j, \quad 0 < \tau \ll 1.$$

There, ϕ_j is a scaling factor for which a popular choice is, e.g., $\phi_j \equiv \|FG^T\|$. Further remarks regarding the efficient computation of the residual norm are given in the next subsection.

Alternatively, the relative changes of the low-rank solution factors $Z,\ Y$ can be used as stopping criteria, as it is also used in the G-LR-ADI iteration for Lyapunov equations [42]. There, one stops the iteration if

$$\frac{\|V_j\|_F}{\|Z_j\|_F} \leq \tau, \quad \frac{\|S_j\|_F}{\|Y_j\|_F} \leq \tau,$$

where $\|Z_j\|_F^2,\ \|Y_j\|_F^2$ can be computed cheaply using the accumulation technique presented in Subsection 3.2.3. However, both quantities can converge quite irregularly as in the Lyapunov case. Moreover, for the G-fADI iteration it is theoretically possible that this stopping criterion is fulfilled for the Z-, but not for the Y-factors, or vise-versa. In principle one could still continue the Y-iterations alone which would, after fulfilling the relative change criterion there as well, produce Z- and Y-factors with different column dimensions and a rectangular Γ-factor. For instance, if Z- and Y-sequences are stopped after j_Z and j_Y iterations, respectively, then $Z_{j_Z} \in \mathbb{C}^{n\times j_Z r}$, $Y_{j_Y} \in \mathbb{C}^{m\times j_Y r}$, and $\Gamma \in \mathbb{C}^{j_Z r\times j_Y r}$. We do not pursue this idea further and exclusively use the residual based stopping criterion in the remainder. For the Sylvester residual matrix we will in the following establish similar results as in the GCALE case in Section 3.2.3.

3.3.2. The Sylvester Residual Matrix within the fADI Iteration

The next lemma reveals that the Sylvester residual matrix within the fADI iteration can be written in a similar expression as the error in (3.36).

Lemma 3.13 (Generalization of [135, Lemma 3.5.2], [94, Lemma 5.3]):

For the residual after j iteration steps of (3.29) it holds

$$\mathcal{S}_j = AX_j - X_jB - FG^T = \left[\prod_{i=1}^{j} \mathcal{C}(A, -\beta_i, \alpha_i)\right] \mathcal{S}_0 \left[\prod_{i=1}^{j} \mathcal{C}(B, -\alpha_i, \beta_i)\right]. \tag{3.39}$$

Proof. Using $AX_j - X_jB - FG^T = A(X_j - X) - (X_j - X)B$ yields with (3.36) the sought expression because A and $\mathcal{C}(A, \beta_j, \alpha_j)$, as well as B and $\mathcal{C}(B, \alpha_j, \beta_j)$, commute for all j by Lemma 3.2. □

The following theorem generalizes Theorem 3.5 for the G-LR-ADI iteration applied to GCALEs [33] and is the main contribution of the current section.

Theorem 3.14 (Generalization of Theorem 3.5, [33, Theorem 1]):

Assume that $\operatorname{rank}(G) = \operatorname{rank}(F) = r$. Then the residual after j iteration steps of the G-fADI iteration (3.35) is of rank at most r and can be factored as

$$\mathcal{S}_j := AX_jC - EX_jB - FG^T = -W_jT_j^H. \tag{3.40}$$

The residual factors $W_j \in \mathbb{C}^{n\times r}$, $T_j \in \mathbb{C}^{m\times r}$ and G-fADI iterates V_j, S_j are given by

$$\begin{aligned}
W_j &:= (A - \alpha_j E)V_j && T_j := (B - \beta_j C)^H S_j && \text{(3.41a)}\\
&= W_{j-1} + \gamma_j E V_j && = T_{j-1} - \overline{\gamma_j} C^T S_j && \text{(3.41b)}\\
&= W_0 + EZ_j\Gamma_j(\mathbf{1}_j \otimes I_r), && = T_0 - C^T Y_j \Gamma_j^H (\mathbf{1}_j \otimes I_r), && \text{(3.41c)}\\
V_j &= (A - \beta_j E)^{-1} W_{j-1}, && S_j = (B - \alpha_j C)^{-H} T_{j-1}, && \text{(3.41d)}
\end{aligned}$$

where $\gamma_j := \beta_j - \alpha_j$, $W_0 := F$, $T_0 := G$. ◊

Proof. For simplification we go through the proof for the fADI iteration applied to the equivalent Sylvester equation defined by $\hat{A} = E^{-1}A$, $\hat{B} = BC^{-1}$, $\hat{F} = E^{-1}F$, and $\hat{G} = C^{-T}G$. Since fADI (3.33) starts from $X_0 = 0$, it holds $\hat{\mathcal{S}}_0 = \hat{A}X_j - X_j\hat{B} - \hat{F}\hat{G}^T = -\hat{F}\hat{G}^T$ and the expression (3.39) yields already $\hat{\mathcal{S}}_j = -\hat{W}_j\hat{T}_j^H$ with

$$\hat{W}_j = \left[\prod_{i=1}^{j} \mathcal{C}(\hat{A},\ -\beta_i,\ \alpha_i)\right] \hat{F}, \quad \hat{T}_j = \left[\prod_{i=1}^{j} \mathcal{C}(\hat{B},\ -\alpha_i,\ \beta_i)^H\right] \hat{G}. \tag{3.42}$$

This shows that $\operatorname{rank}(\hat{\mathcal{S}}_j) = r$ if $\beta_j,\ \alpha_j \notin \Lambda(\hat{A}) \cup \Lambda(\hat{B})$. If $\alpha_j \in \Lambda(\hat{A})$ or $\beta_j \in \Lambda(\hat{B})$ for some j, the inverses still exist but $\hat{A} - \alpha_j I_n$, or respectively $\hat{B} - \beta_j I_m$, is singular, such that the column rank of $\hat{W}_j$ or $\hat{T}_j$ can be smaller than r. Following the manipulations in Theorem 3.5, [33, Theorem 1], the increments V_j, S_j in (3.33a)–(3.33c) can be expressed

as

$$\begin{aligned} V_j &= (\hat{A} - \alpha_{j-1} I_n)(A - \beta_j I_n)^{-1} V_{j-1} \\ &= (\hat{A} - \beta_j I_n)^{-1}(\hat{A} - \alpha_{j-1} I_n)(\hat{A} - \alpha_{j-2} I_n)(\hat{A} - \beta_{j-1} I_n)^{-1} V_{j-2} \\ &= (\hat{A} - \beta_j I_n)^{-1} \mathcal{C}(\hat{A}, -\beta_{j-1}, \alpha_{j-1})(\hat{A} - \alpha_{j-2} I_n) V_{j-2} \\ &= \ldots = (\hat{A} - \beta_j I_n)^{-1} \left[\prod_{i=1}^{j-1} \mathcal{C}(\hat{A},\ \beta_i,\ \alpha_i) \right] \hat{F} = (\hat{A} - \beta_j I_n)^{-1} \hat{W}_{j-1}, \end{aligned} \tag{3.43a}$$

$$S_j = (\hat{B} - \alpha_j I_m)^{-H} \left[\prod_{i=1}^{j-1} \mathcal{C}(\hat{B},\ -\alpha_i,\ \beta_i)^H \right] \hat{G} = (\hat{B} - \alpha_j I_n)^{-H} \hat{T}_{j-1}. \tag{3.43b}$$

A comparison of (3.42) with (3.43) yields $\forall j \geq 1$

$$\hat{W}_j = (\hat{A} - \alpha_j I_n) V_j, \quad \hat{T}_j = (\hat{B} - \beta_j I_n)^H S_j.$$

Consequently,

$$\hat{W}_j = (\hat{A} - \alpha_j I_n)(\hat{A} - \beta_j I_n)^{-1} \hat{W}_{j-1} = \hat{W}_{j-1} + \gamma_j V_j = \hat{W}_0 + Z_j \Gamma_j (\mathbf{1}_j \otimes I_r), \tag{3.44a}$$

$$\hat{T}_j = (\hat{B} - \beta_j I_m)^H (\hat{B} - \alpha_j I_m)^{-H} \hat{T}_{j-1} = \hat{T}_{j-1} - \overline{\gamma_j} S_j = \hat{T}_0 - Y_j \Gamma_j^H (\mathbf{1}_j \otimes I_r), \tag{3.44b}$$

where the right most expressions are obtained from setting $\hat{W}_0 := \hat{F}$, $\hat{T}_0 := \hat{G}$ and using the definition of the low-rank solution factors Z_j, Y_j and Γ_j. The desired formulas (3.41) for the low-rank factors W_j, T_j of $\mathcal{S}_j$ and the iterates V_j, S_j corresponding to GCASEs are easily established from $\mathcal{S}_j = E\hat{\mathcal{S}}_j C = E\hat{W}_j \hat{T}_j^H C = W_j T_j^H$ and transforming the linear systems in (3.43). □

Theorem 3.14 reveals an equivalent formulation of the G-fADI iteration (3.35), where the low-rank factors W_j, T_j of the residual matrix $\mathcal{S}_j$ are an integral part of the iteration. This reformulated G-fADI iteration is illustrated in Algorithm 3.4 which uses the recursive expression (3.41b) for W_j, T_j and a stopping criterion based on the scaled residual norm. A generalization of Corollary 3.8 for the G-LR-ADI iteration (Algorithm 3.2) can be formulated in a straightforward way, i.e., the G-fADI iteration implicitly solves the Sylvester equation $AXC - EXB - W_j S_j^H = 0$ in each iteration step. The factorization (3.40) of $\mathcal{S}_j$ enables a cheap and efficient computation of the residual norm $\|\mathcal{S}_j\| = \|W_j T_j^H\|$ in the spectral norm via

$$\|W_j T_j^H\| = \sigma_{\max}(W_j T_j^H) = \sqrt{\lambda_{\max}(T_j W_j^H W_j T_j^H)} = \sqrt{\lambda_{\max}((T_j^H T_j)(W_j^H W_j))}. \tag{3.45}$$

There, we used that the nonzero eigenvalues of $\mathcal{AB}$ are the same as those of $\mathcal{BA}$ for all $\mathcal{A} \in \mathbb{C}^{n \times m}$, $\mathcal{B} \in \mathbb{C}^{m \times n}$, see, e.g., [128, Theorem 1.32]. This essentially reduces the residual norm computation to the computation of the largest eigenvalue of the $r \times r$ matrix $(T_j^H T_j)(W_j^H W_j)$. Although its eigenvalues are obviously real, roundoff errors might introduce very small imaginary parts, therefore it is wise to take the square root only over the real parts. This approach is slightly cheaper than the approach

Algorithm 3.4: Reformulated generalized factored ADI (G-fADI) iteration for (3.34)

Input : $A,\ B,\ E,\ C,\ F,\ G$ as in (3.34), shift parameters $\{\alpha_1, \ldots, \alpha_{j_{\max}}\}$, $\{\beta_1, \ldots, \beta_{j_{\max}}\}$, and tolerance $0 < \tau \ll 1$.

Output: $Z_{j_{\max}} \in \mathbb{C}^{n \times r j_{\max}}$, $Y_{j_{\max}} \in \mathbb{C}^{m \times r j_{\max}}$, $\Gamma_{j_{\max}} \in \mathbb{C}^{r j_{\max} \times r j_{\max}}$ such that $Z_{j_{\max}} \Gamma_{j_{\max}} Y_{j_{\max}}^H \approx X$.

$W_0 = F,\ T_0 = G,\ Z_0 = \Gamma_0 = Y_0 = [\,],\ k = 1.$
while $\|W_{j-1} T_{j-1}^H\| \geq \tau \|F G^T\|$ **do**
  $V_j = (A - \beta_j E)^{-1} W_{j-1}, \quad S_j = (B - \alpha_j C)^{-H} T_{j-1}.$
  $\gamma_j := \beta_j - \alpha_j.$
  $W_j = W_{j-1} + \gamma_j E V_j, \quad T_j = T_{j-1} - \overline{\gamma_j} C^T S_j.$
  $Z_j = [Z_{j-1},\ V_j],\ Y_j = [Y_{j-1},\ S_j],\ \Gamma_j = \operatorname{diag}(\Gamma_{j-1},\ \gamma_j I_r).$
  $j = j + 1.$

proposed in [32] which uses a thin QR decomposition of either W_j or T_j. Beyond that, it is significantly cheaper than, e.g., estimating the spectral norm indirectly via a Power iteration or a Lanczos process applied to $\mathcal{S}_j^H \mathcal{S}_j$ or $\mathcal{S}_j \mathcal{S}_j^H$. There, the main work for each Lanczos iteration would be matrix-vector products $y = \mathcal{S}_j x,\ x \in \mathbb{C}^m$, and $z = \mathcal{S}_j^H y,\ y \in \mathbb{C}^n$. These products can be formed without an explicit construction of $\mathcal{S}_j$, requiring only matrix-vector products with the involved matrices and their transposes.

Remark 3.15:

Alternative approaches based on the product SVD, e.g. [80], can also be used to efficiently compute $\|W_j T_j^H\|$. These methods are constructed for computing the SVD of products of matrices to a very high accuracy in a numerically robust way. For using the residual norm as stopping criterion, only $\sigma_{\max}(W_j T_j^H)$ is of interest and one is usually satisfied when the exponent ν in $\|W_j T_j^H\| \approx c \cdot 10^{-\nu}$ is computed exactly. Hence, computing $\|W_j T_j^H\|$ by the approach (3.45) is sufficient for our purposes, and also never failed to deliver an accurate estimate of $\|\mathcal{S}_j\|$. ◊

The approach employing a Lanczos process might, however, still be useful if the low-rank solution is improved by Galerkin projection approaches [43] since then (3.41) does not hold anymore. Some numerical evidence that computing the residual norm via (3.45) is more efficient than using Lanczos can be found in Section 3.3.4, [32]. Also, if the linear systems (3.41d) are solved inexactly, considerations similar to Remark 3.7 can be drawn as the computed quantity $\|W_j S_j^H\|$ might be no longer equal to $\|\mathcal{S}_j\|$. Other approaches for computing $\|\mathcal{S}_j\|$ similar to the ones mentioned in Subsection 3.2.3 are also possible but unlikely to be more efficient than (3.45).

The low-rank solution factors Z, Y are solutions of GCASEs.

With the help of Theorem 3.14, the following results can be derived which constitutes a generalization of (3.24a) in Corollary 3.9, [235, Lemma 3.1], [234, Lemma 5.12].

Corollary 3.16 (Sylvester Equations for the fADI low-rank solution factors):
The low-rank solution factors Z_j, Y_j constructed after j steps of the G-fADI iteration (Algorithm 3.4) satisfy the GCASEs

$$AZ_j - EZ_j\boldsymbol{\sigma}_j^{\alpha} = W_j G_j^T, \qquad B^T Y_j - C^T Y_j \boldsymbol{\sigma}_j^{\beta} = T_j G_j^T \tag{3.46}$$

$$\text{with} \quad \boldsymbol{\sigma}_j^{\beta} := \operatorname{diag}(\alpha_1 I_r, \ldots, \alpha_j I_r) - \boldsymbol{\gamma_j}, \qquad \boldsymbol{\sigma}_j^{\beta} := \overline{\operatorname{diag}(\beta_1 I_r, \ldots, \beta_j I_r) + \boldsymbol{\gamma_j}}$$

$$\boldsymbol{\gamma_j} := \begin{bmatrix} 0 & & \\ \gamma_2 & 0 & \\ \vdots & \ddots & \\ \gamma_j & \cdots & \gamma_j \; 0 \end{bmatrix} \otimes I_r \in \mathbb{C}^{jr \times jr}, \qquad G_j := (\mathbf{1}_j \otimes I_r) \in \mathbb{R}^{jr \times r},$$

and W_j, T_j are the low-rank factor of the residual $\mathcal{S}_j$. ◊

Proof. The result can be established in analogy to the proof of Corollary 3.9. By (3.41a) in Theorem 3.14 it holds $AV_i = W_j + \alpha_i E V_i$, $i = 1, \ldots, j$. Inserting (3.41b) in these relations reveals

$$AV_i = W_j + \alpha_i E V_i - E \sum_{k=1}^{j-i} \gamma_{j-k+1} V_{j-k+1}, \quad i = 1, \ldots, j-1$$

such that

$$A[V_1, \ldots, V_j] = [W_j, \ldots, W_j] + E[V_1, \ldots, V_j] \begin{bmatrix} \alpha_1 & & \\ -\gamma_2 & \alpha_2 & \\ \vdots & \ddots & \\ -\gamma_j & \cdots & -\gamma_j \; \alpha_j \end{bmatrix} \otimes I_r$$

from which the GCASE for Z_j in (3.46) can be easily deduced. The GCASE for Y_j is established in the same way using the relations for S_j, T_j in (3.41). □

3.3.3. Special Cases of Generalized Sylvester Equations

We already mentioned that for generalized Lyapunov equations ($B = -A^T$,$C = E^T$, $G = -F$, $\alpha_j = -\overline{\beta_j}$), the G-fADI iteration reduces to the G-LR-ADI iteration. In this section we discuss further special cases of the general Sylvester equation (3.34) which also lead to specially tailored variants of the G-fADI iteration.

Cross Gramian Sylvester Equation

Choosing $B = -A$, $C = E$ leads to GCASEs of the form

$$AXE + EXA = FG^T \tag{3.47}$$

which occur, e.g., in cross Gramian model order reduction [215]. There, the defining matrices A, E, F, G represent a linear, time-invariant control system (2.5). We use the usual assumption that this underlying system defining (3.47) is asymptotically stable, i.e., $\Lambda(A,\ E) \subset \mathbb{C}_-$. The reasonable choice $\beta_j = -\alpha_j$ leads to the iteration

$$V_j = (A + \alpha_j E)^{-1} W_{j-1}, \qquad S_j = -(A + \alpha_j E)^{-H} T_{j-1}, \tag{3.48a}$$

$$W_j = W_{j-1} - 2\alpha_j E V_j, \qquad T_j = T_{j-1} + 2\overline{\alpha_j} E^T S_j. \tag{3.48b}$$

If the linear system for V_j is solved using a sparse LU decomposition $LU = (A + \alpha_j E)$, the LU factors can be reused for solving the second linear system for S_j since $U^H L^H = (A + \alpha_j E)^H$.

Remark 3.17:

Although we mainly employ sparse direct solvers for the solution of the linear systems, a few remark regarding iterative solvers are in order. There exist some iterative Krylov subspace methods which can be used to solve the adjoint systems in (3.48) simultaneously, i.e., only one run of the particular iterative solver is required to obtain both V_j and S_j. This holds especially in the case $r = 1$, where Krylov subspace methods based on the two-sided Lanczos process can be applied. Prominent methods belonging to this class are the bi-conjugate gradient (BiCG) [97] and quasi-minimal residual (QMR) [101, 168] methods. Next to the ability to solve adjoint linear systems simultaneously, another advantage of BiCG, QMR is that both are short-recurrence methods which essentially fixes their computational effort and memory requirements to a constant level through their whole progress. However, due to the inherent two-sided Lanczos process, they are susceptible to breakdowns for which countermeasures can be found, e.g., in [8, 9, 99]. The case $r > 1$ can be tackled by proceeding column-wise through W_{j-1} and T_{j-1} or by employing block versions [179, 100]. Other, less known methods also capable of solving both linear system simultaneously are the GLSQR [188, 110] and unsymmetric MINRES [204] methods. ◊

Lyapunov Equations with Unsymmetric Inhomogeneity

A similar special case is the matrix equation ($B = -A^T$, $C = E^T$)

$$AXE^T + EXA^T = FG^T \tag{3.49}$$

which might be considered as generalized Lyapunov equation with an unsymmetric right hand side. A sufficient condition for a unique solution is $\Lambda(A,\ E) \subset \mathbb{C}_-$. Setting $\beta_j = -\overline{\alpha_j}$ yields

$$V_j = (A + \overline{\alpha_j}E)^{-1} W_{j-1}, \qquad S_j = -(A + \overline{\alpha_j}E)^{-1} T_{j-1}, \tag{3.50a}$$

$$W_j = W_{j-1} - 2\,\mathrm{Re}\,(\alpha_j)\, EV_j, \qquad T_j = T_{j-1} + 2\,\mathrm{Re}\,(\alpha_j)\, ES_j. \tag{3.50b}$$

Since the coefficient matrices in both linear systems are identical, we can solve for $V_j,\ S_j$ at once via

$$[V_j,\ S_j] = (A + \overline{\alpha_j}E)^{-1}[W_{j-1},\ T_{j-1}]$$

which requires, if a sparse direct solver is applied, only one factorization of the coefficient matrix. In Section 7.3, we will encounter GCALEs of the form (3.49).

Discrete-Time Lyapunov Equations

It is obvious that a GCASE (3.34) can be seen as a discrete-time Sylvester equation and, thus, also those equations can be solved by the G-fADI iteration. Choosing $C = A^T$,

$B = E^T$, and $G = -F$ leads to a generalized discrete-time Lyapunov equation (GDALE, see Definition 2.26)

$$AXA^T - EXE^T = -FF^T \tag{3.51}$$

which has a unique symmetric positive (semi)definite solution if $|\lambda_j| < 1$ for all $\lambda_j \in \Lambda(A,\ E)$ by Lemma 2.28. Often, $A,\ E,\ F$ are the matrices defining a discrete-time, linear, time-invariant system (2.6). There exists already some work on solving large-scale GDALEs, e.g., by Krylov subspace [203], Smith [29, 165], or ADI type methods [27],[203, Section 6.3]. Here we follow a rather unconventional approach by solving (3.51) with the G-fADI iteration for generalized Sylvester equations. There, some simplifications occur as follows, where we assume that $0 \notin \Lambda(A,\ E)$ and $\alpha_j \neq 0$, $|\alpha_j| < 1\ \forall j$. Since $\Lambda(B,\ C) = \Lambda(E^T,\ A^T) = 1/\overline{\Lambda(A,\ E)}$, we set $\beta_j = 1/\overline{\alpha_j}$. Then $V_j,\ W_j$ are given by

$$\begin{aligned} V_j &= (A - \beta_j E)^{-1} W_{j-1} = (A - \tfrac{\alpha_j}{|\alpha_j|^2} E)^{-1} W_{j-1}, \\ W_j &= W_{j-1} + \alpha_j \tfrac{1-|\alpha_j|^2}{|\alpha_j|^2} E V_j. \end{aligned}$$

For $S_j,\ T_j$ first note that

$$AV_j = \tfrac{\alpha_j}{|\alpha_j|^2} E V_j + W_{j-1} = \tfrac{1}{1-|\alpha_j|^2} W_j - \tfrac{|\alpha_j|^2}{1-|\alpha_j|^2} W_{j-1}.$$

Then one finds for the first iteration

$$\begin{aligned} S_1 &= (B - \alpha_1 C)^{-H} G = (\overline{\alpha_1} A - E)^{-1} F = -\tfrac{1}{\overline{\alpha_1}} V_1, \\ T_1 &= -F + \tfrac{1-|\alpha_1|^2}{\alpha_1} A S_1 = -F + \tfrac{1-|\alpha_1|^2}{|\alpha_1|^2} A V_1 = -\tfrac{1}{|\alpha_1|^2} S_1. \end{aligned}$$

Subsequently applying this procedure for $j = 2, 3, \ldots$ yields

$$S_j = \tfrac{\theta_{j-1}}{\overline{\alpha_j}} V_j, \quad T_j = -\theta_j W_j$$

with $\theta_j := \left(|\alpha_1|^2 \cdot \ldots \cdot |\alpha_j|^2\right)^{-1}$, $\theta_0 = 1$. Hence, $S_j,\ T_j$ as well as the solution factor Y_j are not required. Accumulating the constants in front of S_j in the above expressions into the Γ-factor leads to the iteration

$$V_j = (A - \tfrac{\alpha_j}{|\alpha_j|^2} E)^{-1} W_{j-1}, \quad W_j = W_{j-1} + \alpha_j \tfrac{1-|\alpha_j|^2}{|\alpha_j|^2} E V_j, \tag{3.52a}$$

$$Z_j = [Z_{j-1}, V_j], \quad \Gamma_j = \operatorname{diag}\left(\Gamma_{j-1}, (1 - |\alpha_j|^2)\theta_j I_r\right),\ \theta_j = \tfrac{\theta_{j-1}}{|\alpha_j|^2}. \tag{3.52b}$$

The spectral norm of the residual can be computed via $\|\mathcal{L}_j\| = |\theta_j| \|W_j\|^2$. The iteration (3.52) can be rewritten such that the coefficient matrices in the linear systems are $|\alpha_j|^2 E - \alpha_j A$ or $\alpha_j A - E$. This will essentially introduce other constants in the defining equations for $V_j,\ W_j$ and Γ_j. Which choice will be the best, e.g., w.r.t. numerical robustness, might be a topic for further research. Note that this iteration is different from the ADI scheme for (3.51) presented in [27, 203]. There, the whole low-rank solution

factor Z_{j-1} has to be processed similar to the iteration (3.8) and no residual factors are incorporated.

The construction of θ_j constitutes a weakness of the above iteration since it can easily become extremely large if $|\alpha_j| \ll 1$ for some shifts. This was also observed in some numerical examples. Moreover, the case $|\alpha_j| \approx 1$ can lead to cancellation. We plan to pursue the circumvention of these issues in future work regarding ADI methods for discrete-time Lyapunov equations.

If the right hand side of (3.51) is FG^T, similar techniques as for (3.49) in Section 3.3.3 can be applied to construct a modified version of iteration (3.52).

3.3.4. Numerical Examples

Efficient Computation of the Norm of the Sylvester Residual

Similar to the experiments in Section 3.2.6, the novel strategy for the residual norm computation in the G-fADI iteration is briefly investigated. To define a GCASE we use two different versions of the *ifiss* example (cf. Section 2.4: the matrices $-A$, E and B, C are taken from the *ifiss16k* ($n = 16641$) and, likewise, the smaller *ifiss4k* example ($m = 4225$). The complete GCASE example is denoted by *ifiss16k/4k*. This setup is motivated by the Sylvester example in [21], where the authors use different versions of the *rail* example to define a GCASE.

We employ the G-fADI iteration (Algorithm 3.4) and use $J = L = 30$ heuristic shift parameters proposed in [162, 43] which are generated from $k_+^A = k_+^B = 10$, $k_-^A = k_-^B = 20$ Ritz and inverse Ritz values. The iteration is terminated when $\varepsilon_j := \|\mathcal{S}_j\|/\|FG^T\| < \tau = 10^{-10}$, where the Sylvester residual norm is computed using the novel approach (3.45) which takes the low-rank structure of $\mathcal{S}_j$ from Theorem 3.14 into account. For comparison we also apply a Lanczos process to $\mathcal{S}_j^H\mathcal{S}_j$ which was set up as in the GCALE examples before. Without taking the residual computation into account, the G-fADI iteration required t_{ADI}=31.2 seconds in j_{it}=89 iteration steps leading to $\varepsilon_j = 7.22 \cdot 10^{-11}$. Using (3.45) for computing $\|\mathcal{S}_j\|$ took in total 0.0771 seconds which amounts to a negligible fraction of t_{ADI}. In comparison, using a Lanczos process required in total 63.2 seconds, i.e., roughly twice the amount of t_{ADI} would be spent in computing the residual norm. In Figure 3.2, the time for computing the residual norm is plotted along with the timings of the remaining computations for each iteration step j and clearly, the time $t_{\text{Lan},j}$ for the Lanczos approach increases notably with j whereas the time $t_{WS,j}$ for the novel approach (3.45) stays approximately constant. The Lanczos timings $t_{\text{Lan},j}$ are also visibly close to the remaining computation times $t_{\text{ADI},j}$.

Modifications of the G-fADI Iteration for Special Sylvester Equations

Here we investigate the proposed versions of the G-fADI iteration for the special cases of Sylvester equations in Section 3.3.3. We use the following examples for comparing the specially tailored iterative schemes (3.48), (3.50), and (3.52) to the standard G-fADI iteration.

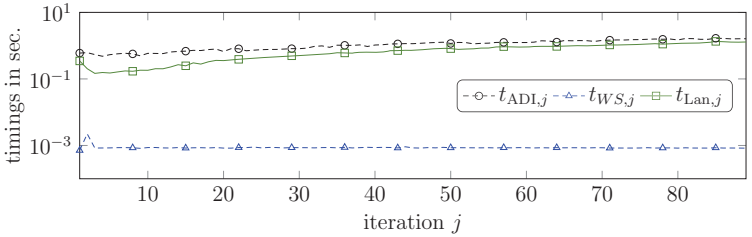


Figure 3.2.: Time (in seconds) needed for computing the Sylvester residual norm with different approaches and for the remaining computations in the G-fADI iteration against the step number j for the *ifiss16k/4k* example.

Table 3.2.: Numerical results obtained with standard and structure exploiting G-fADI Iteration.

Matrix Equation	Parameters k_+^A, k_-^A, J	Results j_{iter}	$\varepsilon_{j_{\mathrm{iter}}}$	t_{standard}	t_{modified}
cross Gramian (3.47)	10, 20, 30	110	$4.7{\cdot}10^{-11}$	71.9	70.5
unsymmetric GCALE (3.49)	10, 20, 30	111	$4.0{\cdot}10^{-11}$	74.1	54.1
DALE (3.51)	10, 0, 10	68	$9.4{\cdot}10^{-11}$	12.6	6.4

To define a cross Gramian Sylvester equation of the form (3.47), we use the matrices $A,\ E$ from example *ifiss16k* and $B = -A$, $C = E$. The resulting equation is dealt with by the specially tailored iteration (3.48) of Section 3.3.3.

Similarly, using the same matrices $A,\ E$, but now $B = -A^T$, $C = E^T$ leads to an equation of the form (3.49) such that the iteration (3.50) is applied as well.

To generate a DALE (3.51) to be solved by the iteration (3.52), we take the matrix $A = C^T$ from [203, Example 1] which is of dimension $n = 50000$, skew-symmetric, and tridiagonal with -0.49, 0, 0.49 on the lower, main, and upper diagonal. Furthermore, $B = E = I_n$ and $F = -G$ consist of the first $r = 2$ canonical vectors.

As tolerance for termination the iterations we use again $\tau = 10^{-10}$. Table 3.2 summarizes the used parameters, the required number of iterations j_{iter}, the final residual norm $\varepsilon_{j_{\mathrm{iter}}}$, as well as the computation times t_{standard} and t_{modified} for the standard and modified G-fADI iterations, respectively. Note that these timings incorporate also the times needed to compute the residual norm and, since for these special cases no β shifts are required, no values for $k_\pm^B$, L are listed.

For the cross Gramian Sylvester equation there is only a negligible difference between the timings of the G-fADI iteration and scheme (3.48). Note that the adjoint linear systems in the iteration are solved separately since computing and reusing LU factors of $A - \alpha E$ as mentioned above took more than twice as long, i.e. around 193 seconds. The reason for those large differences is, to our knowledge, that the MATLAB backslash

command uses different techniques to compute the LU factors than the `lu` command. In other software and hardware environments it can be possible that reusing the LU factors pays off.

For the matrix equation (3.49), the application of the adapted iteration (3.50) leads also to a significant reduction of the computation time. This is mainly because the factorization of the shifted linear system has to be computed only once for constructing both V_j and S_j.

A clear decrease of the run times is also apparent for the DALE example when applying (3.52) instead of the G-fADI iteration. However, in some cases the algorithm broke down when some of the α_j were too small in magnitude which resulted in an overflow when computing the value θ_j. Hence, we set $k_-^A = 0$ and used only the Ritz values approximating the eigenvalues with the largest magnitude.

3.4. Conclusions

The low-rank ADI iteration for computing low-rank solution factors of large-scale generalized Lyapunov and Sylvester equations has been reviewed. Special emphasis was put on considerations regarding the matrix equation's residual. It was shown that the rank of the residual in low-rank ADI methods is at most equal to the rank r of the inhomogeneity. Hence, the residuals can be written in terms of a low-rank factorization. We also proposed explicit and easy ways to obtain the low-rank residual factors which, in fact, turned out to appear directly within the iteration. These relations lead to the derivation of reformulated versions of the G-LR-ADI and G-fADI iterations for GCALEs and GCASEs, respectively, where the low-rank factors of the residual are an integral part of the process. As main contribution of this chapter, the low-rank residual factors enable a very cheap and efficient computation of the residual norm during the iteration. In this way the computation of the residual norm only requires a negligible amount of effort which is significantly cheaper than traditional approaches for estimation the residual norms as the numerical experiments confirm. However, the proposed low-rank expressions for the residuals do not hold if the occurring linear systems are solved inexactly. This issue is subject of current research [202, 219] which also includes investigations regarding the minimum required accuracy of the approximate linear system solves such that convergence of the low-rank ADI methods is still maintained.

We also discussed certain modification of the G-LR-ADI and G-fADI iterations to deal with problems having a special structure, e.g., GCALEs arising from the study of second order control systems. For special GCASEs, e.g., cross-Gramian Sylvester and GDALEs, new modified low-rank ADI methods have been proposed. Numerical experiments with those methods reveal that additional, considerable computational savings can be gained from this structure exploitation. Further research in this direction includes especially to make the proposed low-rank ADI iteration for GDALEs more robust with respect to shift parameters of small magnitude.

CHAPTER 4

EFFICIENT HANDLING OF COMPLEX ADI SHIFT PARAMETERS

Contents

ADI based methods require a number of shift parameters to attain a fast convergence. These shifts are in one way or another related to the spectra of the coefficient matrices (or matrix pairs), where we explicitly focus on the case when there are complex eigenvalues in these spectra. This might lead to complex shift parameters for the ADI iteration, which will produce a complex (low-rank) solution, or complex low-rank solution factors. Since the original matrix equations we consider in this thesis involve only real matrices, but the ADI will then contain complex arithmetic operations and storage, which yields higher computational costs as in the real case, this is an undesirable property. Therefore, our goal is to investigate strategies for computing real low-rank solution factors with modified versions of the low-rank and factored ADI iteration, which should employ no, or at least only an absolutely necessary amount of complex arithmetic operations and storage. In the following section we investigate this issue for the G-LR-ADI iteration for GCALEs (Section 3.2). We start by introducing the concept of proper shifts and give a brief motivation why complex shift parameters might be beneficial for the convergence

speed of the iteration. After that we review two existing approaches to deal with complex shift parameters, where we give a slight improvement of a completely real G-LR-ADI iteration [42, 161, 183]. In Section 4.1, we establish some novel interconnections of the data in the G-LR-ADI iteration w.r.t. complex conjugate shifts. A careful usage of these interconnections will lead to a further modification of the algorithm, which employs a considerably reduced amount of complex arithmetic operations as well as memory demand and produces real low-rank solution factors in the end. Numerical experiments illustrate the performance of this new modified algorithm and compare it to the other approaches. Similar considerations for the G-fADI iteration for generalized Sylvester equations are carried out in Section 4.2, where novel interconnections of complex iterates are proved. A modified G-fADI iteration that computes real low-rank solution factors at a reduced effort is presented. The adaption of this approach to the special GCASEs (cf. Section 3.3.3) is given in Section 4.2.2. Selected numerical examples shown the efficiency of this modified G-fADI iteration in Section 4.2.3 and conclusions are given in Section 4.3.

4.1. Complex Shift Parameters in the G-LR-ADI Iteration for Lyapunov Equations

4.1.1. Proper Shift Parameters

All the considerations and approaches discussed in this section require the natural and important convention that the used sets of shift parameters are *proper* in the sense of the following definition.

Definition 4.1:

(Proper Complex Sets) A complex set $\mathcal{A} \subset \mathbb{C}$ is said to be *proper* if it is closed under complex conjugation ($\mathcal{A} = \overline{\mathcal{A}}$) and if its elements are ordered in the form

$$\{\alpha_1, \ldots, \alpha_J\} = \{\mu_1, \ldots, \mu_L\}, \ L \leq J,$$

where μ_j is either a real number $\mu_j = \alpha_j$ or a pair of complex conjugate numbers $\mu_j = \{\alpha_j, \ \alpha_{j+1} = \overline{\alpha_j}\}$ for $j = 1, \ldots, L$. ◊

This convention is obviously not a restrictive, but very natural, assumption since complex eigenvalues of a real matrix also appear in complex conjugate pairs. Assuming that the set of used ADI shift parameters is proper directly implies that, if for every complex shift also its conjugate is used, $ZZ^H \in \mathbb{R}^{n \times Jm}$ although Z will in general be a complex matrix. Hence, it is not advised to terminate ADI in between a pair of complex conjugated shift parameters. When we speak in the following of proper shifts for the ADI iteration for GCALEs, this automatically implies $\mathcal{A} \subset \mathbb{C}_-$.

4.1.2. A Short Motivation for the Need of Complex Shift Parameters

If the spectrum of the matrices defining the GCALEs contains complex eigenvalues, then the ADI shift parameters, which are in one way or another related to the spectrum, can be complex as well. This can also be seen from the associated optimization problem (3.14) which can have complex solutions. The following small example shows that for matrices with complex eigenvalues, a larger error reduction can be achieved by using complex shifts compared to real shifts. Let for this

$$A = \begin{bmatrix} -1 & 0 & 0 \\ 0 & -\frac{1}{2} & 1 \\ 0 & -1 & -\frac{1}{2} \end{bmatrix} \quad \text{with} \quad \Lambda(A) = \left\{-1, -\tfrac{1}{2} \pm \jmath\right\}.$$

Since the complex eigenvalues occur complex conjugate pairs and by the demand for proper shift parameters, we compare two complex conjugated shifts versus two real shifts. Recall from Lemma 3.4 that the error reduction at iteration step j of the ADI iteration is related to the norm of the product of j Cayley transformations. Hence, for our example, we use

$$f(\alpha_1,\ \alpha_2) := \|\mathcal{C}_2\mathcal{C}_1\| = \|(A+\alpha_2 I)^{-1}(A-\alpha_2 I)(A+\alpha_1 I)^{-1}(A-\alpha_1 I)\|,$$

where $\alpha_1,\ \alpha_2 \in \mathbb{R}_-$ for one instance, and $\alpha_1 = \operatorname{Re}(\alpha_1) + \jmath \operatorname{Im}(\alpha_1) \in \mathbb{C}_-$, $\alpha_2 = \overline{\alpha_1}$ in a second test. Note that since A is normal, the spectral norm coincides with the spectral radius of $\mathcal{C}_2\mathcal{C}_1$. We employ an optimization routine to find the values of $\alpha_1,\ \alpha_2$ where f achieves its minimal value $f_{\min}$ in both situations. This leads, one the one hand, to $f_{\min}(\alpha_1,\ \alpha_2) = 0.382$ with $\alpha_1 = \alpha_2 = -1.118$ for two consecutive real shifts, and, on the other hand, to $f_{\min}(\alpha_1,\ \overline{\alpha_1}) = 0.1998$ with $\alpha_1 = -0.7362 + \jmath 0.8158$. The surface plot of f in Figure 4.1 shows that f w.r.t. a pair of complex conjugated shifts achieves a considerably deeper sink compared to two real shifts.

However, if the imaginary parts of the eigenvalues are small compared to real parts, i.e., the spectrum is located within a small strip in $\mathbb{C}_-$, then using only real shift parameters might be adequate, see, e.g., [232, 233]. Complex shift parameters are therefore particularly interesting for problems having eigenvalues with large imaginary parts, i.e., the spectrum has a large opening angle $\psi(A,\ E)$ (cf. (2.27)).

4.1.3. Previous Approaches

Here we briefly describe two older approaches that deal with complex shift parameters in the G-LR-ADI iteration before we present our new one.

Completely Real Formulation of the LR-ADI Iteration

In [42, 161, 183], a real formulation of the original LR-ADI iteration (3.2) for (3.2) is presented, which exploits the identity

$$(A \pm \alpha I_n)(A \pm \overline{\alpha} I_n) = A^2 + 2\operatorname{Re}(\alpha)\, A + |\alpha|^2 I_n, \quad \forall A \in \mathbb{R}^{n\times n},\ \alpha \in \mathbb{C}.$$

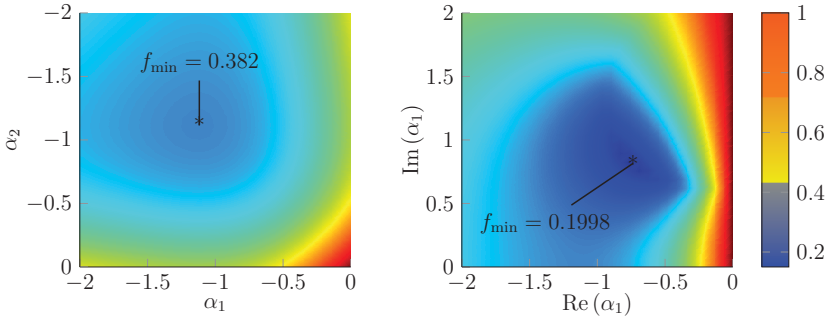


Figure 4.1.: Surface plot of f in dependence of $\alpha_1,\ \alpha_2 \in \mathbb{R}_-$ (left) and Re (α_1), Im (α_1) for $\alpha_1 \in \mathbb{C}_-$ (right).

Concatenating two LR-ADI iteration steps associated with a pair of complex conjugate shift parameters into one step leads to a completely real LR-ADI iteration (LR-ADI-R) [42, Algorithm 4.]. To illustrate this method let, for instance, $\mu_j = \{\alpha_j,\ \alpha_{j+1} := \overline{\alpha_j}\}$ be such a pair, α_{j-1} be real and $j > 2$. Then the LR-ADI-R iteration updates the low-rank solution factor as

$$Z_{j+1} = [Z_{j-1},\ 2\sqrt{-\mathrm{Re}\,(\alpha_j)}|\alpha_j|\tilde{V}_j,\ 2\sqrt{-\mathrm{Re}\,(\alpha_j)}\tilde{V}_{j+1}], \tag{4.1}$$

where $\tilde{V}_j,\ \tilde{V}_{j+1} \in \mathbb{R}^{n\times r}$ are real iterates constructed via

$$\tilde{V}_j = \left(A^2 + 2\,\mathrm{Re}\,(\alpha_j)\,A + |\alpha_j|^2 I_n\right)^{-1}\left((A - \alpha_{j-1}I_n)\tilde{V}_{j-1}\right), \quad \tilde{V}_{j+1} = A\tilde{V}_j \tag{4.2}$$

and $\tilde{V}_{j-1}$ is also real due to an appropriate treatment of the complex shifts in the preceding iterations. The other situations, i.e., $j \leq 2$ as well as $\alpha_{j-1} \in \mathbb{C}_-$ are dealt with similarly. A complete algorithm following this strategy can be found in [42, Algorithm 4.],[161], but unfortunately no derivations of the involved formulas are given. A complete derivation of this implementation can be found in the appendix of the preprint[1] of [38]. This reformulation has the advantage that no complex arithmetic operations are required but the disadvantage that linear systems with coefficient matrices of the form $A^2 + 2\,\mathrm{Re}\,(\alpha_j)\,A + |\alpha_j|^2 I_n$ are encountered. For large-scale matrices, A^2 might not be computable in an efficient way and even if, it will not preserve the original sparsity of A such that the application of sparse direct solvers is derailed. Iterative solvers can still be applied since they work with matrix-vector products only, such that the explicit construction of the operator is not needed. However, the condition number can be increased due to the squaring which might deteriorate the efficiency of iterative solvers as well.

[1] Available at `https://www2.mpi-magdeburg.mpg.de/preprints/2011/08/`.

The situation worsens for GCALEs (3.12) for which the above equations are

$$\tilde{V}_j = \left(AE^{-1}A + 2\,\mathrm{Re}\,(\alpha_j)\,A + |\alpha_j|^2 E\right)^{-1}\left((A - \alpha_{j-1}E)\tilde{V}_{j-1}\right), \quad \tilde{V}_{j+1} = E^{-1}A\tilde{V}_j. \tag{4.3}$$

Solving the linear system in (4.3) requires to build – or multiply a vector to – the matrix $AE^{-1}A$ which can, depending on the structure of E, easily become dense. Hence, except when E is an easy matrix (e.g., diagonal), this real formulation of G-LR-ADI is inefficient in a large-scale setting. Moreover, for each complex pair, an additional solve with E is required which introduces even higher computation costs. Nevertheless, an algorithm following this approach for GCALEs is referred to as G-LR-ADI-R iteration and given in [35, Algorithm 1], where also similar modifications for the SO-LR-ADI and SLRCF-ADI iteration can be found.

So far, for both CALEs and GCALEs, this strategy has been only worked out for the original (G-)LR-ADI iteration (3.2), (3.12) and not for the new formulation (Algorithm 3.2) which we developed in the previous chapter. A straightforward application of the techniques mentioned above to Algorithm 3.1 yields

$$\tilde{V}_j = \left(AE^{-1}A + 2\,\mathrm{Re}\,(\alpha_j)\,A + |\alpha_j|^2 E\right)^{-1} W_{j-1}, \quad \tilde{V}_{j+1} = E^{-1}A\tilde{V}_j, \tag{4.4}$$

$$W_{j+1} = W_{j-1} - 4\,\mathrm{Re}\,(\alpha_j)\,E\tilde{V}_{j+1} = W_{j-1} - 4\,\mathrm{Re}\,(\alpha_j)\,A\tilde{V}_j \tag{4.5}$$

In contrast to (4.2),(4.3), these relations hold for each pair of complex conjugated shifts, independent of j and the type of shifts encountered before, which simplifies the resulting completely real G-LR-ADI iteration.

We show now how (4.4) can be further rewritten to get rid of the difficult coefficient matrices. The occurrence of $AE^{-1}A$ and the additional solve with E for $\tilde{V}_{j+1}$ can be circumvented by solving the augmented linear system

$$\begin{bmatrix} \tilde{V}_j \\ \tilde{V}_{j+1} \end{bmatrix} = \begin{bmatrix} 2\,\mathrm{Re}\,(\alpha_j)\,A + |\alpha_j|^2 E & A \\ A & -E \end{bmatrix}^{-1} \begin{bmatrix} W_{j-1} \\ 0 \end{bmatrix} \tag{4.6}$$

which is equivalent to the construction of $\tilde{V}_j$, $\tilde{V}_{j+1}$ in (4.4). Instead of (4.4) one has to solve a $2n$ dimensional, real, linear system. The numerical properties of the $2n$ dimensional matrix in (4.6) might, however, be better compared to the one in (4.4), especially since it is still sparse if A, E are. This strategy can, of course, also be applied in the case $E = I_n$ to avoid working with A^2 in (4.2). Algorithm 4.1 illustrates the completely real G-LR-ADI in the new formulation and using the augmented linear systems (4.6). We will use the reformulated versions (4.4), (4.6) of the G-LR-ADI-R iteration for comparison to our novel, and often more efficient approach later on.

LyaPack Approach

Another approach can be found in the latest implementations of the LR-ADI iteration in LyaPack [185]. Unfortunately, a derivation of this approach cannot be found anywhere in the literature. The crucial point there is that again for V_j, V_{j+1} corresponding to

Algorithm 4.1: G-LR-ADI-R

Input : Matrices $A,\ E,\ F$ defining (3.12), proper set of shift parameters $\{\alpha_1, \ldots, \alpha_{j_{\max}}\} \subset \mathbb{C}_-$, tolerance $0 < \tau \ll 1$.
Output: $Z = Z_{j_{\max}} \in \mathbb{R}^{n \times r j_{\max}}$, such that $ZZ^T \approx X$.

1 $W_0 = F, \quad Z_0 = [\,], \quad j = 1.$
2 **while** $\|W_{j-1}^T W_{j-1}\| \geq \tau \|F^T F\|$ **do**
3 **if** $\mathrm{Im}\,(\alpha_j) = 0$ **then**
4 Solve $(A + \alpha_j E)\tilde{V}_j = W_{j-1}$ for $\tilde{V}_j$.
5 $W_j = W_{j-1} - 2\,\mathrm{Re}\,(\alpha_j)\, E\tilde{V}_j, \quad Z_j = [Z_{j-1},\ \sqrt{-2\alpha_j}\tilde{V}_j]$.
6 $j = j + 1$.
7 **else**
8 $\gamma_j = 2\sqrt{\mathrm{Re}\,(\alpha_j)}$.
9 Solve $\begin{bmatrix} 2\,\mathrm{Re}\,(\alpha_j)A + |\alpha_j|^2 E & A \\ A & -E \end{bmatrix} \begin{bmatrix} \tilde{V}_j \\ \tilde{V}_{j+1} \end{bmatrix} = \begin{bmatrix} W_{j-1} \\ 0 \end{bmatrix}$ for $\tilde{V}_j,\ \tilde{V}_{j+1}$.
10 $W_{j+1} = W_{j-1} + \gamma_j^2 E\tilde{V}_{j+1}$.
11 $Z_{j+1} = \left[Z_{j-1},\ \gamma_j|\alpha_j|\tilde{V}_j,\ \gamma_j\tilde{V}_{j+1}\right]$.
12 $j = j + 2$.

$\mu_j = \{\alpha_j, \alpha_{j+1} := \overline{\alpha_j}\}$, provided that $\mathrm{Re}\,(V_j)\,,\ \mathrm{Im}\,(V_j)$ have both full column rank r, it holds

$$\mathrm{rank}\,(N_j) = 2r,\ N_j := [\mathrm{Re}\,(V_j)\,,\ \mathrm{Im}\,(V_j)\,,\ \mathrm{Re}\,(V_{j+1})\,,\ \mathrm{Im}\,(V_{j+1})] \in \mathbb{R}^{n \times 4r} \tag{4.7}$$

which will be derived in the next subsection. In LyaPack, the current solution factor Z is expanded by real block columns using Algorithm 4.2 which processes column wise the imaginary and real parts of $V_j,\ V_{j+1}$. Hence, the matrix $\tilde{N}_j$ used there is of dimension $n \times 4$ and has rank 2. To neglect the linearly dependent part, a thin singular value decomposition (SVD)

$$\begin{aligned} \tilde{N}_j = U\Sigma Q^T &= [u_1,\ u_2,\ u_3,\ u_4] \begin{bmatrix} \sigma_1 & & & \\ & \sigma_2 & & \\ & & 0 & \\ & & & 0 \end{bmatrix} \begin{bmatrix} q_1 & q_2 & q_3 & q_4 \end{bmatrix}^T \\ &= [u_1,\ u_2] \begin{bmatrix} \sigma_1 & \\ & \sigma_2 \end{bmatrix} \begin{bmatrix} q_1^T \\ q_2^T \end{bmatrix} \end{aligned}$$

in Line 2 of Algorithm 4.2 reveals the blocks corresponding to range$(\tilde{N}_j)$. Adding $\tilde{N}_j$ to the current low-rank factor will introduce the new blocks[2]

$$\tilde{N}_j\tilde{N}_j^T = [u_1,\ u_2] \begin{bmatrix} \sigma_1^2 & \\ & \sigma_2^2 \end{bmatrix} [u_1,\ u_2]^T = T_j T_j^T$$

[2]We point out that in the original implementation a superfluous SVD $HSP^T = [h_1,\ h_2]^T[p_1,\ p_2]$ is used to get the new blocks via $[\tilde{Z}_j(:,\ell),\ \tilde{Z}_{j+1}(:,\ell)] = T = [\sigma_1 u_1,\ \sigma_2 u_2]HS$.

Algorithm 4.2: Augmentation of Z by real block columns

Input : Low-rank factor $Z_{j-1} \in \mathbb{R}^{n\times(j-1)r}$, iterates $V_j,\ V_{j+1} \in \mathbb{C}^{n\times r}$ w.r.t. $\mu_j = \{\alpha_j,\ \overline{\alpha_j}\}$

Output: $Z_{j+1} \in \mathbb{R}^{n\times(j+1)r}$, that is, Z_{j-1} augmented by $2m$ new real columns.

1 **for** $\ell = 1, \ldots, r$ **do**

2 Compute (thin) singular value decomposition (SVD)

$$U\Sigma Q^T = [\mathrm{Re}\,(V_j(:,\ell)), \mathrm{Im}\,(V_j(:,\ell)), \mathrm{Re}\,(V_{j+1}(:,\ell)), \mathrm{Im}\,(V_{j+1}(:,\ell))] \in \mathbb{R}^{n\times 4}$$

3 Partition $U,\ \Sigma$ w.r.t. nonzero singular values

$$U = \begin{bmatrix} u_1,\ u_2,\ u_3,\ u_4 \end{bmatrix},\ \Sigma = \mathrm{diag}\,(\sigma_1,\ \sigma_2,\ 0,\ 0)$$

4 Determine new real columns $\tilde{Z}_j(:,\ell) = \sigma_1 u_1$, $\tilde{Z}_{j+1}(:,\ell) = \sigma_2 u_2$.

5 $Z_{j+1} = [Z_{j-1},\ \tilde{Z}_j,\ \tilde{Z}_{j+1}] \in \mathbb{R}^{n\times(j+1)r}$.

with $T_j := [\sigma_1 u_1,\ \sigma_2 u_2]$ in the approximate solution of (3.2). This is carried out in Line 4 and ensures that only the columns of U which correspond to the linearly independent part of $\tilde{N}_j$ are added to Z_{j-1}. Obviously, this approach generates a real low-rank factor Z after termination of Algorithm 3.2, but uses complex arithmetic and requires the computation of r singular value decompositions of $n \times 4$ matrices for each complex pair ν_j encountered. Since r is typically much smaller than n this usually yields minor additional costs. This approach does not alter the linear systems to be solved compared to the previous approach, such that the performance of sparse direct or iterative solvers is not negatively affected. In the next part we propose a new approach which explicitly determines the linear dependence of the real and imaginary parts of V_j, V_{j+1} and makes the linear system with $A + \alpha_{j+1}E = A + \overline{\alpha_j}E$ redundant.

4.1.4. A New Approach Based on the Interconnection of ADI Iterates

Our new approach exploits (4.7) by using the correct linear combination of the linearly dependent part of N_j without using a SVD. The following theorem states that V_{j+1} is explicitly known once V_j has been computed.

Theorem 4.2 (Revised version of [38, Theorem 1]):

Assume a proper set of shift parameters and real matrices $A,\ E,\ F$ defining the GCALE (3.12). Then for two subsequent blocks $V_j,\ V_{j+1}$ of the G-LR-ADI iteration (Algorithm 3.2) associated with a pair of complex conjugated shifts $\nu_j = \{\alpha_j,\ \alpha_{j+1} := \overline{\alpha_j}\}$

it holds

$$V_{j+1} = \overline{V_j} + 2\delta_j \operatorname{Im}(V_j), \tag{4.8a}$$
$$W_{j+1} = W_{j-1} - 4\operatorname{Re}(\alpha_j) E (\operatorname{Re}(V_j) + \delta_j \operatorname{Im}(V_j)) \in \mathbb{R}^{n\times r} \tag{4.8b}$$

with $\delta_j := \frac{\operatorname{Re}(\alpha_j)}{\operatorname{Im}(\alpha_j)}$. Furthermore, regardless of the type of shifts encountered before, if α_j is real, then V_j and W_j are real $n \times r$ matrices. ◊

Proof. Assume at first that we are at iteration j of the G-LR-ADI iteration and W_{j-1} is a real matrix. This is obviously the case when $j = 1$ since $W_0 = F$ and when all shifts $\alpha_1, \ldots, \alpha_{j-1}$ are real. According to the G-LR-ADI iteration (3.12) we have

$$(A + \alpha_j E)V_j = W_{j-1}$$

and, by splitting α_j, V_j and W_j into their real and imaginary parts,

$$\begin{aligned}(A + \operatorname{Re}(\alpha_j) E) \operatorname{Re}(V_j) - \operatorname{Im}(\alpha_j) E \operatorname{Im}(V_j) &= W_j,\\ (A + \operatorname{Re}(\alpha_j) E) \operatorname{Im}(V_j) + \operatorname{Im}(\alpha_j) E \operatorname{Re}(V_j) &= 0\end{aligned}$$

since $\operatorname{Im}(W_{j-1}) = 0$. Hence, $E \operatorname{Re}(V_j) = -\frac{1}{\operatorname{Im}(\alpha_j)}(A + \operatorname{Re}(\alpha_j) E) \operatorname{Im}(V_j)$ which reveals

$$\begin{aligned}W_j &= W_{j-1} - 2\operatorname{Re}(\alpha_j) E (\operatorname{Re}(V_j) + \jmath \operatorname{Im}(V_j))\\ &= W_{j-1} + 2\tfrac{\operatorname{Re}(\alpha_j)}{\operatorname{Im}(\alpha_j)} ((A + \operatorname{Re}(\alpha_j) E) \operatorname{Im}(V_j) - \jmath \operatorname{Im}(\alpha_j) E \operatorname{Im}(V_j))\\ &= W_{j-1} + 2\delta_j (A + \overline{\alpha_j} E) \operatorname{Im}(V_j).\end{aligned}$$

For V_{j+1} associated to $\alpha_{j+1} = \overline{\alpha_j}$ this yields

$$\begin{aligned}V_{j+1} &= (A + \overline{\alpha_j} E)^{-1} W_j = (A + \overline{\alpha_j} E)^{-1} W_{j-1} + 2\delta_j (A + \overline{\alpha_j} E)^{-1} (A + \overline{\alpha_j} E) \operatorname{Im}(V_j)\\ &= \overline{V_j} + 2\delta_j \operatorname{Im}(V_j)\end{aligned}$$

and (4.8a) is established. The relation (4.8b) follows easily from

$$W_{j+1} = W_j - 2\operatorname{Re}(\alpha_j) E V_{j+1} = W_{j-1} - 2\operatorname{Re}(\alpha_j) E \left(V_j + \overline{V_j} + 2\delta_j \operatorname{Im}(V_j)\right)$$

which is a real $n \times r$ matrix. Hence, the assumption $W_j \in \mathbb{R}^{n\times r}$ is always fulfilled after a complex conjugated pair has been processed. If $\alpha_j \in \mathbb{R}_-$ it holds obviously $V_j \in \mathbb{R}^{n\times r}$ by (3.12) and $W_j \in \mathbb{R}^{n\times r}$ follows which completes the proof. □

Remark 4.3:

Theorem 4.2 adapts [38, Theorem 1] to the new formulation of the G-LR-ADI iteration proposed in Section 3.2.1 which incorporates the residual factors W_j. In [38] the original iteration (3.2) is used which yields exactly the same expression for V_{j+1}, but the proof is more complicated. The residual factors W_j are the right hand sides of the linear systems to be solved and, since it is proven that these are always real, minor additional computational savings can be achieved in the solutions process [36]. ◊

The above theorem reveals that, in the case of a pair of complex conjugate shifts, V_{j+1}, W_{j+1} corresponding to the shift $\alpha_{j+1} := \overline{\alpha_j}$ can be constructed entirely from the previous iterate V_j and shift α_j without solving a second complex linear system with $A + \overline{\alpha_j}E$. This reduces the numerical costs for generating both iterates roughly by one half and enables to process the whole complex conjugate pair at once. However, it is still required to solve one complex linear system.

Remark 4.4:

To get rid of the remaining complex arithmetic operation, it is possible to rewrite the complex linear system into an equivalent real representation [75], e.g.,

$$\begin{bmatrix} A + \mathrm{Re}\,(\alpha_j)\,E & -\mathrm{Im}\,(\alpha_j)\,E \\ \mathrm{Im}\,(\alpha_j)\,E & A + \mathrm{Re}\,(\alpha_j)\,E \end{bmatrix} \begin{bmatrix} \mathrm{Re}\,(V_j) \\ \mathrm{Im}\,(V_j) \end{bmatrix} = \begin{bmatrix} W_{j-1} \\ 0 \end{bmatrix} \tag{4.9}$$

and also formulate the rest of the G-LR-ADI iteration in terms of $\mathrm{Re}\,(V_j)$, $\mathrm{Im}\,(V_j)$. This leads to another completely real version of the G-LR-ADI iteration. For algorithms see [34, 35], where the original iteration (3.12) is used. The price one has to pay is that the augmented linear systems (4.9) are of dimension $2n$ and typically harder to solve than the original complex system which can usually be dealt with perfectly by modern computing environments. This is also confirmed in the numerical experiments later on. Moreover, there are different ways of an equivalent real representation of the complex linear systems. Sparse direct as well as iterative solvers might perform differently on different real formulations [75]. In the remainder we use the form (4.9).◊

Using (4.8), (4.9), it can be shown that $\mathrm{Re}\,(V_j)$, $\mathrm{Im}\,(V_j)$ and the iterates $\tilde{V}_j$, $\tilde{V}_{j+1}$ in (4.6) of the old completely real G-LR-ADI iteration (4.3) are related to each other via

$$\tilde{V}_j = \tfrac{-1}{\mathrm{Im}\,(\alpha_j)}\,\mathrm{Im}\,(V_j)\,, \quad \tilde{V}_{j+1} = \mathrm{Re}\,(V_j) + 2\beta_j\,\mathrm{Im}\,(V_j)\,,$$

see also [36]. Relation (4.8a) shows moreover that for two subsequent iterates V_j, V_{j+1} generated with a complex pair α_j, $\overline{\alpha_j}$ of shift parameters, statement (4.7) holds indeed, as it is also implicitly exploited in LyaPack using a singular value decomposition, see Algorithm 4.2. Hence, instead of adding V_j, V_{j+1} to the current low-rank solution factor, it is sufficient to add $\mathrm{Re}\,(V_j)$, $\mathrm{Im}\,(V_j)$ in the correct way as we will show next. This will also reduce the amount of memory required to store complex data.

The approximate solution incorporating the data generated in the iteration steps j, $j+1$ w.r.t. α_j, $\overline{\alpha_j}$ is given by

$$X_{j+1} = Z_{j-1}Z_{j-1}^T + \hat{Z}\hat{Z}^H$$

where $\hat{Z}$ denotes the $n \times 2r$ block to be added to Z_{j-1} which we assume to be real. By (4.8a)

$$\hat{Z} = \gamma_j \begin{bmatrix} V_j & V_{j+1} \end{bmatrix} = \begin{bmatrix} \mathrm{Re}\,(V_j) & \mathrm{Im}\,(V_j) \end{bmatrix} \hat{T}, \quad \hat{T} := \gamma_j \begin{bmatrix} I_r & I_r \\ \jmath I_r & (2\delta_j - \jmath) I_r \end{bmatrix}$$

with $\gamma_j := \sqrt{-2\,\mathrm{Re}\,(\alpha_j)}$.

Remark 4.5:

Note that it is claimed in [98] that the columns in the factor Z which were generated using α_j and $\overline{\alpha_j}$ are complex conjugate to each other and hence, Z can be transformed into a real form by applying

$$T = \tfrac{\sqrt{2}}{2}\begin{bmatrix} 1 & \jmath \\ 1 & -\jmath \end{bmatrix}$$

to these columns. As Theorem 4.2 shows, only the imaginary parts of these columns are complex conjugate versions of each other, rendering the above claim and the transformation incorrect. ◊

With $\tilde{Z} := \begin{bmatrix}\text{Re}\,(V_j) & \text{Im}\,(V_j)\end{bmatrix} \in \mathbb{R}^{n\times 2r}$ the contribution of $\hat{Z}$ to the low-rank solution of (3.2) is given by

$$\hat{Z}\hat{Z}^H = \tilde{Z}\hat{T}\hat{T}^H\tilde{Z}^T = \tilde{Z}\gamma_j^2\begin{bmatrix} 2I_r & 2\delta_j I_r \\ 2\delta_j I_r & (4\delta_j^2+1)I_r \end{bmatrix}\tilde{Z}^T.$$

The $2r \times 2r$ matrix $\hat{T}\hat{T}^H$ in the middle is always real symmetric and positive definite which can be shown using the factorization $\hat{T}\hat{T}^H = \gamma_j^2\hat{L}\hat{\Gamma}\hat{L}^T$ with

$$\hat{L} = \begin{bmatrix} 1 & 0 \\ \delta_j & 1 \end{bmatrix} \otimes I_r,\ \hat{\Gamma} = \begin{bmatrix} 2 & 0 \\ 0 & 2(\delta_j^2+1) \end{bmatrix} \otimes I_r.$$

Thus, it has a unique Cholesky factorization given by

$$\hat{T}\hat{T}^H = LL^T, \quad L := \hat{\gamma}_j \begin{bmatrix} 1 & 0 \\ \delta_j & \sqrt{\delta_j^2+1} \end{bmatrix} \otimes I_r \in \mathbb{R}^{2r\times 2r}$$

with $\hat{\gamma}_j := 2\sqrt{-\text{Re}\,(\alpha_j)} = \sqrt{2}\gamma_j$ which follows immediately from the above $\hat{L}\hat{\Gamma}\hat{L}^T$-factorization. Since $\hat{Z}\hat{Z}^H = \tilde{Z}LL^T\tilde{Z}^T$ we can add the $2r$ real columns

$$\tilde{Z}L = [\text{Re}\,(V_j),\ \text{Im}\,(V_j)]\,L = \hat{\gamma}_j\left[\text{Re}\,(V_j) + \delta\,\text{Im}\,(V_j),\ \sqrt{\left(\delta_j^2+1\right)}\cdot\text{Im}\,(V_j)\right] \tag{4.10}$$

to Z_{j-1} and, hence, a purely real low-rank solution factor is generated throughout the whole iteration.

The resulting G-LR-ADI iteration equipped with this strategy is shown in Algorithm 4.3, where storing complex data is only needed for the iterate V_j and the linear system $A + \alpha_j E$ (e.g., for the sparse LU factors) whenever a pair of complex conjugated shifts α_j, $\overline{\alpha_j}$ is encountered. Compared to the standard G-LR-ADI iteration (Algorithm 3.2), the memory required for storing the low-rank solution factor Z is consequently reduced by a factor of two. Notice that the variable V_{j+1} w.r.t. $\alpha_{j+1} = \overline{\alpha_j}$ is not formed since only W_{j-1} is required to continue the iteration. Furthermore, in order to stay as much in real arithmetic as possible, stopping criteria, e.g., the residual norm should be computed only after the complete complex pair is processed. Although we do not discuss strategies like column compression and Galerkin projection [201], they also naturally benefit from real, low-rank solution factors which reduce their computational effort, too.

Algorithm 4.3: G-LR-ADI-r iteration for computing real low-rank solution factors.

Input : Matrices $A,\ E,\ F$ defining (3.12), proper set of shift parameters $\{\alpha_1, \ldots, \alpha_{j_{\max}}\} \subset \mathbb{C}_-$, tolerance $0 < \tau \ll 1$.
Output: $Z_j \in \mathbb{R}^{n \times rj}$, such that $ZZ^T \approx X$.

$W_0 = F, \quad Z_0 = [\,], \quad j = 1.$
while $\|W_{j-1}^T W_{j-1}\| \geq \tau \|F^T F\|$ **do**
  Solve $(A + \alpha_j E)V_j = W_{j-1}$ for V_j.
  if $\operatorname{Im}(\alpha_j) = 0$ **then**
    $W_j = W_{j-1} - 2\operatorname{Re}(\alpha_j) EV_j, \quad Z_j = [Z_{j-1},\ \sqrt{-2\alpha_j} V_j].$
    $j = j + 1.$
  else
    $\hat{\gamma}_j = 2\sqrt{-\operatorname{Re}(\alpha_j)}, \quad \delta_j = \frac{\operatorname{Re}(\alpha_j)}{\operatorname{Im}(\alpha_j)}.$
    $W_{j+1} = W_{j-1} + \hat{\gamma}_j^2 E(\operatorname{Re}(V_j) + \delta_j \operatorname{Im}(V_j)).$
    $Z_{j+1} = \left[Z_{j-1},\ \ \hat{\gamma}_j(\operatorname{Re}(V_j) + \delta_j \operatorname{Im}(V_j)),\ \hat{\gamma}_j\sqrt{(\delta_j^2 + 1)} \cdot \operatorname{Im}(V_j)\right].$
    $j = j + 2.$

The Sylvester equation corresponding to the real low-rank solution factor

We can also generalize Corollary 3.9 in order to express the generated real low-rank solution factor as solution of a generalized Sylvester equation. For the sake of brevity we restrict these consideration to generalization of (3.24b) as we will use this relation in the upcoming chapters. The following corollary is a generalization of Corollary 3.9, [235, Lemma 3.1], [234, Lemma 5.12].

Corollary 4.6 (Generalization of Corollary 3.9):

The real low-rank solution factor Z_j after j steps of the G-LR-ADI-r iteration corresponding to the proper set of j shift parameters satisfies the GCASE

$$AZ_j - EZ_j B_{\text{ADI-r}} = W_j G_{\text{ADI-r}}^T. \tag{4.11}$$

There,

$$G_{\text{ADI-r}} := [\gamma_1^R, \ldots, \gamma_j^R]^T, \quad \gamma_i^R := \begin{cases} \gamma_i I_r, & \alpha_i \in \mathbb{R}_-, \\ [\gamma_i, 0] \otimes I_r, & \alpha_i, \alpha_{i+1} = \overline{\alpha_i} \in \mathbb{C}_-, \end{cases} \tag{4.12a}$$

$$B_{\text{ADI-r}} := \boldsymbol{\delta}^{-1} \boldsymbol{\alpha} \boldsymbol{\delta}, \quad \boldsymbol{\delta} = \operatorname{diag}(\boldsymbol{\delta}_1, \ldots, \boldsymbol{\delta}_j), \quad \boldsymbol{\alpha} = \begin{bmatrix} \boldsymbol{\alpha}_{11} & & \\ \vdots & \ddots & \\ \boldsymbol{\alpha}_{j1} & \cdots & \boldsymbol{\alpha}_{jj} \end{bmatrix}, \tag{4.12b}$$

$$\boldsymbol{\delta}_i := \begin{cases} \gamma_i I_r, & \alpha_i \in \mathbb{R}_-, \\ \sqrt{2}\gamma_i \begin{bmatrix} 1 & 0 \\ \delta_i & \sqrt{1+\delta_i^2} \end{bmatrix} \otimes I_r, & \alpha_i \in \mathbb{C}_-, \end{cases} \tag{4.12c}$$

$$\boldsymbol{\alpha}_{ii} := \begin{cases} \alpha_i I_r, & \alpha_i \in \mathbb{R}_-, \\ \begin{bmatrix} 3\,\mathrm{Re}\,(\alpha_i) & -\mathrm{Im}\,(\alpha_i) \\ \mathrm{Im}\,(\alpha_i)(1+4\delta_j^2) & -\mathrm{Re}\,(\alpha_i) \end{bmatrix} \otimes I_r, & \alpha_i \in \mathbb{C}_-, \end{cases} \tag{4.12d}$$

$$\boldsymbol{\alpha}_{ki} := \begin{cases} 2\,\mathrm{Re}\,(\alpha_k)\, I_r, & \alpha_k, \alpha_i \in \mathbb{R}_-, \\ [2\,\mathrm{Re}\,(\alpha_k), 0] \otimes I_r, & \alpha_k \in \mathbb{R}_-, \alpha_i \in \mathbb{C}_-, \\ 4 \begin{bmatrix} \mathrm{Re}\,(\alpha_k) \\ \delta_k\,\mathrm{Re}\,(\alpha_k) \end{bmatrix} \otimes I_r, & \alpha_k \in \mathbb{C}_-, \alpha_i \in \mathbb{R}_-, \\ 4 \begin{bmatrix} \mathrm{Re}\,(\alpha_k) & 0 \\ \delta_k\,\mathrm{Re}\,(\alpha_k) & 0 \end{bmatrix} \otimes I_r, & \alpha_k \in \mathbb{C}_-, \alpha_i \in \mathbb{C}_- \end{cases} \tag{4.12e}$$

for $1 \le k < i \le j$. ◊

Proof. At first note that by (4.8a) for V_{i+1}, V_i corresponding to a pair of complex conjugate shifts $\alpha_i,\ \alpha_{i+1} = \overline{\alpha_i}$ it holds

$$AV_{i+1} = W_{i+1} + \alpha_j E V_{i+1} = W_{i+1} + \alpha_i E\left(\overline{V_i} + 2\delta_i\,\mathrm{Im}\,(V_i)\right).$$

Inserting (4.8a) also in the left hand side of the above equation, collecting the real and imaginary parts of V_i into $\tilde{Z}_i = [\mathrm{Re}\,(V_i),\ \mathrm{Im}\,(V_i)]$ yields, after performing some basic manipulations,

$$A\tilde{Z}_i = [W_{i+1}, 0] + E\tilde{Z}_i\boldsymbol{\alpha}_{ii} \quad \text{with} \quad \boldsymbol{\alpha}_{ii} := \begin{bmatrix} 3\,\mathrm{Re}\,(\alpha_i) & -\mathrm{Im}\,(\alpha_i) \\ \mathrm{Im}\,(\alpha_i)(1+4\delta_i^2) & -\mathrm{Re}\,(\alpha_i) \end{bmatrix}.$$

If we do this for all the involved shifts and iteration data we end up with

$$A[\tilde{Z}_1, \ldots, \tilde{Z}_j] = [\hat{W}_1, \ldots, \hat{W}_j] + E[\tilde{Z}_1, \ldots, \tilde{Z}_j]\,\mathrm{diag}\,(\boldsymbol{\alpha}_{11}, \ldots, \boldsymbol{\alpha}_{jj}),$$

$$\tilde{Z}_i = \begin{cases} V_i, & \alpha_i \in \mathbb{R}_-, \\ [\mathrm{Re}\,(V_i),\ \mathrm{Im}\,(V_i)], & \alpha_i \in \mathbb{C}_-, \end{cases} \quad \hat{W}_i = \begin{cases} W_i, & \alpha_i \in \mathbb{R}_-, \\ [W_i,\ 0], & \alpha_i \in \mathbb{C}_-, \end{cases}$$

and the diagonal blocks $\boldsymbol{\alpha}_{ii}$ from (4.12d). Using (3.18b) for real and (4.8b) for pairs of complex conjugated shifts yields, similar to the proof of Corollary 3.9 and using the notation introduced in (4.12e),

$$W_{j-i} = \hat{W}_j + 2E\sum_{k=0}^{i-1} \tilde{Z}_{j-k}\boldsymbol{\alpha}_{ki}, \quad i = 1, \ldots, j-1,$$

such that

$$A[\tilde{Z}_1, \ldots, \tilde{Z}_j] = [\hat{W}_j, \ldots, \hat{W}_j] + E[\tilde{Z}_1, \ldots, \tilde{Z}_j]\boldsymbol{\alpha}.$$

By (4.10), we have for the generated real low-rank solution factor

$$Z_j := [\tilde{Z}_1, \ldots, \tilde{Z}_j]\boldsymbol{\delta}$$

such that

$$Z_j = [\hat{W}_j, \ldots, \hat{W}_j]\boldsymbol{\delta} + EZ_j\boldsymbol{\delta}^{-1}\boldsymbol{\alpha}\boldsymbol{\delta}. \tag{4.13}$$

from which the result (4.11) can be easily established after rewriting $[\hat{W}_j, \ldots, \hat{W}_j]\boldsymbol{\delta}$. □

Similar to the short discussion after Corollary 3.9, by the connections of the G-LR-ADI iteration to rational Krylov subspace methods [82, 96, 235, 237, 234], the relation (4.11) can be seen as generalized version of results from [195]. The relation (4.11) will be useful in the next chapter for computing AZ_j without matrix vectors products involving A.

4.1.5. Numerical Examples

In this section we test the different ways to generate real, low-rank solution factors with the G-LR-ADI iteration. We employ the following variants:

M1 G-LR-ADI (Algorithm 3.2) without any handling of complex shifts,

M2 G-LR-ADI-R (cf. Section 4.1.3, [42, Algorithm 4.], [36]) in the new formulation (4.4),

M2* M2 using the augmented linear system (4.6) (Algorithm 4.1),

M3 G-LR-ADI with LyaPack realification (Algorithm 4.2),

M4 novel approach given in Algorithm 4.3 in Section 4.1.4,

M4* completely real version of M4 using real linear systems (4.9) (cf. Remark 4.4).

We test the different approaches for solving standard (3.2) and generalized (3.12) Lyapunov equations. The examples *ocean*, *ifiss16k*, *chain* and *bips* are used (cf. Section 2.4). The latter two examples inherit the structures (2.33b) and (2.34) such that the SO-LR-ADI and SLRCF-ADI iteration is employed (cf. Section 3.2.5), respectively, which includes appropriate adaptations of the variants M2 – M4*. In all variants, the scaled residual norm $\varepsilon_j := \|\mathcal{L}_j\|_2/\|F^TF\|_2$ is used as stopping criterion with a tolerance τ. As before we employ a fixed number J of heuristic shift parameters [183] which are obtained from k_+ and k_- Ritz and inverse Ritz values, respectively. In the remainder $J_{\mathbb{R}}$ denotes the number of real shifts, and, respectively, $J_{\mathbb{C}}$ the number of pairs of complex conjugate shifts.

For the *ocean* example, Figure 4.2 illustrates the scaled residual norm against the iteration number. Apparently, all six tested variants show almost indistinguishable residual norms during the iteration and converge after 86 iterations to the desired accuracy of $\tau = 10^{-10}$. This illustrates the mathematical equivalence of the different approaches. The small deviations in the residuals occur in variant M1 in the iteration steps which work with a pair α_j, $\overline{\alpha_j}$ of complex shifts, see the enlarged area in Figure 4.2 around step $j = 15$. There, the intermediate residual after the first complex shift parameter α_j is computed in M1, but this does not happen in the other variants M2–M4*. After the conjugate shift $\overline{\alpha_j}$ has been processed, too, the residual norms coincide again.

Now we compare the different variants in terms of the computation times required until termination. The results as well as the used setup parameters for the G-LR-ADI iteration and its variants are summarized in Table 4.1. Apparently, for all examples the novel approach M4 requires the smallest computation times, closely followed by M4*

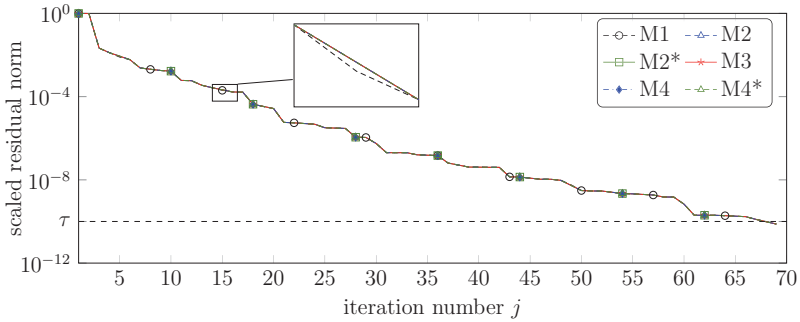


Figure 4.2.: History of the normalized residual norms obtained with the LR-ADI iteration and its variants generating real low-rank solutions factors for the *ocean* example. The dashed thin black line indicates the chosen tolerance τ.

Table 4.1.: Parameters, required ADI iteration steps j_{it}, final normalized residual norm $\varepsilon_{j_{\text{it}}}$, and computation times in seconds for the examples *ocean* and *bips*.

	Parameters				Computation time in seconds					
Example	k_+, k_-, J, ($J_{\mathbb{R}}$, $J_{\mathbb{C}}$)	τ	j_{it}	$\varepsilon_{j_{\text{it}}}$	M1	M2	M2*	M3	M4	M4*
ocean	40, 40, 30, (18, 6)	10^{-10}	69	$7.52{\cdot}10^{-11}$	122.2	103.6	84.1	122.7	**68.2**	81.7
ifiss66k	30, 20, 20, (10, 5)	10^{-10}	165	$8.62{\cdot}10^{-11}$	233.5	–	223.7	234.3	**135.6**	192.9
chain	80, 80, 50, (20, 15)	10^{-8}	476	$9.98{\cdot}10^{-9}$	46.6	>1 h	31.1	40.1	**19.4**	31.5
bips	80, 80, 61, (11, 25)	10^{-8}	194	$9.36{\cdot}10^{-9}$	25.5	16.8	18.3	24.3	**13.3**	17.6

and then M2* is most cases. Only for the examples *FDM* and *bips* the approach M2, which uses the squared matrix A^2, is competitive. It needs considerably more time for the other examples and could not be carried out for example *ifiss66k* in a reasonable amount of time and is therefore neglected there. Over one hour of computation time is consumed by M2 for example chain. Apart from these two examples, M1 and M3 require the longest computation time since two complex linear systems have to be solved for each complex conjugate pair of shift parameters.

To conclude, an intelligent treatment of complex shift parameters can greatly reduce the computational complexity of the G-LR-ADI iteration. The approaches M2*, M4 and M4* outperform the other, existing approaches in all examples. Here, our novel variant M4 performed best w.r.t. the required computation times. For other examples and computing environments which of the approaches M2*, M4, M4* is superior appears to be problem dependent and essentially reduces to the question whether the original complex, or the augmented $2n \times 2n$ linear systems (4.6), (4.9) can be solved more efficiently. Only in a few cases, e.g., if $E = I_n$ and forming A^2 does not introduce too much fill-in, the traditional completely real approach M2 [42] can also be competitive. However, for the

general case with $E \neq I_n$ it is very expensive and cannot be recommended.

4.2. Computing Real Low-rank Solutions of Sylvester Equations by the Factored ADI Iteration

In this section, we consider approaches for generating real low-rank solutions within the fADI iteration. This will lead to a generalization of Theorem 4.2. As for the GCALE case we require that the matrices defining (3.27), (3.34) are real. Moreover, another main and natural ingredient for generating real solution factors is that both sets of shift parameters are proper in the sense of Definition 4.1, i.e., they are of the form

$$\{\alpha_1, \ldots, \alpha_J\} = \{\mu_1, \ldots, \mu_{\tilde{J}}\} \subset \mathbb{C},$$
$$\{\beta_1, \ldots, \beta_L\} = \{\nu_1, \ldots, \nu_{\tilde{L}}\} \subset \mathbb{C},$$

where $\mu_j,\ \nu_i,\ (j = 1, \ldots, \tilde{J},\ i = 1, \ldots, \tilde{L})$ are either real numbers or pairs of complex conjugate numbers $\mu_j := \{\alpha_j, \overline{\alpha_j}\}, \nu_i := \{\beta_i, \overline{\beta_i}\}$. Moreover, we restrict the pairs $(\mu_j,\ \nu_j)$ to the following cases:

1) Both μ_j and ν_j are real numbers α_j and β_j.

2) Both $\mu_j = \{\alpha_j,\ \alpha_{j+1} = \overline{\alpha_j}\}$ and $\nu_j = \{\beta_j,\ \beta_{j+1} = \overline{\beta_j}\}$ are pairs of complex conjugate numbers.

3) One complex pair and two real shifts:

   a) $\alpha_j,\ \alpha_{j+1}$ are real numbers and ν_j is a complex pair,

   b) μ_j is a complex pair and $\beta_j,\ \beta_{j+1}$ are real numbers.

This is not a severe restriction since it can be achieved by a simple and usually slight reordering and rearrangement of the sets of shifts. Moreover, the iterates of the (G-)fADI iteration after a number of iterations do not change if the order of the processed shifts is changed. These restrictions, however, drastically simplify the encountered equations for generating real low-rank factors which will become clear later.

4.2.1. Interconnections Between Complex Iterates

Our approach for computing real solution factors is motivated by Theorem 4.2 regarding the G-LR-ADI iteration for GCALEs. It will turn out that, depending on the type of the current shift parameters, the current iterates can be constructed from the real and imaginary parts of previous iterates. We work through the different cases of possible subsequences of shift parameters and begin with case 2 since nothing has to be taken care of in case 1.

Theorem 4.7:
Let $W_{j-1} \in \mathbb{R}^{n\times r}$, $T_{j-1} \in \mathbb{R}^{m\times r}$, $\{\alpha_j,\ \alpha_{j+1} := \overline{\alpha_j}\}$, $\{\beta_j,\ \beta_{j+1} := \overline{\beta_j}\}$, and define $\gamma_j := \beta_j - \alpha_j$. Then it holds at iteration step $j+1$ of Algorithm 3.4:

$$V_{j+1} = \overline{V_j} + \tfrac{\gamma_j}{\operatorname{Im}(\beta_j)} \operatorname{Im}(V_j), \tag{4.14a}$$

$$\begin{aligned} W_{j+1} = W_{j-1} &+ 2\operatorname{Re}(\gamma_j) E \operatorname{Re}(V_j) \\ &+ 2\operatorname{Re}(\gamma_j) E \left(\tfrac{|\gamma_j|^2}{\operatorname{Im}(\beta_j)} - 2\operatorname{Im}(\gamma_j) \right) \operatorname{Im}(V_j) \in \mathbb{R}^{n\times r}, \end{aligned} \tag{4.14b}$$

$$S_{j+1} = \overline{S_j} + \tfrac{\overline{\gamma_j}}{\operatorname{Im}(\alpha_j)} \operatorname{Im}(S_j), \tag{4.14c}$$

$$\begin{aligned} T_{j+1} = T_{j-1} &- 2\operatorname{Re}(\gamma_j) C^T \operatorname{Re}(S_j) \\ &- \left(\tfrac{|\gamma_j|^2}{\operatorname{Im}(\alpha_j)} + 2\operatorname{Im}(\gamma_j) \right) C^T \operatorname{Im}(S_j) \in \mathbb{R}^{m\times r}. \end{aligned} \tag{4.14d}$$

Proof. At Line 3 of Algorithm 3.4, V_j is obtained from $(A - \beta_j E)V_j = W_{j-1} \in \mathbb{R}^{n\times r}$. Splitting β_j and V_j into their real and imaginary parts gives

$$\begin{aligned} (A - \operatorname{Re}(\beta_j) E) \operatorname{Re}(V_j) &= W_{j-1}, \\ (A - \operatorname{Re}(\beta_j) E) \operatorname{Im}(V_j) &= \operatorname{Im}(\beta_j) E \operatorname{Re}(V_j). \end{aligned} \tag{4.15}$$

For V_{j+1} this yields, employing (4.15),

$$\begin{aligned} V_{j+1} &= (A - \overline{\beta_j}E)^{-1} W_{j-1} = (A - \overline{\beta_j}E)^{-1}(W_{j-1} + \gamma_j E V_j) \\ &= \overline{V_j} + \gamma_j (A - \overline{\beta_j}E)^{-1} \left(\tfrac{1}{\operatorname{Im}(\beta_j)} (A - \operatorname{Re}(\beta_j) E) \operatorname{Im}(V_j) + \imath E \operatorname{Im}(V_j) \right) \\ &= \overline{V_j} + \tfrac{\gamma_j}{\operatorname{Im}(\beta_j)} (A - \overline{\beta_j}E)^{-1} (A - \overline{\beta_j}E) \operatorname{Im}(V_j) \end{aligned}$$

and (4.14a) is established. The relation (4.14b) follows from

$$\begin{aligned} W_{j+1} &= W_j + \gamma_{j+1} E V_{j+1} = W_{j-1} + \gamma_j E V_j + \overline{\gamma_j} E \left(\overline{V_j} + \tfrac{\gamma_j}{\operatorname{Im}(\beta_j)} E \operatorname{Im}(V_j) \right) \\ &= W_{j-1} + 2\operatorname{Re}(\gamma_j) E \operatorname{Re}(V_j) + \left(\tfrac{|\gamma_j|^2}{\operatorname{Im}(\beta_j)} - 2\operatorname{Im}(\gamma_j) \right) E \operatorname{Im}(V_j). \end{aligned}$$

For S_j we have $(B - \operatorname{Re}(\alpha_j) C)^T \operatorname{Im}(S_j) = -\operatorname{Im}(\alpha_j) C^T \operatorname{Re}(S_j)$ and thus,

$$\begin{aligned} S_{j+1} &= (B - \overline{\alpha_j}C)^{-H} \left(T_{j-1} - \overline{\gamma_j} C^T S_j \right) \\ &= \overline{S_j} + \tfrac{\overline{\gamma_j}}{\operatorname{Im}(\alpha_j)} (B - \overline{\alpha_j}C)^{-H} \big((B - \operatorname{Re}(\alpha_j) C)^T \operatorname{Im}(S_j) - \imath \operatorname{Im}(\alpha_j) C^T \operatorname{Im}(S_j) \big) \end{aligned}$$

giving (4.14b) and consequently (4.14d) via

$$\begin{aligned} T_{j+1} &= T_j - \overline{\gamma_{j+1}} C^T S_{j+1} \\ &= T_{j-1} - 2\operatorname{Re}(\gamma_j) C^T \operatorname{Re}(S_j) - \left(\tfrac{|\gamma_j|^2}{\operatorname{Im}(\alpha_j)} + 2\operatorname{Im}(\gamma_j) \right) C^T \operatorname{Im}(S_j). \end{aligned}$$

□

Note that only W_{j+1}, T_{j+1} are required to continue the iteration and since both are real, the result also holds if the algorithm is continued with two further pairs of complex conjugate shifts.

Exactly as in the result for the G-LR-ADI iteration for Lyapunov equations, the data corresponding to iteration step $j+1$ is constructed from quantities already available after iteration step j such that the linear systems w.r.t. the complex conjugate shifts $\overline{\alpha_j}$, $\overline{\beta_j}$ are obsolete which significantly reduces the overall computational costs.

The solution factors Z_{j-1}, Y_{j-1}, and Γ_{j-1} are augmented by

$$\begin{aligned}
[V_j,\ V_{j+1}] &= [\operatorname{Re}(V_j) + \jmath \operatorname{Im}(V_j),\ \operatorname{Re}(V_{j+1}) + \jmath \operatorname{Im}(V_{k+1})] \\
&= \underbrace{[\operatorname{Re}(V_j),\ \operatorname{Im}(V_j)]}_{=:\hat{Z}_j} \underbrace{\begin{bmatrix} 1 & \frac{1}{\jmath \frac{\operatorname{Re}(\beta_j)-\alpha_j}{\operatorname{Im}(\beta_j)}} \end{bmatrix} \otimes I_r}_{=:G_{Z_j}}, \\
[S_j,\ S_{j+1}] &= \underbrace{[\operatorname{Re}(S_j),\ \operatorname{Im}(S_j)]}_{=:\hat{Y}_j} \underbrace{\begin{bmatrix} 1 & \frac{1}{\jmath \frac{\overline{\beta_j}-\operatorname{Re}(\alpha_j)}{\operatorname{Im}(\alpha_j)}} \end{bmatrix} \otimes I_r}_{=:G_{Y_j}}, \\
\Gamma_{(j,j+1)} &:= \operatorname{diag}(\gamma_j,\ \overline{\gamma_j}) \otimes I_r.
\end{aligned}$$

The corresponding part of the solution X_{j+1} is

$$[V_j,\ V_{j+1}]\Gamma_{(j,j+1)}[S_j,\ S_{j+1}]^H = \hat{Z}_j \hat{\Gamma}_j^{(2)} \hat{Y}_j^T \in \mathbb{R}^{n\times m}$$

with

$$\begin{aligned}
\hat{\Gamma}_j^{(2)} &:= G_{Z_j} \Gamma_{(j,j+1)} G_{Y_j}^H \\
&= \begin{bmatrix} 2\operatorname{Re}(\gamma_j) & \frac{|\gamma_j|^2}{\operatorname{Im}(\alpha_j)} + 2\operatorname{Im}(\gamma_j) \\ \frac{|\gamma_j|^2}{\operatorname{Im}(\beta_j)} - 2\operatorname{Im}(\gamma_j) & \left(\frac{|\gamma_j|^2}{\operatorname{Im}(\beta_j)\operatorname{Im}(\alpha_j)} + 2\right)\operatorname{Re}(\gamma_j) \end{bmatrix} \otimes I_r \in \mathbb{R}^{2r\times 2r},
\end{aligned} \tag{4.16}$$

such that only real data is added to the existing low-rank factors. This finishes the treatment of case 2.

In case 3, i.e., if only one of the two pairs is complex and the other two involved shifts are real, the situation changes slightly but can be deduced directly from the previous theorem.

Corollary 4.8:

Let W_{j-1}, T_{j-1} be real. For case 3a, i.e., $\beta_j,\ \beta_{j+1} := \overline{\beta_j}$ and $\alpha_j,\ \alpha_{j+1} \in \mathbb{R}$ it holds

$$V_{j+1} = \operatorname{Re}(V_j) + \tfrac{\operatorname{Re}(\gamma_j)}{\operatorname{Im}(\beta_j)} \operatorname{Im}(V_j) \in \mathbb{R}^{n\times r}, \tag{4.17a}$$

$$\begin{aligned}
W_{j+1} &= W_{j-1} + (\operatorname{Re}(\gamma_j) + \operatorname{Re}(\gamma_{j+1}))E \operatorname{Re}(V_j) \\
&\quad + \tfrac{\operatorname{Re}(\gamma_j)\operatorname{Re}(\gamma_{j+1}) - \operatorname{Im}(\beta_j)^2}{\operatorname{Im}(\beta_j)} E \operatorname{Im}(V_j) \in \mathbb{R}^{n\times r},
\end{aligned} \tag{4.17b}$$

$$S_{j+1} = S_j - \gamma_{j+1}\tilde{S}_{j+1}, \quad \tilde{S}_{j+1} := (B - \alpha_{j+1}C)^{-T} C^T S_j \in \mathbb{R}^{n\times r}, \tag{4.17c}$$

$$T_{j+1} = T_{j-1} - (\operatorname{Re}(\gamma_j) + \operatorname{Re}(\gamma_{j+1}))C^T S_j + \delta_j C^T \tilde{S}_{j+1} \in \mathbb{R}^{m\times r}, \tag{4.17d}$$

where $\delta_j := \alpha_{j+1}^2 - 2\,\mathrm{Re}\,(\beta_j)\,\alpha_{j+1} + |\beta_j|^2$.
For case 3b, i.e. $\beta_j,\ \beta_{j+1} \in \mathbb{R}$ and $\alpha_j,\ \alpha_{j+1} = \overline{\alpha_j} \in \mathbb{C}$ we have

$$V_{j+1} = V_j + \gamma_{j+1}\tilde{V}_{j+1} \in \mathbb{C}^{n\times r}, \quad \tilde{V}_{j+1} := (A - \beta_{j+1}E)^{-1}EV_j \in \mathbb{R}^{n\times r}, \tag{4.18a}$$

$$W_{j+1} = W_{j-1} + (\mathrm{Re}\,(\gamma_j) + \mathrm{Re}\,(\gamma_{j+1}))EV_j + \delta_j E\tilde{V}_{j+1} \in \mathbb{R}^{n\times r}, \tag{4.18b}$$

$$S_{j+1} = \mathrm{Re}\,(S_j) + \tfrac{\mathrm{Re}\,(\gamma_j)}{\mathrm{Im}\,(\alpha_j)}\,\mathrm{Im}\,(S_j) \in \mathbb{R}^{m\times r}, \tag{4.18c}$$

$$\begin{aligned} T_{j+1} = {} & T_{j-1} - (\mathrm{Re}\,(\gamma_j) + \mathrm{Re}\,(\gamma_{j+1}))C^T\,\mathrm{Re}\,(T_j) \\ & + \tfrac{-\mathrm{Re}\,(\gamma_j)\,\mathrm{Re}\,(\gamma_{j+1}) + \mathrm{Im}\,(\alpha_j)^2}{\mathrm{Im}\,(\alpha_j)} C^T\,\mathrm{Im}\,(T_j) \in \mathbb{R}^{m\times r}, \end{aligned} \tag{4.18d}$$

where $\delta_j := \beta_{j+1}^2 - 2\,\mathrm{Re}\,(\alpha_j)\,\beta_{j+1} + |\alpha_j|^2$. ◊

Proof. In case 3a the relation (4.17a) follows directly from relation (4.14a) by using $\mathrm{Im}\,(\alpha_j) = 0$. Plugging this into the constituting equations for W_{j+1} leads, after some simplifications, to (4.17b). The equation (4.17d) is nothing else but the original constructing formula for $S_{j+1} = S_j - \gamma_{j+1}\left((B - \alpha_{j+1}C)^{-T}(C^T S_j)\right)$ and exploiting that the solution of the occurring linear system is real due to $\alpha_{j,j+1} \in \mathbb{R}$, $S_j \in \mathbb{R}^{m\times r}$. Using this with some simple rearrangements leads then to (4.17d). The case 3b is proved using similar steps. □

Along the lines of case 2 the new parts in the low-rank factors in case 3a are given by the real blocks $\hat{Z}_j := [\mathrm{Re}\,(V_j),\ \mathrm{Im}\,(V_j)]$, $\hat{Y}_j := [S_j,\ \tilde{S}_{j+1}]$, and

$$\hat{\Gamma}_j^{(3a)} := \begin{bmatrix} \mathrm{Re}\,(\gamma_j) + \mathrm{Re}\,(\gamma_{j+1}) & -\delta_j \\ \frac{\mathrm{Re}\,(\gamma_j)\,\mathrm{Re}\,(\gamma_{j+1}) - \mathrm{Im}\,(\beta_j)^2}{\mathrm{Im}\,(\beta_j)} & \frac{-\mathrm{Re}\,(\gamma_j)\delta_j}{\mathrm{Im}\,(\beta_j)} \end{bmatrix} \otimes I_r \in \mathbb{R}^{2r\times 2r}. \tag{4.19}$$

Similarly, the low-rank factors in case 3b are augmented by $\hat{Z}_j := [V_j,\ \tilde{V}_{j+1}]$, $\hat{Y}_j := [\mathrm{Re}\,(S_j),\ \mathrm{Im}\,(S_j)]$, and

$$\hat{\Gamma}_j^{(3b)} := \begin{bmatrix} \mathrm{Re}\,(\gamma_j) + \mathrm{Re}\,(\gamma_{j+1}) & \frac{\mathrm{Re}\,(\gamma_j)\,\mathrm{Re}\,(\gamma_{j+1}) - \mathrm{Im}\,(\alpha_j)^2}{\mathrm{Im}\,(\alpha_j)} \\ \delta_j & \frac{\delta_j\,\mathrm{Re}\,(\gamma_j)}{\mathrm{Im}\,(\alpha_j)} \end{bmatrix} \otimes I_r \in \mathbb{R}^{2r\times 2r} \tag{4.20}$$

which finishes the discussion regarding case 3.

Remark 4.9:

Without the restrictions on the possible shift subsequences (cases 1 - 3) it is possible that there are longer recurrences of the $V_j,\ W_j$ iterates. Consider, for instance, the sequence $\beta_j,\ \beta_{j+1} = \overline{\beta_j},\ \beta_{j+2},\ \beta_{j+3} = \overline{\beta_{j+2}} \in \mathbb{C},\ \alpha_j \in \mathbb{R},\ \alpha_{j+1},\ \alpha_{j+2} = \overline{\alpha_{j+1}}$ and $\alpha_{j+3} \in \mathbb{R}$. Using the same techniques as above, it is possible to show, e.g., that

$$\begin{aligned} V_{j+3} = {} & \tfrac{\mathrm{Im}\,(\beta_{j+2}) - \mathrm{Im}\,(\alpha_{j+1})}{\mathrm{Im}\,(\beta_{j+2})}\,\mathrm{Re}\,(V_{j+2}) + \tfrac{\mathrm{Re}\,(\beta_{j+2}) - \mathrm{Re}\,(\alpha_{j+1})}{\mathrm{Im}\,(\beta_{j+2})}\,\mathrm{Im}\,(V_{j+2}) \\ & - \tfrac{\mathrm{Im}\,(\alpha_{j+1})}{\mathrm{Im}\,(\beta_{j+2})}\,\mathrm{Re}\,(V_j) - \tfrac{\mathrm{Im}\,(\alpha_{j+1})}{\mathrm{Im}\,(\beta_{j+2})}\tfrac{\mathrm{Re}\,(\beta_j) - \alpha_j}{\mathrm{Im}\,(\beta_j)}\,\mathrm{Im}\,(V_j) \in \mathbb{R}^{n\times r} \end{aligned}$$

and similar, equally complicated relations exist for $W_{j+3},\ S_{j+3}$, and T_{j+3}. Hence, $4r$ new columns are added to $Z_{j-1},\ Y_{j-1}$, and $j-1$ is augmented by an $4r \times 4r$ block at

its diagonal. However, if α_{j+3} is complex, the expressions get even longer and more complicated. Real extensions for the low-rank factors can only be generated if the shift sequence ends with either an α or β shift being real. However, since ADI type iterations are independent of the order of the shifts, these longer relations are not required. ◊

Using the relations and real expansions discovered in the previous section, we are now able to provide a modified G-fADI iteration which takes proper care of complex shifts with respect to the above considered cases. Note that the $\tilde{V}$, $\tilde{S}$ quantities encountered in case 3 do not require additional storage of an $n \times r$ array in a clever implementation. Algorithm 4.4 illustrates the complete G-fADI iteration for generating real low-rank solution factors (G-fADI-r). Moreover, some of the constants calculated for the computation of the W, T matrices can be reused in the $\hat{\Gamma}$ factor. It is also reasonable to test for convergence via the residual norm when the W, T variables are real, i.e., after the current case has been processed. The same holds if one wishes to improve the solution X_j via Galerkin projection ideas [43, 162], or decrease the storage requirements by employing column compression techniques on the low-rank factors [19, 201], which then can be carried out in real arithmetic only.

Avoiding All Complex Operations Algorithm 4.4 computes real low-rank solution factors, but temporarily employs some complex arithmetic operations and storage although the amount is significantly reduced compared to the original Algorithm 3.4. In particular, these are the solutions of the linear systems when a shift is complex, and the corresponding V_j and S_j iterates. These remaining complex instances can be circumvented by rewriting the linear system w.r.t. complex shifts into augmented real ones as in Remark 4.4 and formulate Algorithm 4.4 in terms of Re (V_j), Im (V_j), Re (S_j), and Im (S_j). The solution of the augmented $2n \times 2n$ system can, however, be more expensive compared to solving the original $n \times n$ complex ones.

Another way to work exclusively with real arithmetic operations is given by generalizing the ideas of the LR-ADI-R iteration . Exemplary for $\{\alpha_1, \overline{\alpha_1}\}$ and $\{\beta_1, \overline{\beta_1}\}$, we obtain $V_2 = -\alpha_1 \tilde{V}_1 + \tilde{V}_2$, $S_2 = -\overline{\beta_1}\tilde{S}_1 + \tilde{S}_2$, where

$$\tilde{V}_1 := (AE^{-1}A - 2\,\mathrm{Re}\,(\beta_1)\,A + |\beta_1|^2 E)^{-1}F, \qquad \tilde{V}_2 := E^{-1}(A\tilde{V}_1),$$
$$\tilde{S}_1 := (BC^{-1}B - 2\,\mathrm{Re}\,(\alpha_1)\,B + |\alpha_1|^2 B)^{-T}G, \qquad \tilde{S}_2 := C^{-T}(B^T\tilde{S}_1).$$

It is possible to solve instead $2n \times 2n$ linear systems similar to (4.6). As the numerical experiments in Section 4.1.5 suggest, keeping the $n \times n$ complex system seems to be the most efficient way in the majority of cases and we do not pursue these completely real approaches either.

4.2.2. Special Sylvester Equations

In this section we discuss how to deal with complex shift parameters in the G-fADI modifications proposed in Section 3.3.3 for certain special GCASEs. This will lead to specially tailored variants of Algorithm 4.4.

Algorithm 4.4: G-fADI-r iteration for generating real low-rank solutions of (3.34)

Input : $A,\ B,\ E,\ C,\ F,\ G$ as in (3.34), proper, suitably ordered sets of shift parameters $\{\alpha_1, \ldots, \alpha_{j_{\max}}\}$, $\{\beta_1, \ldots, \beta_{j_{\max}}\}$, tolerance $0 < \tau \ll 1$.

Output: $Z_{j_{\max}} \in \mathbb{R}^{n \times r j_{\max}}$, $Y_{j_{\max}} \in \mathbb{R}^{m \times r j_{\max}}$, $\Gamma_{j_{\max}} \in \mathbb{R}^{r j_{\max} \times r j_{\max}}$ such that $Z_{j_{\max}} \Gamma_{j_{\max}} Y_{j_{\max}}^T \approx X$.

$W_0 = F,\ T_0 = G,\ j = 1.$

while $\|W_{j-1} T_{j-1}^T\| \geq \tau \|F G^T\|$ **do**

  $V_j = (A - \beta_j E)^{-1} W_{j-1},\ S_j = (B - \alpha_j C)^{-H} T_{j-1}.$

  $\gamma_j = \beta_j - \alpha_j.$

  if *Case 1:* $\beta_j \in \mathbb{R} \ \wedge \alpha_j \in \mathbb{R}$ **then**

    $W_j = W_{j-1} + \gamma_j E V_j,\ T_j = T_{j-1} - \overline{\gamma_j} C^T S_j.$

    $Z_j = [Z_{j-1},\ V_j],\ Y_j = [Y_{j-1},\ S_j],\ \Gamma_j = \operatorname{diag}(\Gamma_{j-1}, \gamma_j I_r).$

    $j = j + 1.$

  if *Case 2:* $\beta_j \in \mathbb{C} \ \wedge \alpha_j \in \mathbb{C}$ **then**

    $W_{j+1} = W_{j-1} + 2\operatorname{Re}(\gamma_j) E \operatorname{Re}(V_j) + \left(\frac{|\gamma_j|^2}{\operatorname{Im}(\beta_j)} - 2\operatorname{Im}(\gamma_j)\right) E \operatorname{Im}(V_j).$

    $T_{j+1} = T_{j-1} - 2\operatorname{Re}(\gamma_j) C^T \operatorname{Re}(S_j) - \left(\frac{|\gamma_j|^2}{\operatorname{Im}(\alpha_j)} + 2\operatorname{Im}(\gamma_j)\right) C^T \operatorname{Im}(S_j).$

    $Z_{j+1} = [Z_{j-1},\ \operatorname{Re}(V_j),\ \operatorname{Im}(V_j)],\ Y_{j+1} = [Y_{j-1},\ \operatorname{Re}(S_j),\ \operatorname{Im}(S_j)].$

    $\Gamma_j = \operatorname{diag}\left(\Gamma_{j-1}, \hat{\Gamma}_j^{(2)}\right)$ with $\hat{\Gamma}_j^{(2)}$ from (4.16).

    $j = j + 2.$

  if *Case 3a:* $\beta_j \in \mathbb{C} \ \wedge \ \alpha_j \in \mathbb{R} \ \wedge \ \alpha_{j+1} \in \mathbb{R}$ **then**

    $\gamma_{j+1} = \overline{\beta_j} - \alpha_{j+1},\ \delta_j := \alpha_{j+1}^2 - 2\operatorname{Re}(\beta_j)\alpha_{j+1} + |\beta_j|^2.$

    $W_{j+1} = W_{j-1} + (\operatorname{Re}(\gamma_j + \gamma_{j+1})) E \operatorname{Re}(V_j) + \frac{\operatorname{Re}(\gamma_j)\operatorname{Re}(\gamma_{j+1}) - \operatorname{Im}(\beta_j)^2}{\operatorname{Im}(\beta_j)} E \operatorname{Im}(V_j).$

    $T_j = T_{j-1} - (\operatorname{Re}(\gamma_j) + \operatorname{Re}(\gamma_{j+1})) C^T S_j.$

    $S_{j+1} = (B - \alpha_{j+1} C)^{-T} C^T S_j,\ T_{j+1} = T_j + \delta_j C^T S_{j+1}.$

    $Z_{j+1} = [Z_{j-1},\ \operatorname{Re}(V_j),\ \operatorname{Im}(V_j)],\ Y_{j+1} = [Y_{j-1},\ S_j,\ S_{j+1}].$

    $\Gamma_j = \operatorname{diag}\left(\Gamma_{j-1}, \hat{\Gamma}_j^{(3a)}\right)$ with $\hat{\Gamma}_j^{(3a)}$ from (4.19).

    $j = j + 2.$

  if *Case 3b:* $\beta_j \in \mathbb{R} \ \wedge \ \beta_{j+1} \in \mathbb{R} \ \wedge \ \alpha_j \in \mathbb{C}$ **then**

    $\gamma_{j+1} = \beta_{j+1} - \overline{\alpha_j},\ \delta_j := \beta_{j+1}^2 - 2\operatorname{Re}(\alpha_j)\beta_{j+1} + |\alpha_j|^2.$

    $W_j = W_{j-1} + (\operatorname{Re}(\gamma_j) + \operatorname{Re}(\gamma_{j+1})) E V_j.$

    $V_{j+1} = (A - \beta_{j+1} E)^{-1} E V_j,\ W_{j+1} = W_j + \delta_j E V_{j+1}.$

    $T_{j+1} = T_{j-1} - (\operatorname{Re}(\gamma_j) + \operatorname{Re}(\gamma_{j+1})) C^T \operatorname{Re}(S_j)$

    $\qquad + \frac{-\operatorname{Re}(\gamma_j)\operatorname{Re}(\gamma_{j+1}) + \operatorname{Im}(\alpha_j)^2}{\operatorname{Im}(\alpha_j)} C^T \operatorname{Im}(S_j).$

    $Z_{j+1} = [Z_{j-1},\ V_j,\ V_{j+1}],\ Y_{j+1} = [Y_{j-1},\ \operatorname{Re}(S_j),\ \operatorname{Im}(S_j)].$

    $\Gamma_j = \operatorname{diag}\left(\Gamma_{j-1}, \hat{\Gamma}_j^{(3b)}\right)$ with $\hat{\Gamma}_j^{(3b)}$ from (4.20).

    $j = j + 2.$

Cross Gramian Sylvester Equation A variant of the G-fADI iteration for the cross Gramian Sylvester equation

$$AXE + EXA = FG^T \tag{4.21}$$

is given in (3.48). Obviously, case 3 cannot happen and for α_j, $\alpha_{j+1} = \overline{\alpha_j}$ one gets, using (4.14a)–(4.14d),

$$\begin{aligned}
V_{j+1} &= \overline{V_j} + 2\tfrac{\alpha_j}{\operatorname{Im}(\alpha_j)} \operatorname{Im}(V_j), \\
W_{j+1} &= W_{j-1} - 4\operatorname{Re}(\alpha_j) E \operatorname{Re}(V_j) - 4\operatorname{Re}(\alpha_j)\delta_j E \operatorname{Im}(V_j), \\
S_{j+1} &= \overline{S_j} + 2\tfrac{\overline{\alpha_j}}{\operatorname{Im}(\alpha_j)} \operatorname{Im}(S_j), \\
T_{j+1} &= T_{j-1} + 4\operatorname{Re}(\alpha_j) E^T \operatorname{Re}(S_j) - 4\operatorname{Re}(\alpha_j)\delta_j E^T \operatorname{Im}(W_j).
\end{aligned}$$

with $\delta_j := \operatorname{Re}(\alpha_j) / \operatorname{Im}(\alpha_j)$. The $2r \times 2r$ augmentation of Γ_{j-1} can also be deduced from (4.16):

$$\hat{\Gamma}_j = -4\operatorname{Re}(\alpha_j) \begin{bmatrix} 1 & -\delta_j \\ \delta_j & 2\delta_j^2+1 \end{bmatrix} \otimes I_r = -4\operatorname{Re}(\alpha_j) \begin{bmatrix} 1 & 0 \\ \delta_j & 1 \end{bmatrix} \begin{bmatrix} 1 & \\ & \delta_j^2+1 \end{bmatrix} \begin{bmatrix} 1 & -\delta_j \\ 0 & -1 \end{bmatrix} \otimes I_r,$$

where we assumed that $\alpha_j \in \mathbb{C}_-$, $\forall k$. This factorization of $\hat{\Gamma}_j$ can implicitly be accumulated into the augmentations of Z_{j-1} and Y_{j-1} such that $\hat{\Gamma}_j$ is not required. The resulting modification of the G-fADI-r iteration for solving the cross Gramian equation (3.47) is given in Algorithm 4.5. Note that the equations for W_{j+1} as well as Z_{j+1}, Y_{j+1} are very close to the formulas in Algorithm 4.3 for GCALEs.

Lyapunov Equation with Unsymmetric Inhomogeneity For the matrix equation

$$AXE^T + EXA^T = FG^T \tag{4.22}$$

an adapted G-fADI iteration is given in (3.50), where case 3 can also not occur. If α_j is a complex shift followed by its complex conjugate, we find, using (4.14a),(4.14b), that

$$\begin{aligned}
W_{j+1} &= W_{j-1} - 4\operatorname{Re}(\alpha_j) E \operatorname{Re}(V_j) + 4\tfrac{\operatorname{Re}(\alpha_j)^2}{\operatorname{Im}(\alpha_j)} E \operatorname{Im}(V_j), \\
T_{j+1} &= T_{j-1} + 4\operatorname{Re}(\alpha_j) E \operatorname{Re}(W_j) - 4\tfrac{\operatorname{Re}(\alpha_j)^2}{\operatorname{Im}(\alpha_j)} E \operatorname{Im}(W_j),
\end{aligned}$$

as well as a similar $2r \times 2r$ augmentation of Γ_{j-1} by the

$$\begin{aligned}
\hat{\Gamma}_j &= -4\operatorname{Re}(\alpha_j) \begin{bmatrix} 1 & \delta_j \\ \delta_j & 2\delta_j^2+1 \end{bmatrix} \otimes I_r = -4\operatorname{Re}(\alpha_j) \hat{L}_j \hat{M}_j \hat{L}_j^T, \\
\hat{L}_j &= \begin{bmatrix} 1 & 0 \\ \delta_j & 1 \end{bmatrix} \otimes I_r, \quad \hat{M}_j = \operatorname{diag}\left(1, \delta_j^2 + 1\right) \otimes I_r
\end{aligned}$$

which is the same LDL^T factorization as used in the G-LR-ADI-r iteration for GCALEs ([38], Algorithm 4.3 in Section 4.1.4). The factors $\hat{L}_j$, $\hat{M}_j$ can be again implicitly multiplied to $[\operatorname{Re}(V_j), \operatorname{Im}(V_j)]$ and $[\operatorname{Re}(W_j), \operatorname{Im}(W_j)]$. The resulting G-fADI-r iteration for solving (3.49) is given in Algorithm 4.6. Note that the conjugation of α_j in the linear system has been revoked since it is not important in which order the shifts of a complex conjugated pair are processed.

Algorithm 4.5: G-fADI-r for generating real low-rank solutions of (4.21)

Input : $A,\ E,\ F,\ G$ as in (4.21) and proper shift parameters $\{\alpha_1, \ldots, \alpha_{j_{\max}}\} \subset \mathbb{C}_-$, tolerance $0 < \tau \ll 1$.

Output: $Z_{j_{\max}} \in \mathbb{R}^{n \times r j_{\max}}$, $Y_{j_{\max}} \in \mathbb{R}^{n \times r j_{\max}}$ such that $Z_{j_{\max}} Y_{j_{\max}}^T \approx X$.

$W_0 = F,\ T_0 = G,\ j = 1.$
while $\|W_{j-1} T_{j-1}^T\| \geq \tau \|FG^T\|$ **do**
  $V_j = (A + \alpha_j E)^{-1} W_{j-1},\ S_j = -(A + \alpha_j E)^{-H} T_{j-1}.$
  if $\alpha_j \in \mathbb{R}$ **then**
    $W_j = W_{j-1} - 2\alpha_j E V_j,\ T_j = T_{j-1} + 2\alpha_j E^T S_j.$
    $Z_j = [Z_{j-1}, \sqrt{-2\alpha_j} V_j],\ Y_j = [Y_{j-1}, \sqrt{-2\alpha_j} S_j],\ j = j + 1.$
  else
    $\gamma_j := 2\sqrt{-\operatorname{Re}(\alpha_j)},\ \delta_j := \frac{\operatorname{Re}(\alpha_j)}{\operatorname{Im}(\alpha_j)}.$
    $W_{j+1} = W_{j-1} + \gamma_j^2 E (\operatorname{Re}(V_j) - \delta_j \operatorname{Im}(V_j)).$
    $T_{j+1} = T_{j-1} - \gamma_j^2 E^T (\operatorname{Re}(S_j) + \delta_j \operatorname{Im}(S_j)).$
    $Z_{j+1} = \left[Z_{j-1}, \gamma_j (\operatorname{Re}(V_j) + \delta_j \operatorname{Im}(V_j)),\ \gamma_j \sqrt{(\delta_j^2 + 1)} \cdot \operatorname{Im}(V_j) \right].$
    $Y_{j+1} = \left[Y_{j-1}, \gamma_j (\operatorname{Re}(S_j) - \delta_j \operatorname{Im}(S_j)),\ \gamma_j \sqrt{(\delta_j^2 + 1)} \cdot \operatorname{Im}(S_j) \right].$
    $j = j + 2.$

Discrete-time Lyapunov Equations A low-rank ADI method for the GDALE

$$AXA^T - EXE^T = -FF^T \tag{4.23}$$

was previously derived and given in (3.52). For dealing with complex shifts, let $W_{j-1} \in \mathbb{R}^{n \times r}$ and $\alpha_j,\ \alpha_{j+1} = \overline{\alpha_j} \in \mathbb{D} \backslash \{0\}$. By using similar techniques as above we obtain

$$V_{j+1} = \overline{V_j} + \tfrac{\alpha_j (1 - |\alpha_j|^2)}{\operatorname{Im}(\alpha_j)} \operatorname{Im}(V_j),$$
$$W_{j+1} = W_{j-1} + \tfrac{2 \operatorname{Re}(\alpha_j)(1 - |\alpha_j|^2)}{|\alpha_j|^2} E \operatorname{Re}(V_j) + \tfrac{(1 - |\alpha_j|^2)(|\alpha_j|^2 - |\alpha_j|^4 - 2 \operatorname{Im}(\alpha_j)^2)}{|\alpha_j|^2 \operatorname{Im}(\alpha_j)} E \operatorname{Im}(V_j),$$

and

$$\hat{\Gamma}_j := \sqrt{(1 - |\alpha_j|^2)\gamma_{j+1}} \begin{bmatrix} 1 + |\alpha_j|^2 & \delta_j (1 - |\alpha_j|^2) \\ \delta_j (1 - |\alpha_j|^2) & |\alpha_j|^2 + \delta_j^2 (1 - 2|\alpha_j|^2) + |\alpha_j|^4 (1 + \delta_j^2) \end{bmatrix} \otimes I_r \in \mathbb{R}^{2r \times 2r}$$

with $\gamma_{j+1} = \frac{\gamma_{j-1}}{|\alpha_j|^4},\ \delta_j := \frac{\operatorname{Re}(\alpha_j)}{\operatorname{Im}(\alpha_j)}$. It can be easily shown that $\hat{\Gamma}_j$ is positive definite since we assumed $|\alpha_j| < 1$, and its Cholesky factor can implicitly be multiplied to $[\operatorname{Re}(V_j), \operatorname{Im}(V_j)]$ such that $\hat{\Gamma}_j$ is also not required.

As a small extension of the method proposed in [32, Algorithm 5], we present a further simplification for dealing with pairs of purely imaginary shifts $\alpha_j = \jmath \operatorname{Im}(\alpha_j),\ \alpha_{j+1} = -\jmath \operatorname{Im}(\alpha_j)$, $0 < |\operatorname{Im}(\alpha_j)| < 1$. It is in this case easily established that

$$V_{j+1} = \overline{V_j} + \jmath (1 - \operatorname{Im}(\alpha_j)^2) \operatorname{Im}(V_j),$$
$$W_{j+1} = W_{j-1} + \tfrac{\operatorname{Im}(\alpha_j)^4 - 1}{\operatorname{Im}(\alpha_j)} E \operatorname{Im}(V_j)$$

Algorithm 4.6: G-fADI-r for generating real low-rank solutions of (4.22)

Input : $A,\ E,\ F,\ G$ as in (4.22) and proper shift parameters $\{\alpha_1, \ldots, \alpha_{j_{\max}}\} \subset \mathbb{C}_-$, tolerance $0 < \tau \ll 1$.

Output: $Z_{j_{\max}} \in \mathbb{R}^{n \times r j_{\max}}$, $Y_{j_{\max}} \in \mathbb{R}^{n \times r j_{\max}}$ such that $Z_{j_{\max}} Y_{j_{\max}}^T \approx X$.

1 $W_0 = F,\ T_0 = G,\ j = 1.$
2 **while** $\|W_{j-1} T_{j-1}^T\| \geq \tau \|F G^T\|$ **do**
3 $\quad [V_j,\ S_j] = (A + \alpha_j E)^{-1} [W_{j-1}, -T_{j-1}].$
4 $\quad$ **if** $\alpha_j \in \mathbb{R}$ **then**
5 $\quad\quad [W_j,\ T_j] = [W_{j-1},\ T_{j-1}] + 2\alpha_j E [-V_j,\ S_j].$
6 $\quad\quad Z_j = [Z_{j-1},\ \sqrt{-2\alpha_j} V_j],\ Y_j = [Y_{j-1},\ \sqrt{-2\alpha_j} S_j],\ j = j + 1.$
7 $\quad$ **else**
8 $\quad\quad \gamma_j = 2\sqrt{-\operatorname{Re}(\alpha_j)},\ \delta_j := \frac{\operatorname{Re}(\alpha_j)}{\operatorname{Im}(\alpha_j)}.$
9 $\quad\quad W_{j+1} = W_{j-1} + \gamma_j^2 E (\operatorname{Re}(V_j) - \delta_j \operatorname{Im}(V_j)).$
10 $\quad\quad T_{j+1} = T_{j-1} - \gamma_j^2 E (\operatorname{Re}(S_j) - \delta_j \operatorname{Im}(S_j)).$
11 $\quad\quad Z_{j+1} = \left[Z_{j-1},\ \gamma_j (\operatorname{Re}(V_j) + \delta_j \operatorname{Im}(V_j)),\ \gamma_j \sqrt{(\delta_j^2 + 1)} \cdot \operatorname{Im}(V_j) \right].$
12 $\quad\quad Y_{j+1} = \left[Y_{j-1},\ \gamma_j (\operatorname{Re}(S_j) + \delta_j \operatorname{Im}(S_j)),\ \gamma_j \sqrt{(\delta_j^2 + 1)} \cdot \operatorname{Im}(S_j) \right].$
13 $\quad\quad j = j + 2.$

and, since $\delta_j = 0$, $\hat{\Gamma}_j = \sqrt{(1 - |\alpha_j|^2)\gamma_{j+1}}(1 + |\alpha_j|^2) \operatorname{diag}(I_r, |\alpha_j|^2 I_r)$.

The resulting algorithm for computing real, low-rank solutions factors for (4.23) is given in Algorithm 4.7. The constants $\ell_{1,2,3}$ in the Steps 9, 12, and 13 are the entries of the Cholesky factor of $\hat{\Gamma}_j (\sqrt{\gamma_{j+1}})^{-1}$.

4.2.3. Numerical Examples

Now we test the modification of the G-fADI iteration that exploits the interconnections between the complex iterates. As before we stop the iteration when $\varepsilon_j := \|S_j\| / \|F G^T\| < \tau = 10^{-10}$, where the residual norm was computed using the novel relation (3.45). The shift parameters were computed as in Section 3.3.4.

The following CASE example is constructed entirely for testing purposes and without any background in applications. To obtain the matrix A, we use the example FDM (cf. Section 2.4) with $n_0 = 110$, $f_1 = \mathrm{e}^{\xi_1 + \xi_2}$, $f_2 = 1000\xi_2$, and $f_3 = \xi_1$. Likewise, $-B$ is obtained from using $n_0 = 90$ and $f_1 = \sin(\xi_1 + 2\xi_2)$, $f_2 = 20\mathrm{e}^{\xi_1 + \xi_2}$, and $f_3 = \xi_1 \xi_2$. This yields $n = 12100$ and $m = 8100$ and the matrices $F,\ G$ are random matrices with $r = 4$ columns. We abbreviate this example by *FDM-S* which is similar to [138, Example 2].

For a generalized Sylvester equation we take the example *ifiss16k/4k* introduced in Section 3.3.4 consisting of the matrices from the *ifiss16k* and *ifiss4k* examples.

The special Sylvester equations used in Section 3.3.4 are also reused here and dealt with by the iterations (3.48), (3.50), (3.52) as well as their real implementations (Algorithms 4.5–4.7) proposed in Section 4.2.2.

Algorithm 4.7: G-LR-ADI-r iteration for GDALEs (4.23)

Input : $A,\ E,\ F$ as in (4.23) and proper shift parameters $\{\alpha_1, \ldots, \alpha_{j_{\max}}\}$ with $0 < |\alpha_j| < 1$, tolerance $0 < \tau \ll 1$.

Output: $Z_{j_{\max}} \in \mathbb{R}^{n \times r j_{\max}}$ such that $Z_{j_{\max}} Z_{j_{\max}}^T \approx X$.

$W_0 = F,\ \gamma_0 = 1,\ j = 1.$
while $|\gamma_{j-1}|^2 \|W_{j-1}\|^2 \geq \tau \|F\|^2$ **do**
  $V_j = (A - \frac{\alpha_j}{|\alpha_j|^2} E)^{-1} W_{j-1}.$
  if $\alpha_j \in \mathbb{R}$ **then**
    $W_j = W_{j-1} + \frac{1-\alpha_j^2}{\alpha_j} E V_j,\ \ \gamma_j = \frac{\gamma_{j-1}}{\alpha_j^2}.$
    $Z_j = \left[Z_{j-1}, \sqrt{(1-\alpha_j^2)\gamma_j} V_j \right].$
    $j = j + 1.$
  else
    $\gamma_{j+1} = \frac{\gamma_{j-1}}{|\alpha_j|^4},\ \ell_1 := \sqrt{1 - |\alpha_j|^4}.$
    if $\mathrm{Re}\,(\alpha_j) \neq 0$ **then**
      $W_{j+1} = W_{j-1} + \frac{2\,\mathrm{Re}\,(\alpha_j)(1-|\alpha_j|^2)}{|\alpha_j|^2} E\,\mathrm{Re}\,(V_j) + \frac{(1-|\alpha_j|^2)(|\alpha_j|^2-|\alpha_j|^4-2\,\mathrm{Im}\,(\alpha_j)^2)}{|\alpha_j|^2\,\mathrm{Im}\,(\alpha_j)} E\,\mathrm{Im}\,(V_j).$
      $\delta_j := \frac{\mathrm{Re}\,(\alpha_j)}{\mathrm{Im}\,(\alpha_j)},\ \ell_2 := \ell_1^{-1} \delta_j (1 - |\alpha_j|^2)^2.$
      $\ell_3 := \sqrt{(1-|\alpha_j|^2)(|\alpha_j|^2 + \delta_j^2(1 - 2|\alpha_j|^2) + |\alpha_j|^4 \left(1 + \delta_j^2\right)) - \ell_2^2}.$
      $Z_{j+1} = \left[Z_{j-1}, \sqrt{\gamma_{j+1}} \left(\ell_1\,\mathrm{Re}\,(V_j) + \ell_2\,\mathrm{Im}\,(V_j) \right), \sqrt{\gamma_{j+1}} \ell_3\,\mathrm{Im}\,(V_j) \right].$
    else
      $W_{j+1} = W_{j-1} + \frac{\mathrm{Im}\,(\alpha_j)^4 - 1}{\mathrm{Im}\,(\alpha_j)} E\,\mathrm{Im}\,(V_j).$
      $Z_{j+1} = \left[Z_{j-1}, \sqrt{\gamma_{j+1}} \ell_1\,\mathrm{Re}\,(V_j), \sqrt{\gamma_{j+1}} \ell_1 |\,\mathrm{Im}\,(\alpha_j)\,|\,\mathrm{Im}\,(V_j) \right].$
    $j = j + 2.$

The settings $k_+^A,\ k_-^A,\ k_+^B,\ k_-^B,\ J,\ L$ for the shift parameter computation, the number of the required fADI iteration steps j_{it} until termination, the final scaled residual norm $\rho_{j_{\mathrm{it}}}$, and the computation times $t_{\mathbb{C}}$, $t_{\mathbb{R}}$ in seconds for the standard complex as well as the G-fADI-r iteration (Algorithms 4.4 – 4.7) for computing real, low-rank solutions are summarized in Table 4.2. There, $J_{\mathbb{R}}$, $L_{\mathbb{R}}$ denote the numbers of real α, β shifts whereas $J_{\mathbb{C}}$, $L_{\mathbb{C}}$ refer to the number of complex pairs of shifts. It is evident that for all examples the methods computing real solution factors required, depending on the number of processed complex shifts, substantially less computation time. By exploiting the interconnections between complex iterates up to half of computation time could be saved. The obtained solutions were almost identical, except for possible deviations on the level of rounding errors. This can also be seen in the residual history shown in Figure 4.3 for the *FDM-S* example. Both the complex (Algorithm 3.4) and real version (Algorithm 4.4) have almost identical residual norms throughout the iteration. Only

Table 4.2.: Parameters, required iteration steps j_{it}, final scaled residual norm $\varepsilon_{j_{\text{it}}}$, and computation times $t_{\mathbb{C}}$, $t_{\mathbb{R}}$ in seconds of basic and real implementations of G-fADI iteration and its special modifications.

	Parameters		Results			
Example	$k_+^A, k_-^A, k_+^B, k_-^B$	$J(J_{\mathbb{R}}, J_{\mathbb{C}}), L(L_{\mathbb{R}}, L_{\mathbb{C}})$	j_{it}	$\varepsilon_{j_{\text{it}}}$	$t_{\mathbb{C}}$	$t_{\mathbb{R}}$
FDM-S	10,10,10,10	20 (6,7), 20 (12,4)	70	$6.5{\cdot}10^{-11}$	13.7	8.1
ifiss16k/4k	10,20,10,20	30 (12,9), 30 (12,9)	89	$6.2{\cdot}10^{-11}$	31.2	18.1
ifiss16k (4.21)	10,20,–	30 (12,9), –	110	$4.7{\cdot}10^{-11}$	70.5	40.1
ifiss16k (4.22)	10,20,–	30 (12,9), –	111	$4.01{\cdot}10^{-11}$	54.1	29.3
DALE (4.23)	10,0,–	10 (0,5), –	68	$9.4{\cdot}10^{-11}$	6.4	3.3

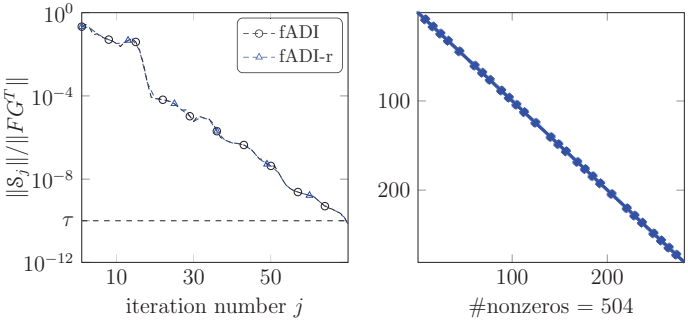


Figure 4.3.: Results for the *FDM-S* example: residual norms against iteration number j for the fADI and fADI-r iteration (left), sparsity pattern of the matrix $\Gamma_{j_{\text{it}}}$ in the fADI-r iteration (right).

minor deviations occurred exactly when the complex method is in between processing a pair of complex conjugated shifts, i.e., cases 2 or 3. This is a similar observation as in [38, Example 1, Figure 1] and Figure 4.2. From the sparsity pattern of the matrix $\Gamma_{j_{\text{it}}}$ in Figure 4.3, one can see that the cases 2 and 3 were encountered 27 times which is revealed by the respective number of $2r \times 2r$ blocks along the diagonal.

4.3. Conclusions

We considered the issue of dealing with complex shift parameters in low-rank ADI methods for both GCALEs as well as GCASEs. Novel interconnections between the iterates corresponding to pairs of complex conjugate shift parameters have been established. Based on these results, modified G-LR-ADI and G-fADI iterations were proposed which are able to compute real, low-rank solution factors in the presence of complex shift pa-

rameters. They achieve this at a reduced amount of complex computations and storage requirements. Similar modifications are also derived for certain special cases of GCASEs. Several numerical tests show that these modified algorithms are indeed more efficient than the standard complex iterations. In the GCALE case they also outperform existing approaches for handling complex shifts in the majority of cases.

CHAPTER 5

SELF-GENERATING ADI SHIFT PARAMETERS

Contents

5.1. Introduction and Motivation

In the previous two chapters we mainly discussed structural properties and their improvement of the low-rank ADI iteration for GCALEs and GCASEs. We have not extensively touched the problem of finding good shift parameters which are crucial for a fast convergence. The dependence on shift parameters is probably the largest disadvantage of ADI methods. Optimal or high quality shifts are usually difficult to obtain,

especially for large-scale problems. Either, they rely on spectral data which is hard to get for large problems, or their generation involves inefficient and expensive computations. Thus, the emphases in the current chapter are new and efficient strategies for computing shift parameters that also lead to fast convergence but without these drawbacks. We especially focus on approaches that are automatic in the sense that they do not require any special a-priori knowledge or setup data. In the following, we start by restricting the discussion to the G-LR-ADI iteration for GCALEs, where we review some existing strategies, which compute shifts before the actual ADI iteration is started, in the next section. After that we present in Section 5.3 novel shift strategies whose goal is to compute shifts adaptively in the course of the G-LR-ADI iteration, ideally without the need for any setup data. These new strategies will be evaluated in several numerical experiments regarding their computational efficiency as well as their influence on the convergence speed of the G-LR-ADI iteration. There, we also compare the G-LR-ADI iteration equipped with the most promising shift approaches with some other, competitive, low-rank methods for GCALEs. Afterwards, in Section 5.4, the shift parameter strategies, both the precomputed and proposed self-generating versions, are adapted to the G-fADI iteration for Sylvester equations. Finally, we conclude and give possible future research perspectives in Section 5.5.

5.2. A Short Overview of Precomputed ADI Shift Parameters

By Lemma 3.4 in Section 3.2, the error of the ADI iteration (3.4) and its low-rank versions (Algorithms 3.1-3.2) after iteration step j is given by

$$X_j - X = \mathcal{A}_j(X_0 - X)\mathcal{A}_j^H, \quad \mathcal{A}_j := \prod_{i=1}^{j} \mathcal{C}_i, \quad \mathcal{C}_i := (A + \alpha_i E)^{-1}(A - \overline{\alpha_i} E). \tag{5.1}$$

Clearly, the convergence speed of the ADI iteration is influenced by the spectral radii $\rho(\mathcal{A}_j)$ of the $\mathcal{A}_j$, see also [119, 202, 233]. One strategy to produce good shifts is to make the radii $\rho(\mathcal{A}_j)$ as small as possible to ensure fast convergence. We briefly reestablish a result from Section 3.2.2. Assuming that all eigenvalues of $(A,\ E)$ are semi-simple and taking norms in the equation above yields

$$\begin{aligned} \frac{\|X_j - X\|}{\|X_0 - X\|} \leq \|\mathcal{A}_j\|^2 &\leq \kappa(U)^2 \left\| \operatorname{diag}\left(\prod_{i=1}^{j} \tfrac{\lambda_1 - \overline{\alpha_i}}{\lambda_1 + \alpha_i}, \ldots, \prod_{i=1}^{j} \tfrac{\lambda_n - \overline{\alpha_i}}{\lambda_n + \alpha_i} \right) \right\|^2 \\ &= \kappa(U)^2 \max_{1 \leq \ell \leq n} \left| \prod_{i=1}^{j} \tfrac{\lambda_\ell - \overline{\alpha_i}}{\lambda_\ell + \alpha_i} \right|^2, \end{aligned}$$

where U is the matrix containing the (right) eigenvectors of $(A,\ E)$ with $AU = EU\Lambda$, $\Lambda = \operatorname{diag}(\lambda_1, \ldots, \lambda_n)$.

A well known result for minimizing the spectral radii of $\mathcal{A}_j$ is, see, e.g., [232, 233], that the optimal shifts $\{\alpha_1, \ldots, \alpha_j\}$ for j iteration steps of (3.4) (and, thus, also of its low-rank versions) are given by the solution of the rational min–max problem

$$\min_{\alpha_1,\ldots,\alpha_j \subset \mathbb{C}_-} \left(\max_{1 \le \ell \le n} \left| \prod_{i=1}^{j} \frac{\lambda_\ell - \overline{\alpha_i}}{\lambda_\ell + \alpha_i} \right| \right), \ \lambda_\ell \in \Lambda(A,\ E), \tag{5.2}$$

which is also referred to as the *ADI shift parameter problem* [231, 232, 233].

One very apparent conceptual issue of using this optimization problem for finding ADI shift parameters is that in the transition from the error expression (5.1) to the min–max problem (5.2), all information regarding the inhomogeneity FF^T of the GCALE is lost. However, we know that the inhomogeneity is of tremendous significance for the existence of low-rank solutions as we discussed in Section 2.3.3, see also, e.g., [184, 4, 113, 222]. In particular, the low-rank of FF^T, i.e, $\operatorname{rank}(F) = r \ll n$, is one influential factor, which is not embraced in (5.2). Moreover, no information regarding the eigenvectors of (A, E) enters (5.2).

Apart from this issues, (5.2) has lead to a number of different shift strategies which are frequently and often also successfully applied in low-rank ADI methods. We briefly describe some of those strategies, which we are also going to employ in our numerical tests.

5.2.1. Wachspress Shifts and Related Approaches

In [233], an analytic solution for (5.2) is proposed which uses

$$a := \min_{1 \le \ell \le n} \operatorname{Re}(\lambda_\ell), \quad b := \max_{1 \le \ell \le n} \operatorname{Re}(\lambda_\ell), \quad \text{and} \quad \psi := \psi(A,\ E) = \max_{1 \le \ell \le n} \arctan \left| \frac{\operatorname{Im}(\lambda_\ell)}{\operatorname{Re}(\lambda_\ell)} \right|$$

for $\lambda_\ell \in \Lambda(A,\ E)$ to estimate the shape of the spectrum $\Lambda(A,\ E)$ via an elliptic functions domain. The computation of optimal shifts to achieve $\|X_j - X\| \le \epsilon$ is then based on elliptic integrals involving the desired tolerance ϵ and the above spectral data a, b, and ψ, see [233, 201] for details. If the spectrum $\Lambda(A,\ E)$ is real or the imaginary parts of the complex eigenvalues are small compared to the real parts, this approach always provides real shift parameters. In the case of large imaginary parts which dominate the real parts, there exists a modification that produces complex shift parameters. We will refer to these shifts as Wachspress shifts in the following. For large-scale matrices the required spectral data, especially the angle ψ for complex spectra, can be hard to obtain exactly. An easy way to get approximate Wachspress shifts [45] (also called suboptimal shifts [201, Section 4.3.2.]) is to approximate $\Lambda(A,\ E)$ by small numbers of k_+ Ritz and k_- inverse Ritz values, i.e., Ritz values w.r.t. $E^{-1}A$ and $A^{-1}E$, respectively. These Ritz values can be computed using two Arnoldi processes. If $A = A^T \prec 0$, $E = E^T \succ 0$, an alternative is to use a single Lanczos process equipped with the E-inner product [6, Chapter 5.5] to get approximation to both small and large eigenvalues. One then computes $a,\ b,\ \psi$ on the basis of this typically small set of Ritz values and

carries out the Wachspress computations as before. This approach will be referred to as approximate Wachspress shifts for which an implementation can be found in [201, Algorithm 4.2]. The quality of these approximate Wachspress shifts depends on the quality of the approximation of a, b, and ψ provided by the Ritz values. Differences in at least one of these values can lead to a considerable different error reduction as it is investigated, e.g., in [202]. Hence, the prescribed values k_+, k_-, but also ϵ, have a certain influence. From a computational point of view, the employed Arnoldi methods introduce additional costs which are dominated by the k_+ and k_- solves with E and A for generating the Ritz and inverse Ritz values. For symmetric systems, i.e. $A = A^T \succ 0$ and $E = E^T \prec 0$, only $a,\ b$ need to be estimated which can be done less costly in one run of a Lanczos process using the inner product induced by E. The computability of a, b, ψ obtained from the Ritz values may be increased by using shifted matrices [45]. The employed Krylov methods also require a starting vector for which there is also no known result on how to choose a suitable one. The authors in [38] used $F\mathbf{1}_r$ in their numerical experiments, but whether there are better choices remains unclear. The computed Ritz values can have positive real parts if $AE^T + EA^T$ is indefinite. These must be neglected.

5.2.2. The Heuristic Penzl Strategy

A frequently used heuristic approach to obtain ADI shifts was proposed by Penzl in [183]. There, $\Lambda(A,\ E)$ is again replaced by a much smaller set $\mathcal{N}$ consisting of Ritz values and reciprocals of Ritz values w.r.t. $E^{-1}A$ and $A^{-1}E$, respectively, also using k_+ and k_- steps of Arnoldi or Lanczos processes. Then, the ADI shift parameter problem is dealt with heuristically in the sense that a prescribed number $J \leq k_+ + k_-$ of elements are chosen from $\mathcal{N}$ that minimize the rational function in (5.2) with $\Lambda(A,\ E)$ replaced by $\mathcal{N}$. The complete procedure for the generation of J shift parameters is illustrated in [183, Algorithm 5.1]. The obtained shifts are often called *heuristic* or *Penzl shifts*. Although this strategy has been used successfully in numerous cases, it comes with several drawbacks similar to the approximate Wachspress shifts. The procedure requires that the values k_+, k_- and here, additionally, J are provided by the user, but there is no known rule how to actually set these values. The same holds for the initial vector of the Arnoldi or Lanczos process. Of course, the quality of the Ritz values influences the quality of the shifts in the end. If the convergence of the Arnoldi or Lanczos processes is slow and the Ritz values are poor approximations of eigenvalues, the shifts may be of poor quality. In some cases, the values k_+, k_- need to be so large that the cost for the Arnoldi or Lanczos processes is non-negligible. Numerical experiments show that the performance of these heuristic shifts is more susceptible to changes in $k_+,\ k_-$ than that of the approximate Wachspress shifts. Even small changes in at least one of these parameters can lead to a significantly differing performance of the G-LR-ADI iteration in the end.

5.2.3. Other Shift Strategies

There exist a number of other shift parameter approaches. For completeness, we mention a few here. For $E = I_n$, an approach based on Leja points is given in [216], where the spectra of $I_n \otimes A^T$ and $A^T \otimes I_n$ are embedded in subsets $\mathcal{E},\ \mathcal{F} \subset \mathbb{C}$. For arbitrary values from $\mathcal{E},\ \mathcal{F}$, shift parameters are recursively obtained by maximizing the rational function in (5.2). A related potential theory based approach can be found in [202]. For real spectra and shifts, an improvement of Penzl's heuristic selection strategy (Section 5.2.2) which introduces marginal additional costs is also proposed in [202, Section 2.2.4]. In [223], a shift strategy is presented which uses the eigenvalues of a small sub-block of A corresponding to the nonzero block of the inhomogeneity FF^T which is present in certain applications. For the case where the considered Lyapunov equation is related to a linear, time-invariant control system, dominant pole based shifts are mentioned in [201, Section 4.3.3.], [28]. The investigations show that these shifts can be beneficial for a subsequent model order reduction process. A number of further shift parameter approaches can be found in [202].

The iterative rational Krylov algorithm (IRKA) [117] is a prominent method for computing reduced order models of large dynamical systems (2.5) which are locally optimal in the $\mathcal{H}_2$-norm. In [21] it is shown, by drawing connections to a Riemannian optimization framework [228], that IRKA can also be used for computing low-rank solutions of large Lyapunov equations. If $A = A^T \prec 0$ and $E = E^T \succ 0$, the obtained approximate solution satisfies an optimality condition w.r.t. a certain energy norm. For the unsymmetric case a similar optimality property holds w.r.t. the residual. In [21, 82, 96] it is shown that this IRKA solutions are identical to the ADI solutions if the shifts provided by IRKA are used as shifts for the G-LR-ADI iteration. These shifts are usually referred to as IRKA-shifts and have attracted some attention recently. They represent a rather theoretical tool because their computation, i.e., running IRKA until a certain stopping criterion is met, is very expensive. Some numerical experiments with IRKA shifts can be found in [39], where they cannot compete with other shift strategies in regard of the computational costs. Moreover, their success appears to be highly dependent on the particular problem as well as on the data used to initialize the IRKA process.

5.3. Self-Generating Shifts

The previously mentioned shift parameters are computed before the actual iteration of G-LR-ADI is started. Here we investigate two approaches to compute shift parameters automatically during the iteration. The first one is based on projecting the coefficient matrices $A,\ E$ onto spaces spanned by certain iterates available in the iteration and then solving the resulting small eigenvalue problem. The second approach aims at finding the shift that minimizes the Lyapunov residual norm in each iteration step.

5.3.1. Shifts Obtained from a Galerkin Projection on Spaces Spanned by LR-ADI Iterates

The heuristic shifts in Section 5.2.2 are essentially Ritz values w.r.t. $A,\ E$ and F. Here we investigate the use of Ritz values which are generated from different spaces where the possibly expensive Krylov subspace construction is not needed. Let $\mathcal{Q} = \operatorname{span}\{Q\} \subset \mathbb{C}^n$ be a k-dimensional subspace with $k \ll n$ and $Q = [q_1, \ldots, q_k] \in \mathbb{C}^{n\times k}$ contain the basis vectors which we assume to be orthonormal. The approaches will be based on a Galerkin projection of $A,\ E$, i.e., the shift will be taken from

$$\Lambda(\tilde{A}, \tilde{E}) \cap \mathbb{C}_-, \quad \tilde{A} := Q^H A Q,\ \tilde{E} := Q^H E Q.$$

The intersection with $\mathbb{C}_-$ ensures that possible unstable eigenvalues of $\Lambda(\tilde{A}, \tilde{E})$ are neglected since those would be of no use in the G-LR-ADI iteration. Alternatively, unstable eigenvalues can just be reflected at the imaginary axis. The matrices $\tilde{A},\ \tilde{E}$ are typically called restrictions of $A,\ E$ onto $\mathcal{Q}$. In the following we present two approaches from [39] where $\mathcal{Q}$ is generated cheaply in the course of the G-LR-ADI iteration. For this we assume that we have already processed j iteration steps and wish to compute shifts for the $j+1$-st and subsequent iteration steps.

Using Already Constructed Parts of the Low-rank Solution Factor

The first idea is to simply take the subspace $\mathcal{Q} = \operatorname{span}\{V_j\}$, where V_j is the last computed block iterate of the G-LR-ADI iteration, and use a matrix Q_j which has orthonormal basis vectors of $\mathcal{Q}$ as columns. In [39] it is proposed to take eigenvalues of $\tilde{A}_j, \tilde{E}_j$ as the shift parameters for the next r iteration steps. The shift parameters constructed that way will in the following be referred to as *V-shifts*. After these r shifts have been used, too, the next shifts are generated from V_{j+r} in the same way and the iteration is continued. In case of a pair of complex conjugate shifts, i.e $\alpha_j = \overline{\alpha_{j-1}}$, V_j and also $\tilde{A},\ \tilde{E}$ are complex matrices. In this situation it is wise to process α_{j-1}, too, and work with $\mathcal{Q} = \operatorname{span}\{[\operatorname{Re}(V_j),\ \operatorname{Im}(V_j)]\}$ instead which will yield $2r$ new shift parameters.

An extension of the original V-shift approach from [39] is mentioned in [54], where all iterates $V_{j-u+1}, \ldots, V_j$, $u \geq 1$, corresponding to the last u processed shifts are used to span the subspace $\mathcal{Q}$. In other words, the last ur columns which were added to Z are used. We refer to this generalization as $V(u)$-shifts, where the number u can be seen as horizon which defines how many previous iteration steps are taken into account. The set of these u shift parameters should be proper, such that it can happen that $r(u+1)$ columns of Z have to be considered.

A further enhancement of the basic V-shift approach from [39] but also of the $V(u)$-shift extension from [54] is obtained by recalling from Corollary 4.6 that

$$AZ_j = W_j G_{\text{ADI-R}}^T + EZ_j B_{\text{ADI-R}}.$$

This also holds similarly for $\hat{Z}_{j,u} := Z(:, (j-u)r+1 : jr)$ by taking the appropriate blocks $\hat{G}_{j,u} := G(: (j-u)r+1 : jr)$ and $\hat{B}_{j,u} := B_{\text{ADI-R}}((j-u)r+1 : jr, (j-u)r+1 : jr)$

of $G_{\text{ADI-R}}$ and $B_{\text{ADI-R}}$, respectively:

$$A\hat{Z}_{j,u} = W_j\hat{G}_{j,u}^T + E\hat{Z}_{j,u}\hat{B}_{j,u}. \tag{5.3}$$

Assume for now that $\hat{Z}_{j,u}$ has full column rank and let $P_j \in \mathbb{R}^{ur\times ur}$ be a nonsingular matrix such that $Q_j := \hat{Z}_{j,u}P_j$ is orthonormal, i.e., $Q_j^TQ_j = I_{ur}$. Then

$$\tilde{A}_j = Q_j^TAQ_j = Q_j^TW_j\hat{G}_{j,u}^TP_j + \tilde{E}_jP_j^{-1}\hat{B}_{j,u}P_j, \quad \tilde{E}_j = Q_j^TEZ_jP_j.$$

Hence, no additional matrix vector products of A with Q_j are required to form the restriction $\tilde{A}_j$. Only ur matrix vector products with E have to be stored or recomputed to build $\tilde{A}_j$ and $\tilde{E}_j$. The number of obtained Ritz values is ur and a smaller number, e.g., r or $2r$, of these values should be used as the next set of shift parameters. We carry out a selection by using the ur Ritz values as inputs for the heuristic Penzl strategy mentioned before.

Using the Residual Factors

Instead of using the space spanned by V_j as projection basis, it is proposed in [39] that one can conceptually also use the GCALE residual factors W_j, i.e., $\mathcal{Q} = \operatorname{span}\{W_j\}$. Using an orthonormal matrix Q_j for $\operatorname{span}\{W_j\}$ and proceeding similarly as above yields r shift parameters, which we call *W-shifts* in the remainder. Unfortunately, Corollaries 3.9 and 4.6 cannot be applied such that it is not possible to construct the small matrices $\tilde{A}_j$, $\tilde{E}_j$ without additional multiplications with A and E.

According to (4.8b), the residual factors are always a real $n \times r$ matrices such that no distinction between real and complex shifts is necessary as for the V-shifts. This, of course, implies that one uses W_{j+1} after a complex conjugated pair $\alpha_j,\ \alpha_{j+1} = \overline{\alpha_j}$ has been fully processed.

Using the Whole Low-rank Solution Factors

For completeness we mention a third approach which uses $\mathcal{Q} = \operatorname{span}\{Z_j\}$ at iteration step j as projection basis and, thus, essentially equals the $V(j)$-shift strategy. There, after j shifts have been processed, jm Ritz values are computed. The selection of a smaller number of shift parameters from these jm Ritz values can be done as for the $V(u)$-shifts.

Clearly, this third variant is significantly more expensive than the V-, $V(u)$-, and W-shifts since computing an orthogonal space for $\operatorname{span}\{Z_j\}$ requires the orthogonalization of the span of V_j for each $j = 1, \ldots, j$ against the previous Z_{j-1}. This might lead to costs that are not negligible anymore and the same holds for the solution of the eigenvalue problem which is now of dimension jr. As for the $V(u)$-shifts, Corollary 4.6 can be used such that savings for constructing the projected matrices $\tilde{A}_j$, $\tilde{E}_j$ can be achieved by storing EZ_j and using (3.24b), see also [235, 237, 234]. We do not pursue this approach further but remark that in [201, 50], $\operatorname{span}\{Z_j\}$ is used to perform a Galerkin projection on the GCALE to improve the convergence behavior of the G-LR-ADI iteration.

Initialization, Orthogonalization, and Other Implementational Considerations

Before the $V(u)$- or W-shifts can be used, G-LR-ADI has to be started with at least one initial shift parameter. In order to explicitly introduce the matrix F in the process, we propose to use an initial Ritz-Galerkin projection with $\mathcal{Q} = \operatorname{span}\{F\}$ which provides r shifts to begin with. Let Q_0 be an orthogonal matrix for $\operatorname{span}\{F\}$. Obviously, building the small, projected matrices $\tilde{A}_0 = Q_0^T A Q_0$, $\tilde{E}_0 = Q_0^T E Q_0$ requires matrix vector products of A, E with Q_0. Another possibility is to use any number of the previously mentioned precomputed shift parameters as initial shifts and start generating $V(u)$- or W-shifts once these have been depleted.

In the considerations we have not specified how to choose the matrix P_j that orthonormalizes V_j, $Z(:, (j-u)r+1 : jr)$, or W_j. We now briefly discuss some possibilities for the orthogonalization of a matrix $N \in \mathbb{R}^{n\times k}$ which can be any of the choices above.

One possibility to obtain the orthonormal basis is to use a thin QR-factorizations, e.g., $N = QR$ such that $P = R^{-1}$. Since $k \ll n$, the numerical cost for their computation is often negligible. One could, e.g., also use a singular value decomposition of N to obtain the orthogonal basis matrix.

It is also possible to employ implicit orthogonalization procedures. The applied Ritz-Galerkin projection for (A, E) works with the orthogonal projection $\mathcal{P} = N(N^T N)^{-1} N^T$. Using the eigenvalue decomposition

$$\hat{N}\hat{D}\hat{N}^T = N^T N, \quad \hat{N}^T\hat{N} = I_k, \quad \hat{D} = \operatorname{diag}(\theta_1, \ldots, \theta_k) \succ 0, \quad \theta_1 \geq \ldots \geq \theta_k$$

leads obviously to $P = \hat{N}\hat{D}^{-\frac{1}{2}}$. The advantage is that computing the eigenvalue decomposition of size $k \times k$ is typically less costly than computing a QR-factorization or SVD of the $n \times k$ matrix N. Moreover, it allows to check N for nearly linearly dependent columns by monitoring the magnitudes of the eigenvalues θ_i, $i = 1, \ldots, k$ and take only those which are larger than a certain tolerance (e.g., $\theta_i > k\theta_1 \mathbf{u}_{\text{mach}}$) and the corresponding eigenvectors in $\hat{N}$ into consideration. If some nearly linearly dependent columns are detected and neglected, the number of obtained new shifts will be smaller than k. We will employ this form of implicit orthogonalization in the remainder. To account for numerical instabilities caused by possible very small magnitudes of the θ_i, it is advised to perform this orthogonalization twice.

We have already mentioned above that $\Lambda(\tilde{A}_j,\ \tilde{E}_j)$ might contain unstable eigenvalues which should not be used. Also, if it happens that $\tilde{E}_j$ is singular, the occurring infinite eigenvalues should be neglected as well. Throwing away nearly linearly depended columns in Q_j usually ensures that $\tilde{E}_j$ is nonsingular. Taking all this into account, it may happen in rare occurrences that $\Lambda(\tilde{A}_j,\ \tilde{E}_j)$ contains no usable shift parameters at all. In that case one can simply reuse the previously used $V(u)$- or W-shifts and try the shift generation again afterwards.

As already stated, the main computational costs for carrying out the $V(u)$-, and W-shift strategies arise in the orthogonalization of an $n \times ru$ matrix as well as solving a small eigenvalue problem whenever new shifts are required. Since $ru \ll n$, this is considered as inexpensive. Only a very small number of additional matrix vector products with

E but no linear system solves are required. This makes both proposed shift strategies significantly cheaper than the previously mentioned precomputed shifts which rely on spectral data generated from an Arnoldi or Lanczos process. A further big advantage of both proposed variants is, compared to the Wachspress and heuristic approaches in Sections 5.2.1–5.2.2, that no setup parameters such as J, k_+, k_- are required which makes these approaches completely automatic and, hence, user-friendly. Additionally, for several numerical tests these shifts even seem to outperform the heuristic shifts. This seems to be the case especially for the $V(u)$-shifts and problems defined by matrix pairs $(A,\ E)$ having a complex spectrum. Problems can occur for GCALEs with a rank-one right hand side, i.e., when $r = 1$. Then, the single shift computed in the W-shift approach is actually a generalized Rayleigh quotient

$$\alpha_{j+1} = \frac{Q_j^T A Q_j}{Q_j^T E Q_j},$$

and, hence, will always be a real number which can be disadvantageous for problems with a complex spectrum. A similar observation can be made for the V-shifts, where, if $\alpha_j \in \mathbb{R}_-$ is the last encountered shift before the computation of new shifts, all subsequent constructed V-shifts will be real as well. In this situation the $V(u)$-shift strategy with $u > 1$ should be employed. Another drawback of the $V(u)$- and W-shifts is the lack of a deeper theoretical foundation. It is also not clear which of the two variants is better, although in most of our numerical tests the $V(u)$-shifts seem to be superior.

Special Cases

In this section we briefly discuss the adaption of the proposed $V(u)$-, and W-shift strategies within the structure exploiting versions of the G-LR-ADI iteration for the special GCALEs mentioned in Section 3.2.5

SO-LR-ADI The SO-LR-ADI iteration (Algorithm 3.3) for dealing with problems related to second order dynamical systems 2.32 can easily be equipped with the $V(u)$- and W- shifts. For instance, consider the augmented block matrices

$$A = \begin{bmatrix} -K & 0 \\ 0 & M \end{bmatrix},\ E = \begin{bmatrix} D & M \\ M & 0 \end{bmatrix} \in \mathbb{R}^{2n \times 2n}, \quad F = \begin{bmatrix} B_1 \\ 0 \end{bmatrix} \in \mathbb{R}^{2n \times r}$$

as in (2.33b), where $M,\ D,\ K \in \mathbb{R}^{n \times n}$, $B_1 \in \mathbb{R}^{n \times r}$. The construction of $\tilde{A}_j,\ \tilde{E}_j$ is implicitly carried out with $A,\ E$, i.e., only matrix vector products with $M,\ D,\ K$ and $n \times r$ matrices are required. By Corollary 4.6 and (5.3), these products are not required for the $V(u)$-shifts. The W-shifts, on the contrary, demand multiplications with A and E. There, some further savings can be achieved at the construction of $\tilde{A}_j$, $\tilde{E}_j$ if $M = M^T$:

$$\begin{aligned} \tilde{A}_j &= Q_j^T A Q_j = -(Q_j^{(\mathrm{p})})^T K Q_j^{(\mathrm{p})} + (Q_j^{(\mathrm{v})})^T Q_M^{(\mathrm{v})}, \quad Q_M^{(\mathrm{v})} := M Q_j^{(\mathrm{v})}, \\ \tilde{E}_j &= Q_j^T E Q_j = (Q_j^{(\mathrm{p})})^T D Q_j^{(\mathrm{p})} + M_s + M_s^T \quad M_s := (Q_j^{(\mathrm{p})})^T Q_M^{(\mathrm{v})}, \end{aligned}$$

where $Q_j^{(\mathrm{p})},\ Q_j^{(\mathrm{v})} \in \mathbb{R}^{n\times r}$ denote the lower and upper n rows of Q_j, adopting the notation used in Section 3.2.5. Analog observations can be made for block matrices of the form (2.33a). Regardless if $V(u)$- or W-shifts are used, the resulting small matrices $\tilde{A}_j,\ \tilde{E}_j$ do not inherit the block structure of $A,\ E$.

SLRCF-ADI Recall that for index one descriptor systems defined by (2.34), the SLRCF-ADI iteration [98] computes a low-rank solution of the GCALE $\hat{A}\hat{P}\hat{E}^T + \hat{E}\hat{P}\hat{A}^T = -\hat{F}\hat{F}^T$ with $\hat{E} := E_{11},\ \hat{A} := A_{11} - A_{12}A_{22}^{-1}A_{21} \in \mathbb{R}^{n_f\times n_f},\ \hat{F} := F_1 - A_{12}A_{22}^{-1}F_2 \in \mathbb{R}^{n_f\times r}$, see (2.35), where these matrices are only dealt with implicitly. Applying the $V(u)$-shifts in this framework is straightforward since by Corollary 4.6 and (5.3) at most ru additional matrix vector products with E_{11} are required. The W-shifts in contrast, require the computation of the matrices

$$Q_j^T\hat{A}Q_j = Q_j^TA_{11}Q_j - Q_j^TA_{12}\left(A_{22}^{-1}\left(A_{21}Q_j\right)\right), \quad Q_j^TE_{11}Q_j$$

such that ru solves with the matrix A_{22} of size $n - n_f$ are needed. This can make the W-shifts substantially more expensive than the V-shifts. Notice that the above matrices have to be used also if the initial shifts are based on $\operatorname{span}\{\hat{F}\}$ which requires one-time r further solves with A_{22}.

A modification of the $V(1)$-shifts proposed in [39] is to carry out the Galerkin projection with the original matrices (2.34) of dimension n and use $\operatorname{span}\left\{V_j^{\mathrm{aug}}\right\}$ with the augmented iterates $V_j^{\mathrm{aug}} := [V_j^T, \Psi^T]^T \in \mathbb{R}^{n\times r}$ from (3.26). We refer to this modification as V^{aug}-shifts. The spectrum of the projected matrices can be deduced from (5.3) and is for $j > 1$ given by

$$\Lambda(Q_j^HAQ_j,\ Q_j^HEQ_j) = \Lambda\left((Q_j^{(1)})^TW_{j-1}, (Q_j^{(1)})^TE_{11}V_j\right) - \alpha_j$$
$$\text{with} \quad V_j^{\mathrm{aug}} = Q_jR_j = \begin{bmatrix} Q_j^{(1)} \\ Q_j^{(2)} \end{bmatrix} R_j$$

for $\alpha_j \in \mathbb{R}_-$ and similarly for complex α_j. This shows that the construction of the V^{aug}-shifts is very close to the $V(1)$-shifts. The exception is the case $j = 1$ for which $\Lambda(Q_j^HAQ_j,\ Q_j^HEQ_j) = \Lambda\left((Q_j^{(1)})^TF_1 + (Q_j^{(2)})^TF_2, (Q_j^{(1)})^TE_{11}V_j\right) - \alpha_j$.

Similarly, we can modify the W-shifts to W^{aug}-shifts and work with the augmented residual factors for the W-shifts

$$W_j^{\mathrm{aug}} = W_{j-1}^{\mathrm{aug}} - 2\operatorname{Re}(\alpha_j)EV_j^{\mathrm{aug}} = \begin{bmatrix} W_j \\ \Upsilon \end{bmatrix},\ W_0^{\mathrm{aug}} = F$$

with an auxiliary matrix $\Upsilon \in \mathbb{C}^{n-n_f\times m}$. A simple calculation exploiting the structure of E shows that $\Upsilon = F_2$. Compared to the $V(1)$- and W-shifts, the V^{aug}- and W^{aug}-shifts are slightly more expensive since they need an orthogonalization of an $n \times r$ instead of an $n - n_f \times r$ matrix. The initial shifts for both V^{aug}- and W^{aug}-shifts are obtained from using an orthonormal basis of F.

5.3.2. Residual-Norm Minimizing Shifts

As shown in Section 3.2.4, the residual in the spectral or Frobenius norm is, combining (3.18b) and (3.19), given by

$$\|\mathcal{L}_j\| = \|W_j\|^2 \quad \text{with} \quad W_j = W_j(\alpha_j) = W_{j-1} - 2\,\mathrm{Re}\,(\alpha_j)\,E\left((A+\alpha_j E)^{-1}W_{j-1}\right).$$

Assume that iteration step $j-1$ is completed and we look for the next shift α_j. Apart from that shift every quantity in the above formula is known after iteration step $j-1$. Hence, an intuitive idea is to choose the next shift α_j as the value that minimizes $\|W_j\|$ because this will also minimize $\|\mathcal{L}_j\|$. Let $\alpha = \nu + \jmath\xi$ with $\nu < 0$ and define the bivariate function

$$\begin{aligned} f_j(\nu,\ \xi) &:= \|W_{j-1} - 2\nu E\left((A+(\nu+\jmath\xi)E)^{-1}W_{j-1}\right)\|^2 \\ &= \|\mathcal{C}(A,E,\alpha)W_{j-1}\|^2, \end{aligned} \tag{5.4}$$

where $\mathcal{C}(A,E,\alpha)$ denotes the generalized Cayley transformation (cf. Definition 2.15b and Proposition 2.16a). Following the idea proposed in [39], the real and imaginary parts of the next shift $\alpha_j = \nu_j + \jmath\xi_j$ can be obtained from solving the minimization problem

$$[\nu_j, \xi_j] = \underset{\nu\in\mathbb{R}_-,\xi\in\mathbb{R}}{\operatorname{argmin}}\ f_j(\nu,\ \xi). \tag{5.5}$$

In other words, the optimal shift α_j is obtained by minimizing the norm of the action of the generalized Cayley transform $\mathcal{C}(A,E,\alpha)$ on the matrix W_{j-1}. If it is known that the spectrum of $A,\ E$ is real, e.g., when $A = A^T \prec 0$, $E = E^T \succ 0$, the ADI shifts will also be real and (5.4) should be simplified accordingly by setting $\xi = 0$. Complex shifts can also be alternatively produced by using the relations (4.8) and minimizing the function

$$g_j(\nu,\ \xi) := \|W_{j+1}\| = \left\| W_{j-1} - 4\nu E\left[\mathrm{Re}\,(V_j) + \frac{\nu}{\xi}\,\mathrm{Im}\,(V_j)\right]\right\|^2, \tag{5.6}$$

where $V_j = (A+(\nu+\jmath\xi)E)^{-1}W_{j-1}$. In that case, the residual norm is minimized with respect to two iteration steps associated with a pair of complex conjugate shifts. Numerical tests did not reveal a significant difference between using (5.4) or (5.6), and, hence, we decided to use the objective function (5.4) for problems for which it is unknown if $\Lambda(A,\ E)$ contains only real eigenvalues. In practice, unless $\Lambda(A,\ E) \subset \mathbb{R}_-$, one usually does not know in advance if the minimizing shift α_j will be real or complex such that the simplification $\xi = 0$ cannot be used. However, the imaginary part ξ_j of the obtained solution α_j of (5.5) should be neglected if it is very small compared to the real parts ν_j, e.g., if $\xi_j/|\alpha_j| \approx 0$.

Assuming we found the globally optimal shift α_j^* satisfying (5.5), the decrease from $\|\mathcal{L}_{j-1}\|$ to $\|\mathcal{L}_j\|$ will be the largest one possible in the sense

$$\|W_j(\alpha_j^*)\|^2 \le \|W_j(\alpha_j)\|^2 \quad \forall \alpha_j \in \mathbb{C}_-.$$

Since we only consider the residual norm minimization from one iteration step to the next one, it does in general not hold that the above inequality also holds for $k > j$. Some remarks regarding a possible minimization with respect to several future iteration steps are given later on.

In the given form these residual-norm minimizing shifts are very difficult to obtain. On the one hand, solving (5.5) is computationally not feasible. Applying any optimization method to handle (5.5) will require to evaluate the function f_j at several values. For each of these values a linear system with r right hand sides has to be solved. Hence, the computation of the optimal shift α_j itself will easily become more expensive than carrying out the current iteration step of the G-LR-ADI iteration.

On the other hand, a further issue is that f_j (and also g_j) might have several local minima. For the best result regarding the reduction of the residual norm $\|\mathcal{L}_j\|$, the global minimum should be used. Taking only a local minima will still decrease $\|\mathcal{L}_j\|$, but the convergence speed of the G-LR-ADI iteration will be slower. Including the constraint $\nu_j \in \mathbb{R}_-$ is essential since f_j will have minima at locations in the right half plane $\mathbb{C}_+$ which cannot be used as shifts for G-LR-ADI. Moreover, f_j has poles in $\mathbb{C}_+$ at $-\Lambda(A,\ E)$. Figure 5.1 illustrates these issues for two small examples. On the left, the CD player example [66] ($n = 120$, $E = I_n$) is used and the plot shows the norm of the scaled Lyapunov residual $\|\mathcal{L}_j\|/\|F\|^2$ at iteration step $j = 18$ of the G-LR-ADI iteration in dependence of real and imaginary part of the shift α_j. The shifts before this iteration step were also global minima of (5.5). Because $A,\ E,\ W_{j-1}$ are real matrices, only the upper half of $\mathbb{C}_-$ is shown. There are clearly two minima of f_j visible: the global one at $\alpha^{\min} \approx -10.34 + \jmath 579.3$ with $\|\mathcal{L}_j\|/\|F\|^2 = 1.26 \cdot 10^{-3}$ and, respectively, a local one at $\alpha \approx -10.34 + \jmath 55.2$ with $\|\mathcal{L}_j\|/\|F\|^2 = 1.29 \cdot 10^{-3}$. Apparently, the difference of the function values at both minima is in the present situation only minor. In other situations the difference might be larger, but the LR-ADI iteration will still converge as long as $\alpha \in \mathbb{C}_-$. Hence, the possibility of detecting only local minima in $\mathbb{C}_-$ is not a severe issue. For the second example, we use a centered finite-difference discretization of the negative Laplace operator on $[0,1]^2$. Using Dirichlet boundary conditions and 10 grid points yields $n = 100$. Since $A = A^T \prec 0$, only real values for α were used and the right plot in Figure 5.1 shows the scaled residual norm for the first iteration, i.e., $\|\mathcal{C}(A,\alpha)F\|/\|F\|^2$. Clearly, there are local minima located in $\mathbb{R}_+$ between the singularities at $\alpha = -\lambda_i(A)$. These minima should not be used as shift because $\rho(\mathcal{C}(A,\alpha)) > 1$ which can deteriorate the performance of the LR-ADI iteration or even lead to divergence. Using a constrained optimization routine as mentioned above can ensure that the detected minima are stable. Of course, one could, similar to the V- and W-shifts, simple negate any computed unstable shift. However, the restriction to the open left half plane ensures that (5.4) is continuous.

Another issue that one might face in the case $r > 1$ is that the function (5.4) might not be differentiable at some values α. Further details on this issue and how to overcome it are given later.

In the following we prioritize strategies that aim at reducing the costs of the numerical realization of computing these residual-norm minimizing shifts.

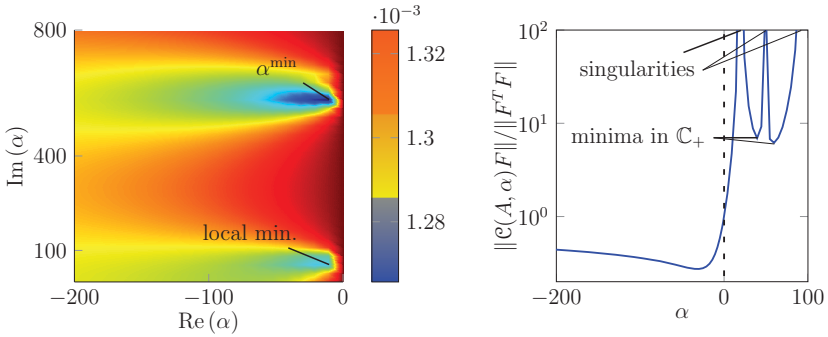


Figure 5.1.: Plot of $\|\mathcal{C}(A,\alpha)W_{j-1}\|/\|F\|^2$ in $\mathbb{C}_-$ and $\mathbb{R}$ for CD-Player (left) and, respectively, 2D-Laplacian (right).

Enhancing the Computation of the Residual Norm-Minimizing Shifts

Here we present some strategies for improving the generation of the residual-norm minimizing shifts. We mainly restrict to the strategy where the minimization problem (5.5) is solved by any form of smooth optimization algorithm which requires evaluations of the objective function and possibly also of its derivatives.

Including First and Second Order Derivatives Several optimization methods for finding $\min f$ are based on an iterative solution of the first order optimality system $\nabla f = 0$, provided f is sufficiently smooth. Hence, adding explicitly computed gradients and Hessians of (5.4) can help to accelerate the iteration of the employed optimization routine. Note that since the only constraints we will consider are upper and lower bounds for the optimization variables, the gradient and Hessian of the original objection function can be used for the constrained optimization problem. We omit the iteration index j if this does not lead to confusion. The following theorem states how the gradient and Hessian of (5.4) can be expressed and computed.

Theorem 5.1 (Gradient and Hessian of the objective function (5.4)):
Let $\alpha = \nu + \jmath\xi \in \mathbb{C}_-$, $W \in \mathbb{R}^{n\times r}$, and define $L(\nu,\xi) := A + \alpha E$,

$$W_A := EL(\nu,\xi)^{-1}W, \qquad W_{A^i} := EL(\nu,\xi)^{-1}W_{A^{i-1}}, \qquad W_+ := W - 2\nu W_A,$$
$$\hat{W} := W_A - \nu W_{A^2}, \qquad W_{(i)} := W_{A^i} - 2\nu W_{A^{i+1}},$$
$$\tilde{R}_\nu := -W_+^H\hat{W}, \qquad \tilde{R}_\xi := W_+^H W_{A^2}$$

for $i = 1, 2$. Moreover, assume $W_+^H W_+$ has r distinct eigenvalues and let $(\theta_k\ , u_k) = (\theta_k(\nu,\ \xi)\ , u_k(\nu,\ \xi))$ with $\|u_k\| = 1$, $k = 1,\dots,r$, be its eigenpairs ordered such that

$\theta_1 > \ldots > \theta_r > 0$. Then, gradient and Hessian of (5.4) are given by

$$\nabla f = 4 \begin{bmatrix} \operatorname{Re}\left(u_1^H \left(W_+^H \hat{W}\right) u_1\right) \\ -\nu \operatorname{Im}\left(u_1^H \left(W_+^H W_{A^2}\right) u_1\right) \end{bmatrix} = 4 \begin{bmatrix} \operatorname{Re}\left(u_1^H \tilde{R}_\nu u_1\right) \\ -\nu \operatorname{Im}\left(u_1^H \tilde{R}_\xi u_1\right) \end{bmatrix} \tag{5.7}$$

and

$$\begin{aligned} \nabla^2 f =& 8 \begin{bmatrix} \operatorname{Re}\left(u_1^H \left((W_{A^2} - \nu W_{A^3})^H W_+ + \hat{W}^H \hat{W}\right) u_1\right) & h_{12} \\ h_{12} & \nu \operatorname{Re}\left(u_1^H (W_{A^3}^H W_+ + \nu W_{A^2}^H W_{A^2}) u_1\right) \end{bmatrix} \\ &+ \sum_{k=2}^{r} \frac{8}{\theta_1 - \theta_k} \begin{bmatrix} |u_1^H (\tilde{R}_\nu^H + \tilde{R}_\nu) u_k|^2 & \tilde{h}_{12}^{(k)} \\ \tilde{h}_{12}^{(k)} & |u_1^H (\tilde{R}_\xi^H - \tilde{R}_\xi) u_k|^2 \end{bmatrix} \end{aligned} \tag{5.8}$$

$$\begin{aligned} \text{with} \quad h_{12} &:= \frac{1}{2} \operatorname{Im}\left(u_1^H W_{(2)}^H W_+ - 2\nu W_{A^2}^H W_{(1)} u_1\right), \\ \tilde{h}_{12}^{(k)} &:= -\operatorname{Re}\left((u_1^H (\tilde{R}_\nu^H + \tilde{R}_\nu) u_k)(\jmath \nu u_k^H (\tilde{R}_\xi^H - \tilde{R}_\xi) u_1)\right). \end{aligned} \qquad \lozenge$$

Proof. With $\hat{\mathcal{C}}(A, E, \nu, \xi) = \mathcal{C}(A, E, \alpha = \nu + \jmath\xi)$, the objective function (5.4) can be expressed equivalently as

$$\begin{aligned} f(\nu,\ \xi) &= \sigma_{\max}^2 \left(\hat{\mathcal{C}}(A, E, \nu, \xi) W\right) = \lambda_{\max}\left(\mathcal{F}(\nu,\ \xi)\right), \\ \text{where} \quad \mathcal{F}(\nu,\ \xi) &:= W^T \hat{\mathcal{C}}(A, E, \nu, \xi)^H \hat{\mathcal{C}}(A, E, \nu, \xi) W = W_+^H W_+ \in \mathbb{C}^{r \times r}. \end{aligned} \tag{5.9}$$

It is clear that $W_+ = W - 2\nu E L(\nu, \xi)^{-1} W$ is the factor of the Lyapunov residual w.r.t. the shift $\alpha = \nu + \jmath\xi$. For a parameter-dependent matrix $M(s,t)$ with distinct eigenvalues, it holds by [149, 173] for the eigenvalues $\theta(s,t)$ that $\frac{\partial \theta(s,t)}{\partial s} = y(s)^H \frac{\partial M(s,t)}{\partial s} x(s)$ and similarly for $\frac{\partial \theta(s,t)}{\partial t}$. Here, $y(s,t)$ and $x(s,t)$ are the left and right eigenvectors of $M(s,t)$ scaled such that $y(s,t)^H x(s,t) = 1$. In our situation, the parameter-dependent matrix $\mathcal{F}(\nu,\ \xi)$ is Hermitian such that $x(s,t) = y(s,t)$. Hence, the partial derivatives of $f(\nu,\ \xi)$ are

$$\frac{\partial f}{\partial \nu} = u_1^H \left(\frac{\partial}{\partial \nu} \mathcal{F}(\nu,\ \xi)\right) u_1, \quad \frac{\partial f}{\partial \xi} = u_1^H \left(\frac{\partial}{\partial \xi} \mathcal{F}(\nu,\ \xi)\right) u_1, \tag{5.10}$$

where u_1 is the eigenvector which corresponds to the largest eigenvalue θ_1 of $\mathcal{F}(\nu,\ \xi)$. Since

$$\hat{\mathcal{C}}(A, E, \nu, \xi)^H = E^{-T} \hat{\mathcal{C}}(A^T, E^T, \nu, -\xi) E^T$$

it is easy to see that the derivatives of $\mathcal{F}$ are

$$\begin{aligned} \frac{\partial}{\partial \nu} \mathcal{F}(\nu,\ \xi) &= W^T \left(\mathcal{G}_\nu^H + \mathcal{G}_\nu\right) W, \quad \mathcal{G}_\nu := \hat{\mathcal{C}}(A, E, \nu, \xi)^H \frac{\partial}{\partial \nu} \hat{\mathcal{C}}(A, E, \nu, \xi), \\ \frac{\partial}{\partial \xi} \mathcal{F}(\nu,\ \xi) &= W^T \left(\mathcal{G}_\xi^H + \mathcal{G}_\xi\right) W, \quad \mathcal{G}_\xi := \hat{\mathcal{C}}(A, E, \nu, \xi)^H \frac{\partial}{\partial \xi} \hat{\mathcal{C}}(A, E, \nu, \xi), \end{aligned} \tag{5.11}$$

Both partial derivatives of the Cayley transformation are

$$\begin{aligned}\frac{\partial}{\partial\nu}\hat{\mathcal{C}}(A,E,\nu,\xi) &= -2EL(\nu,\xi)^{-1}(I-\nu EL(\nu,\xi)^{-1}),\\ \frac{\partial}{\partial\xi}\hat{\mathcal{C}}(A,E,\nu,\xi) &= 2\jmath\nu EL(\nu,\xi)^{-1}EL(\nu,\xi)^{-1},\end{aligned}$$

where we used $\hat{\mathcal{C}}(A,E,\nu,\xi) = I - 2\nu EL(\nu,\xi)^{-1}$. The abbreviations defined above yield

$$\begin{aligned}W^T\mathcal{G}_\nu W &= -2W^T\big(\hat{\mathcal{C}}(A,E,\nu,\xi)^H EL(\nu,\xi)^{-1}(I-\nu EL(\nu,\xi)^{-1})\big)W\\ &= 2W_+^H(W_A - \nu W_{A^2}) = 2\tilde{R}_\nu\\ W^T\mathcal{G}_\xi W &= -2\jmath\nu W^T\big(\hat{\mathcal{C}}(A,E,\nu,\xi)^H EL(\nu,\xi)^{-1}EL(\nu,\xi)^{-1}\big)W = 2\jmath\nu W_+^H W_{A^2} = 2\jmath\nu\tilde{R}_\xi\end{aligned}$$

and with (5.10), (5.11)

$$\nabla f = \begin{bmatrix} 2u_1^H(\tilde{R}_\nu^H + \tilde{R}_\nu)u_1 \\ 2\jmath\nu u_1^H(-\tilde{R}_\xi^H + \tilde{R}_\xi)u_1\end{bmatrix} = 4\begin{bmatrix}\operatorname{Re}\left(u_1^H\tilde{R}_\nu u_1\right)\\ -\nu\operatorname{Im}\left(u_1^H\tilde{R}_\xi u_1\right)\end{bmatrix}$$

and, consequently, (5.7) follows immediately. For the Hessian of f recall from, e.g., [149], [173, Lemma 2.1.(iv)] that the second order derivatives of the eigenvalue $\theta_1(s,\ t)$ of the matrix $M(s,\ t)\in\mathbb{C}^{r\times r}$ are given by

$$\frac{\partial^2\theta_1(s,\ t)}{\partial s\partial t} = y_1^H\frac{\partial^2 M(s,\ t)}{\partial s\partial t}u_1 + 2\operatorname{Re}\left(\sum_{k=2}^{r}\frac{\left(y_1^H\frac{\partial M(s,\ t)}{\partial s}u_k\right)\left(y_k^H\frac{\partial M(s,\ t)}{\partial t}u_1\right)}{\theta_1-\theta_k}\right).$$

As above, (θ_i,u_i,y_i), $i=1,\dots,r$, are the eigentriplets of $M(s,t)$ scaled such that $y_i^Hu_i = 1$. For the second partial derivative w.r.t. ν we have, exploiting $y_i = u_i$ again and using (5.11),

$$\begin{aligned}f_{\nu\nu} &= u_1^H\frac{\partial^2\mathcal{F}(\nu,\ \xi)}{\partial\nu^2}u_1 + 2\operatorname{Re}\left(\sum_{k=2}^{r}\frac{\left(u_1^H\frac{\partial\mathcal{F}(\nu,\ \xi)}{\partial\nu}u_k\right)\left(u_k^H\frac{\partial\mathcal{F}(\nu,\ \xi)}{\partial\nu}u_1\right)}{\theta_1-\theta_k}\right)\\ &= u_1^HW^T\frac{\partial}{\partial\nu}\left(\mathcal{G}_\nu^H+\mathcal{G}_\nu\right)Wu_1 + \sum_{k=2}^{r}\tfrac{8}{\theta_1-\theta_k}|u_1^H(\tilde{R}_\nu^H+\tilde{R}_\nu)u_k|^2.\end{aligned}\tag{5.12}$$

Furthermore,

$$\begin{aligned}\frac{\partial}{\partial\nu}\mathcal{G}_\nu =& \frac{\partial}{\partial\nu}\hat{\mathcal{C}}(A,E,\nu,\xi)^H\frac{\partial}{\partial\nu}\hat{\mathcal{C}}(A,E,\nu,\xi) + \hat{\mathcal{C}}(A,E,\nu,\xi)^H\frac{\partial^2}{\partial\nu^2}\hat{\mathcal{C}}(A,E,\nu,\xi),\\ \frac{\partial^2}{\partial\nu^2}\hat{\mathcal{C}}(A,E,\nu,\xi) =& -2\frac{\partial}{\partial\nu}EL(\nu,\xi)^{-1}(I-\nu EL(\nu,\xi)^{-1})\\ =& 2\left(EL(\nu,\xi)^{-1}\right)^2(I-\nu EL(\nu,\xi)^{-1})\\ &+2\left(EL(\nu,\xi)^{-1}\right)^2 - 2\nu\left(EL(\nu,\xi)^{-1}\right)^3\\ =& 4EL(\nu,\xi)^{-1}EL(\nu,\xi)^{-1} - 4\nu EL(\nu,\xi)^{-1}EL(\nu,\xi)^{-1}EL(\nu,\xi)^{-1}\end{aligned}$$

Algorithm 5.1: Evaluation of f, ∇f and $\nabla^2 f$ at $\alpha = \nu + \jmath\xi$

Input : $A,\ W,\ E,\ \nu,\ \xi$.

Output: $f(\nu,\ \xi)$, $\nabla f(\nu,\ \xi)$, $\nabla^2 f(\nu,\ \xi)$.

Compute $W_A = E\left((A + (\nu + \jmath\xi)E)^{-1}W\right)$ and overwrite $W = W - 2\nu W_A$.

if $r := coldim(W) > 1$ **then**

  Compute eigendecomposition $U^H W^H W U = \operatorname{diag}(\theta_1, \ldots, \theta_r)$ with $U^T U = I_r$, $\|u_i\| = 1$, $\theta_1 > \ldots \theta_r > 0$. Set $f = \theta_1$.

  Overwrite $W = WU$, $W_A = W_A U$.

else

  Set $f = \|W\|^2$.

Compute $W_{A^2} = E\left((A + (\nu + \jmath\xi)E)^{-1}W_A\right)$.

Set $\nabla f = \begin{bmatrix} 4\operatorname{Re}\left(\hat{R}_\nu(1,1)\right) \\ -4\nu \operatorname{Im}\left(\hat{R}_\xi(1,1)\right) \end{bmatrix}$, $\hat{R}_\nu := (-W_A + \nu W_{A^2})^H W$, $\hat{R}_\xi := W^H W_{A^2}$.

Compute $W_{A^3} = E\left((A + (\nu + \jmath\xi)E)^{-1}W_{A^2}\right)$.

Define $\hat{W} := W_A - \nu W_{A^2}$, $W_{(i)} := W_{A^i} - 2\nu W_{A^{i+1}}$, $i = 1, 2$.

$$\nabla^2 f = 8\begin{bmatrix} \operatorname{Re}\left((W_{A^2}(:,1) - \nu W_{A^3}^H(:,1))W(:,1) + \hat{W}^H(:,1)\hat{W}(:,1)\right) & \frac{1}{2}\operatorname{Im}\left(W_{(2)}^H(:,1)W(:,1) - 2\nu W_{A^2}^H(:,1)W_{(1)}(:,1)\right) \\ \frac{1}{2}\operatorname{Im}\left(W_{(2)}^H(:,1)W(:,1) - 2\nu W_{A^2}^H(:,1)W_{(1)}(:,1)\right) & \nu\operatorname{Re}\left(W_{A^3}^H(:,1)W(:,1) + \nu W_{A^2}^H(:,1)W_{A^2}(:,1)\right) \end{bmatrix}$$

if $r > 1$ **then**

  for $k = 2, \ldots, r$ **do**

$$\nabla^2 f = \nabla^2 f + \frac{8}{\theta_1 - \theta_k}\begin{bmatrix} \left|\overline{\hat{R}_\nu(k,1)} + \hat{R}_\nu(1,k)\right|^2 & \tilde{h} := \operatorname{Re}\left(\left(\overline{\hat{R}_\nu(k,1)} + \hat{R}_\nu(1,k)\right)\jmath\nu\left(\overline{\hat{R}_\xi(k,1)} - \hat{R}_\xi(1,k)\right)\right) \\ \tilde{h} & \nu^2\left|\overline{\hat{R}_\xi(k,1)} - \hat{R}_\xi(1,k)\right|^2 \end{bmatrix}$$

such that by incorporating the introduced notation

$$\begin{aligned} W^T\frac{\partial}{\partial\nu}\mathcal{G}_\nu W &= 4(W_A - \nu W_{A^2})^H(W_A - \nu W_{A^2}) + 4W_+^H(W_{A^2} - \nu W_{A^3}) \\ &= 4\left(\hat{W}^H\hat{W} + W_+^H(W_{A^2} - \nu W_{A^3})\right). \end{aligned} \tag{5.13}$$

Setting (5.13) into (5.12) leads, after some basic manipulations, to the $(1,1)$ entry of $\nabla^2 f$. The other entries, i.e., the partial derivatives $f_{\nu\xi}$, $f_{\xi\xi}$, can be found using similar steps yielding the Hessian formula (5.8). □

Obviously, Theorem 5.1 reveals that to evaluating both ∇f and $\nabla^2 f$ at ν and ξ, requires the solution of two additional linear systems with the coefficient matrix $(A + (\nu + \jmath\xi)E)$ and r right hand sides. The extra work is usually compensated by the significant lower number of iteration steps of the optimization routine. If only ∇f is needed or sufficient, a closer inspection of the gradient formula (5.7) reveals that only $W_{A^2}u_1$ is required, i.e., only a single right hand side $W_A u_1$ is needed in the additional linear system for

W_{A^2}. A clever incorporation of the necessary eigenvectors u_i, $i = 1, \ldots, r$, and reuse of already computed quantities leads to the following pseudo code in Algorithm 5.1 which illustrates the evaluation of the objective function, its gradient and its Hessian at the point $\alpha = \nu + \jmath\xi$.

To obtain a new shift α_j for the G-LR-ADI iteration for the use in step j, we can use W_{j-1} and call Algorithm 5.1 from within any optimization algorithm. In the upcoming numerical experiments we employ the `fmincon` routine which is part of the MATLAB optimization toolbox™. It comes with different available optimization algorithms from which the *interior-point* and *trust-region-reflective* (see, e.g., [177]) algorithms led to the best results for our purpose. An often used stopping criterion in optimization methods is to monitor the gradient of the objective function and stop the iteration, e.g., when $\|\nabla f\| \leq \tau_{\text{opt}}$. The tolerance $0 < \tau_{\text{opt}} \ll 1$ should be chosen sufficiently small w.r.t. our goal $\|\mathcal{L}\| = \|W\|^2 \leq \tau \|F^T F\|$ and we use $\tau_{\text{opt}} < \tau$. Since f will become small in the course of the iteration, it can be wise to work with a scaled version instead to account for numerical difficulties, e.g., dividing by the residual norm of the previous iteration step $f/\|W_{j-1}\|^2$.

Dealing with Non-Differentiability The assumption in Theorem 5.1 that the parameter dependent function $\mathcal{F}(\nu, \xi)$ in (5.9) has r distinct eigenvalues for all ν, ξ is rather strict but crucial for our purpose as it ensures analyticity of the parameter dependent eigenvalues, i.e., the existence of the derivatives of the objective function. In practice this is, however, difficult to ensure since it can happen for some values of ν, ξ that, e.g., the largest and second largest eigenvalue θ_1, θ_2 coalesce. It is, in fact, often observed that this coalescence occurs exactly at the minimum of f. Thus, $\theta_1(\nu, \xi)$ will not be differentiable at this point which essentially prohibits applying smooth optimization techniques based on the first order optimality conditions. For more details regarding these issues and also possible alternative optimization approaches we refer to, e.g., [180, 159, 173, 174] and the references therein. If $r = 1$, the optimization problem reduces to a scalar one and such issues are not present. Moreover, if we are only interested in real shifts, i.e. the variable ξ is not considered, conditions ensuring the analyticity of the largest eigenvalue of univariate Hermitian parameter dependent matrices are mentioned in [173, 174]. The situation for the multivariate case, i.e. an optimal complex shift is sought in our application, is more difficult. To tackle these potential issues regarding $\theta_1(\nu, \xi)$ not being differentiable, we simply transform our problem to a scalar one by enforcing $r = 1$ in the optimization algorithm. This is easily achieved by multiplying the residual factor W with an appropriate vector $d \in \mathbb{R}^r$ before we enter the optimization routine. The computation effort w.r.t. the gradients and Hessian matrices in Algorithm 5.1 is also relaxed by this simplification since the Lines 2–4 and 12–14 are not required. An obvious choice is to set d equal to the left singular vector corresponding to the largest singular value of W. With respect to the upcoming projection approach, this transformation to a scalar optimization problem represents an additional layer of approximation of (5.5). In several numerical experiments this approximation does not appear to significantly deteriorate the quality of the obtained shift parameters. It is, of course, also possible to

use derivative-free optimization routines to avoid computing gradients and Hessians at all, as is was done in [39].

Decreasing the Numerical Costs by Projection The main issue with the presented residual-norm minimizing shifts is that solving the optimization problem involves expensive evaluations of the function (5.4). The main numerical effort of these evaluations stems from the solutions of the inherent linear systems $(A+\alpha E)\hat{V} = W_{j-1}$ for $\hat{V}$. If the optimization algorithm needs j_{opt} iteration steps to detect the local minimum of f and if gradient as well as Hessian information is used, approximately $3j_{\text{opt}}$ linear systems have to be solved to obtain a single shift parameter. This number reduces to $2j_{\text{opt}}$ if only the function and its gradient are used. Here, we present strategies for reducing the costs of the linear systems by performing a projection approach which leads to an approximate objective function and, thus, the obtained shift parameters will be referred to as approximate residual-norm minimizing shifts.

The first idea is to use, exactly as for the V and $V(u)$-shifts, the spaces spanned by block columns of the low-rank solution factor Z corresponding to one or more previous iteration steps. With the matrix $Q_j \in \mathbb{R}^{n\times ru}$ containing the orthonormal basis vector of these spaces, we replace $A,\ E,\ W_j$ in (5.4) by the projected matrices $\tilde{A}_j = Q_j^T A Q_j$, $\tilde{E}_j = Q_j^T E Q_j$, $\tilde{W}_j := Q_j^T W_j$ to obtain a reduced objective function

$$\tilde{f}(\nu,\ \xi) := \|\tilde{W}_j - 2\nu\tilde{E}_j\left((\tilde{A}_j + (\nu + \jmath\xi)\tilde{E}_j)^{-1}\tilde{W}_j\right)\|^2. \tag{5.14}$$

By using the same replacements in (5.7), (5.8), also reduced gradients $\nabla\tilde{f}$ and Hessians $\nabla^2\tilde{f}$ can be constructed. The matrices $\tilde{A}_j,\ \tilde{E}_j$ are constructed efficiently as we showed for the $V(u)$-shifts, see (5.3), Corollaries 3.9, 4.6. Since the linear systems to be solved are now of dimension $ur \ll n$, evaluating $\tilde{f}, \nabla\tilde{f}, \nabla^2\tilde{f}$ by Algorithm 5.1 is much less costly as with the original matrices. These approximate residual-norm minimizing shifts can be seen as an augmentation of the $V(u)$-shift approach. Although there is no guarantee that $\tilde{f}(\nu,\ \xi) \approx f(\nu,\ \xi)$, our numerical experiments show that the approximate residual-norm minimizing shifts can lead to a comparable or even better convergence speed compared to other shift strategies. Increasing the horizon u typically increases the quality of the obtained shifts. Especially for the situation $r = 1$ choosing $u > 1$ is recommended.

An alternative and also obvious approach is to use (block) Krylov subspaces for the projection. For this, one can exploit the fact that only the shift α in (5.4) changes but the right hand sides remain constant. For $E = I_n$, $W = W_{j-1} \in \mathbb{R}^n$ $(r = 1)$, the well known shift-invariance of Krylov spaces provides that $\mathcal{K}_k(A,\ W) = \mathcal{K}_k(A+\alpha I_n,\ W)$ for any scalar α. Let $\mathcal{K}_k(A,\ W) = \operatorname{span}\{Q_k\}$ with $Q_k^T Q_k = I_k$ and $\tilde{A}_k := Q_k^T A Q_k \in \mathbb{R}^{k\times k}$ the restriction of A onto $\mathcal{K}_k(A,\ W)$. This enables to extract approximate solution to all shifted linear systems

$$(A + \alpha I_n)\tilde{V}^\alpha = W. \tag{5.15}$$

from one single Krylov subspace $\mathcal{K}_k(A,\ W)$. For instance, if a Krylov solver based on a Galerkin condition such as the full orthogonalization method (FOM) [199] and a zero

initial condition is used, the approximate solutions of the linear systems extracted from $\mathcal{K}_k(A,\ W)$ can be represented as

$$\tilde{V}^{\alpha} \approx \tilde{V}_k^{\alpha} = Q_k((\tilde{A}_k + \alpha I_k)^{-1} Q_k^T W), \tag{5.16}$$

which basically corresponds to the multi-shift FOM approach [208]. Hence, for every shift α we obtain approximate solutions from the same matrices Q_k, $\tilde{A}_k$, which have to be constructed only once. For each shift, the approximate solution $\tilde{V}^{\alpha}$ is obtained by solving a k dimensional linear system. Numerous other Krylov subspace methods for (5.15) built upon the shift-invariance exist and might be used as well, e.g., multi-shift variants of GMRES [103], IDR(s) [85, 226], BiCG [2], or BiCGstab(ℓ) [102], to name a few. Using as example the approximations by the multi-shift FOM approach in (5.16), leads to an approximation of the objective function (5.4) of the form

$$f(\nu,\ \xi) \approx f_k(\nu,\ \xi) := \|W - 2\nu \tilde{V}_k^{\nu+\jmath\xi}\|^2 \tag{5.17}$$

$$= \|W\| \|e_1 - 2\nu \left((\tilde{A}_k + (\nu + \jmath\xi) I_k)^{-1} e_1 \right) \|^2, \tag{5.18}$$

where we exploited the orthogonality of Q_k and $Q_k^T W = \|W\| e_1$. Using Algorithm 5.1 with $\tilde{A}_k$, I_k, e_1 to evaluate f_k and its derivatives, is now, exactly as with the spaces generated in the $V(u)$ approach, much cheaper compared to (5.4). For $r > 1$ everything can be carried out by employing block Krylov subspace methods which leads to $\tilde{A}_k \in \mathbb{R}^{rk \times rk}$. The usage of preconditioners, which is usually mandatory to achieve a fast convergence of Krylov methods for linear systems, unfortunately destroys the shift-invariance in general. For special classes of preconditioners, the shift-invariance can be restored which leads to preconditioned multi-shift Krylov methods see, e.g., [85, 226, 2]. We do not pursue this topic further since a sufficient discussion of the usage of different multi-shift Krylov methods as well as appropriate preconditioners would be beyond the scope of this thesis. Additionally, the subspaces spanned by ur columns of Z typically lead to better approximate residual norm-minimizing shifts compared to Krylov subspaces. This was observed in several experiments even if $u < k$. Moreover, using Krylov subspaces requires matrix multiplication with A to construct the restriction $\tilde{A}$ in contrast to the spaces w.r.t. block columns of Z, where this is by the Corollaries 3.9, 4.6 not necessary. There are also other reasons that prevent a successful application of these preconditioned multi-shift Krylov methods. For $E \neq I_n$, the shift-invariance of Krylov subspaces is lost since it holds in general $\mathcal{K}_k(A,\ W) \neq \mathcal{K}_k(A + \alpha E,\ W)$. In [226], a shift-and-invert preconditioner $P \approx A + \hat{\alpha} E$ is used on top of a multi-shift Krylov subspace method such that the shift-invariance holds for the coefficient matrix of the preconditioned system. For this the knowledge of a seed shift $\hat{\alpha}$ is required for which a reasonable choice is not known for our application. The numerical costs for applying P, i.e., solving linear systems defined by P, in every iteration step in every call of the employed Krylov subspace method will easily accumulate such that this approach will become expensive. Alternatively, one can, similar to (5.14), formally use the restriction $\tilde{E}_k = Q_k^T E Q_k$ to obtain a reduced objective function even though this is not justified theoretically by the shift-invariance. Moreover, the preconditioned multi-shift Krylov

methods often require that the shifts α in (5.15) are known in advance which is not possible in our situation since we want to solve the optimization problem (5.5) iteratively by an optimization algorithm.

Restrictions of the Optimization Variables and Initial Guesses In (5.4), we introduced the constraints $\nu \in \mathbb{R}_-$ and $\xi \in \mathbb{R}$ to ensure that the obtained local minimizer $\alpha = \nu + \jmath\xi \in \mathbb{C}_-$. Since we only consider GCALEs defined by real matrices and always demand proper sets of shift parameters for the G-LR-ADI iteration, we can use the restriction $\xi \in \mathbb{R}_+$. We can further tighten these restrictions, e.g., by

$$\nu_- \leq \nu \leq \nu_+, \quad 0 \leq \xi \leq \xi_+. \tag{5.19}$$

This can be helpful for the used optimization method because, roughly speaking, a smaller set of values of ν, ξ has to be scanned. An obvious choice is to use information regarding the extremal eigenvalues of $(A,\ E)$ for $\nu_\pm$, ξ_+, i.e., $\nu_- \leq \min \mathrm{Re}\,(\lambda)$, $\nu_+ \geq \max \mathrm{Re}\,(\lambda)$, and $\xi_+ \leq |\max \mathrm{Im}\,(\lambda)|$ for $\lambda \in \Lambda(A,\ E)$. The minimal and maximal real and imaginary parts can be obtained approximately by iterative eigenvalue algorithms or via estimates read off from Ritz values as for the approximate Wachspress shifts. In [2], a harmonic Arnoldi process is used to obtain approximations of the smallest eigenvalues in magnitude. A straightforward and easy to compute simplification is $\nu_- = \xi_+ = \rho(A,\ E) = \max|\lambda|$ which can be estimated easily by an Arnoldi process. For the approximate residual norm-minimizing shifts employing the above projection approach, we propose to use the eigenvalues of $(\tilde{A}_j,\ \tilde{E}_j)$ to read off the bounds $\nu_\pm, \xi_+$.

Some optimization algorithms require an initial guess of the sought local minimizer to start. Here, we use the first value returned by the heuristic shift approach on the basis of $\Lambda(\tilde{A}_j,\ \tilde{E}_j)$.

Alternative Optimization Approaches The `eigopt` software proposed in [173, 174] contains a global eigenvalue optimization method for computing global minimizers or maximizers of the eigenvalues of a parameter dependent matrix $M(\boldsymbol{s}) \in \mathbb{C}^{n\times n}$, $\boldsymbol{s} = (s_1, \ldots, s_d)$. It restricts to boxed domains $s_{i,-} \leq s_i \leq s_{i,+}$, $i = 1, \ldots, d$, similar to (5.19). It can, therefore, also be applied to find a global minimizer of $\lambda_{\max}(\mathcal{F}(\nu, \xi))$ in (5.9) for solving the present minimization problem (5.5) or its reduced counterpart (5.14). It requires, however, a lower bound for $\lambda_{\min}[\nabla^2 \lambda_{\max}(\mathcal{F}(\nu, \xi))]$. Since a reliable bound is currently not available for (5.9), we do not consider this approach further. First preliminary tests, where $\lambda_{\min}[\nabla^2 \lambda_{\max}(\mathcal{F}(\nu, \xi))]$ was computed explicitly, showed that especially in the later iteration steps of the G-LR-ADI iteration, `eigopt` seems to have difficulties in finding the minimizer.

The `chebfun` and `chebfun2` packages [79] can also be used to deal with (5.4). For uni- or bivariate functions f, `chebfun` and `chebfun2` construct approximations of f by expansions in uni- and bivariate Chebychev polynomials, respectively. The commands `min` and `min2` can then be used to find minima. This can be done in MATLAB, for instance, by the following commands:

```
cf=chebfun2(@(nu,xi) norm(W-2*nu*E*((A+(nu+1i*xi)E)\W),2),...
              [nu$_-$,nu$_+$,0, xi_$+$],'vectorize','on');
[fmin,p]=min2(cf); nu=p(1); xi=p(2);
```

Although technically not designed as optimization routine, `chebfun` and `chebfun2` seemed to be very successful in finding the global minimum of f. However, the construction of the Chebychev polynomials was significantly more expensive compared to the other approaches, even for the projected objective function (5.14).

5.3.3. Numerical Experiments

We are now going to evaluate and compare the performance of the presented shift generation strategies. As usual, the G-LR-ADI iteration (Algorithm 4.3) is carried out until $\|\mathcal{L}\| \leq \tau \|F\|^2$ with $0 < \tau \ll 1$ is achieved or a maximum allowed number $j^{\max}$ of iterations is reached. We use the test examples *FDM*, *rail79k*, *ifiss66k*, *chain*, and *bips*. For *FDM*, $n_0 = 50$ and $n_0 = 350$ grid points lead to dimensions $n = 2500$ and $n = 122500$, respectively. The functions defining the convective and reactive terms are $f_1 = 10^2\xi_1$, $f_2 = 10^3\xi_2$, and $f_3 = 0$, similar to the settings in [184, Example 1].

The heuristic shift strategy and its settings are denoted by heur(J, k_+, k_-). Likewise, Wachs(ϵ, k_+, k_-) stands for the approximate Wachspress shifts obtained from k_+, k_- Ritz values and a tolerance ϵ which results in a number of J shifts parameters. For these two approaches, the initial vector for the Arnoldi processes is $F\mathbf{1}_m$. For the *rail79k* example having $A = A^T$, $E = E^T \succ 0$, we do not use a less expensive Lanczos process in the inner product induced by E [6] because this led to worse approximations of the eigenvalues of small magnitude, resulting in heuristic and approximate Wachspress shifts of worse quality. As in the chapters before, these precomputed shifts are used in a cyclic manner if it occurs that the number of required G-LR-ADI iteration steps is higher than the number of the available shifts. The computation of the orthonormal bases for the $V(u)$- and W-shifts, where u denotes the horizon, is carried out using an implicit orthogonalization. The residual norm-minimizing shift strategy using the projected matrices are denoted by $V(u)$-res.min. and FOM(k)-res.min., where FOM(k) refers to the usage of the multi-shift FOM approach w.r.t. a Krylov subspace of order k. Because of the symmetry properties and $\Lambda(A,\ E) \subset \mathbb{R}_-$ for example *rail79k*, all of the self-generating shift parameter strategies are adapted such that only real shifts are generated.

Experiments with Exact and Approximate Residual Norm-Minimizing Shifts

At first we investigate the residual norm-minimizing shifts as well as their approximate versions using different subspaces for the projection. For this we use the *FDM* example with $n_0 = 50$ such that $n = 2500$. This size admits the computation of the exact residual norm-minimizing shift in a reasonable time. For solving the optimization problems w.r.t. exact (5.4) and approximate objective function (5.14), we employ the MATLAB routine `fmincon` with the interior point algorithm. Both first and second order derivatives are provided to `fmincon` by using Algorithm 5.1. This led to better results compared to

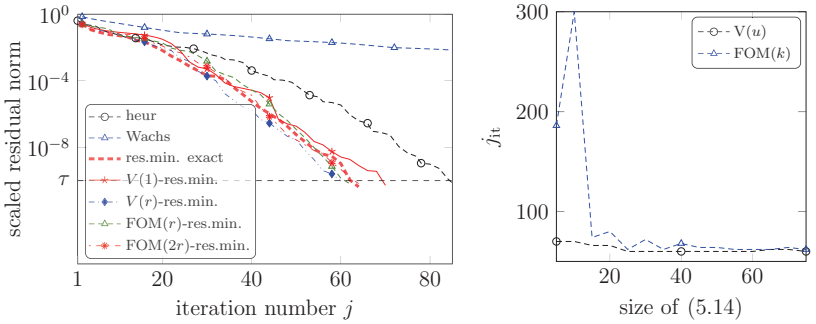


Figure 5.2.: Left: History of the normalized residual norms obtained with different versions of the residual norm-minimizing shifts. Right: number of LR-ADI iteration steps j_{it} against dimension of (5.14) for $V(u)$ and FOM(k) bases.

the first occurrence [39] of the residual norm-minimizing shifts, where the derivative free `fminsearch` routine was used. Notice that using `fmincon` without second order derivatives still led to useful results. However, providing also no first order derivatives usually leads to a failure of `fmincon`. The stopping tolerance for the optimization iteration was set to $\tau_{\text{opti}} = \frac{1}{2}\tau$. The constraints on the optimization variables and initial guesses are set as described above. For the exact residual minimizing shifts the restrictions $\tilde{A}_j$, $\tilde{E}_j$ w.r.t. the $V(1)$-shift approach were used for determining the initial guesses. For the approximate residual norm minimizing shifts the subspaces provided by the $V(u)$-shift approach and the multi-shift FOM(k) approach are used for different values u and k. For comparison, we also employ the heuristic and approximate Wachspress shift parameters which we set up as heur$(10, 20, 10)$ and Wachs$(10^{-10},\ 20,\ 20)$, respectively. The history of the scaled CALE residual norm against the iterations index j for the different shift parameters is shown in the left plot in Figure 5.2.

Apparently, the residual norm-minimizing shifts lead to a faster convergence of the LR-ADI iteration than the heuristic or approximate Wachspress shifts, where the latter ones yield an extraordinarily slow reduction of the residual norm. It is important to mention that only minor improvements are attained if more accurate or even exact spectral data is used for the quantities $a,\ b,\ \psi$. The lowest number of iteration steps is provided by the projected residual norm-minimizing shifts using the subspaces from the $V(r)$-approach and FOM(r). These subspaces yield equally many or even slightly less iteration steps than the exact residual norm-minimizing shifts. We observed that the employed optimization method for the exact approach encountered problems in computing the local minimizers in several iteration steps. These problems showed through a premature termination of the optimization algorithm before an approximate local minimum was found which was caused by too small relative changes of the optimization iterates. The optimization routine seems to have less difficulties to solve the significant smaller,

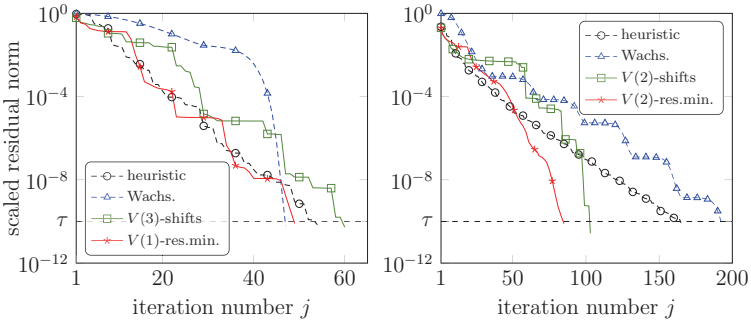


Figure 5.3.: Scaled residual norm against iteration step j of the G-LR-ADI iteration employing different shift strategies for *rail79k* (left) and *ifiss66k* (right) examples.

projected optimization problem. For this, equally good results can also be obtained if no exact Hessians are provided and also if the trust-region-reflective algorithm within `fmincon` is used. We also see that increasing the dimension of the reduced optimization problem by higher values u for the $V(u)$-approach usually yields shift parameters of better quality, i.e., less required LR-ADI iteration steps j_{it}. In the right plot in Figure 5.2 we observe that the $V(u)$-approach appears to require smaller subspace dimensions compared to FOM(k). Therefore, this is the method of choice in the remaining experiments for obtaining the projected optimization problem.

Comparison of Different Shift Parameters for the G-LR-ADI Iteration

The results for all five test examples and different shift strategies are summarized in Table 5.1. The approximate residual minimizing shifts are now computed with the trust-region-reflective algorithm within `fmincon`. We do not include exact Hessians by Algorithm 5.1 because this brought only minor performance gains of the optimization routine at the expense of higher shift generation times.

The results for these examples and the different shift strategies are summarized in Table 5.1: t_{shift} and t_{ADI} denote the times (in seconds) spent for computing the shifts and executing the G-LR-ADI iteration, respectively, and the total consumed time is t_{total}. The required iteration steps j_{it} and the final obtained residual norm $\epsilon_{j_{\text{iter}}}$ are also given. The smallest values of t_{shift}, t_{total}, and j_{it} for each example are emphasized using bold letters. We only report the shift strategies where the desired accuracy was met in in less than $j^{\max}$ iteration steps. In addition, Figure 5.3 shows the scaled residual norm against the ADI iteration number for the examples *rail79k* and *ifiss66k*.

For the heuristic and approximate Wachspress shifts it is apparent that, compared to the plain ADI computation time t_{ADI}, a significant portion t_{shift} of the total execution time t_{total} is spent in the involved Arnoldi processes. They manage to steer the

Table 5.1.: Results for the examples using different shift strategies.

Example	shift strategy	t_{shift}	t_{ADI}	t_{total}	j_{it}	$\epsilon_{j_{\text{it}}}$
FDM $\tau = 10^{-10}$ $j_{\max} = 250$	heur(10, 20, 10)	10.4	245.9	256.2	157	$7.71{\cdot}10^{-11}$
	Wachs(10^{-10}, 20, 20), $J = 31$	19.8	149.4	169.2	99	$9.94{\cdot}10^{-11}$
	$V(2)$-shifts	**0.7**	150.5	151.2	104	$9.44{\cdot}10^{-11}$
	W-shift	1.8	160.8	162.6	113	$9.59{\cdot}10^{-11}$
	$V(3)$-res.min.	7.4	118.3	**125.7**	**77**	$8.51{\cdot}10^{-11}$
rail79k $\tau = 10^{-10}$ $j_{\max} = 100$	heur(20, 40, 40)	24.9	56.8	81.7	54	$7.00{\cdot}10^{-11}$
	wachs(10^{-10}, 20, 10), $J = 47$	6.5	73.9	80.4	**47**	$6.27{\cdot}10^{-11}$
	$V(3)$-shifts	**1.2**	74.4	75.6	60	$5.31{\cdot}10^{-11}$
	$V(1)$-res.min.	2.7	57.6	**60.3**	49	$8.05{\cdot}10^{-11}$
ifiss66k $\tau = 10^{-10}$ $j_{\max} = 200$	heur(20, 30, 20)	19.2	141.6	160.8	165	$8.62{\cdot}10^{-11}$
	Wachs(10^{-10}, 20, 10), $J = 33$	10.2	163.4	173.6	193	$6.88{\cdot}10^{-11}$
	$V(2)$-shifts	**1.2**	84.9	86.1	103	$2.69{\cdot}10^{-11}$
	$V(2)$-res.min.	3.3	79.1	**82.4**	**91**	$4.03{\cdot}10^{-11}$
chain $\tau = 10^{-8}$ $j_{\max} = 400$	heur(50, 80, 80)	18.8	19.6	38.4	476	$9.98{\cdot}10^{-9}$
	Wachs(10^{-10}, 20, 10), $J = 160$	1.5	8.6	10.1	370	$8.96{\cdot}10^{-9}$
	$V(1)$-shifts	**0.4**	6.5	**6.9**	167	$8.95{\cdot}10^{-9}$
	$V(1)$-res.min.	4.3	12.2	16.5	**153**	$9.57{\cdot}10^{-9}$
bips $\tau = 10^{-8}$ $j_{\max} = 400$	heur(40, 50, 70)	9.3	23.9	33.1	378	$5.94{\cdot}10^{-9}$
	heur(60, 80, 80)	8.2	12.3	20.5	194	$9.36{\cdot}10^{-9}$
	Wachs(10^{-8}, 30, 30), $J = 216$	2.9	16.7	19.6	187	$9.65{\cdot}10^{-9}$
	$V(2)$-shifts	**0.1**	5.6	**5.7**	85	$7.64{\cdot}10^{-9}$
	W-shifts	1.7	7.2	8.9	96	$9.88{\cdot}10^{-9}$
	V^{aug}-shifts	**0.1**	5.7	5.9	85	$9.81{\cdot}10^{-9}$
	$V(2)$-res.min.	2.9	6.0	8.9	**81**	$8.47{\cdot}10^{-9}$

iteration to the desired accuracy, but for each example there is at least one other shift strategy which required less iteration steps. The number of used Arnoldi steps k_+, k_- can substantially influence the quality of the heuristic shifts as it is seen in the *bips* example. There, two settings are used: the first one uses exactly the values J, k_+, k_- as in the original SLRCF-ADI paper [98, Section V, Table V] while the second one was chosen through extensive trial and error optimization. The difference in both execution time as well as iteration steps is distinct. As expected, the approximate Wachspress shifts lead to a very good performance both in terms of execution time and required iterations for the symmetric example *rail79k*. Their typical residual curves can be seen in Figure 5.3 (left plot). They loose this superiority for the examples where complex spectra with large imaginary parts are encountered. There, they can often not compete with the heuristic, and in several accounts also not with the other shift strategies which is evident in the right plot of Figure 5.3 for the *ifiss66k* example. In additional tests the

Wachspress shifts seemed to be less sensitive w.r.t. the values k_+, k_- than the heuristic shifts.

Now, we move on to the novel self-generating shifts proposed in Section 5.3.1-5.3.2. The $V(u)$- and W-shifts in all examples require a very small construction time t_{shift} which is in most cases a negligible fraction of t_{total}. However, except for *FDM* and *bips*, the W-shifts did not achieve the required accuracy before $j^{\max}$ ADI iterations. The $V(u)$-shifts outperform the heuristic and approximate Wachspress shifts for all but except the *FDM* and *rail79k* examples. Especially for the *chain* and *bips* examples they lead to a drastically reduced number of required iterations. In fact, we never experienced a faster ADI convergence for the *bips* system. There, the performance of the V^{aug}-shifts is comparable to the V-shifts. The W-shifts achieved convergence while the W^{aug}-shifts did not.

The residual norm-minimizing shifts using the basis of the $V(u)$-shifts lead to a further speed up of the G-LR-ADI iteration for most examples. For *FDM* and *ifiss66k* this led also to a lower computation time t_{total}. Due to the inherent optimization problems, the generation times t_{shift} are noticeable larger compared to the $V(u)$-shifts. Their generation is still less costly than for the heuristic and approximate Wachspress shifts which leads to even smaller computation times for the symmetric *rail79k* examples. In the left plot in Figure 5.3 we see that they lead to a more even decrease of the residual norm, compared to the Wachspress shifts which achieve a fast rate of convergence after a comparable slow phase in the beginning. In some cases, convergence problems in the employed optimization algorithm were encountered in the residual norm-minimizing shift approach. This can be seen from the short stagnation phases in the left plot Figure 5.3 for the *rail* example. To conclude, the $V(u)$-shifts appear to be a very promising approach, especially for Lyapunov equations with nonsymmetric coefficient matrices, where the spectrum contains complex eigenvalues. For problems with real spectra the approximate Wachspress shift are usually also a reasonable choice. The extension to the $V(u)$-residual norm-minimizing shifts appears to lead to further improvements in most cases.

Another big advantage of both approaches is, although not reflected in the timings and iteration counts, that they can be applied completely automatically in the sense that they can be implemented without the user having to take care of selecting ADI shifts at all. The exception is the horizon parameter u for which $u = 2$ appears to be sufficient for most situations. In [39], the setting $u = 1$ is used which already delivered significant speed ups compared to existing shift strategies. The $V(u)$-res.min. shifts rely on a successful numerical solution of the projected optimization problem, where problems occurred for some examples. The success of the inherent optimization routine seemed, in some instances, also to critically depend on the chosen constraints of the optimization variables and the initial guesses. Further enhancement of this optimization phase are mandatory for making the $V(u)$-res.min. shift strategy more robust.

Comparison with Other Low-Rank Methods for GCALEs

Having improved and automated the shift parameter issue for the G-LR-ADI iteration, we now run some comparative studies with two other low-rank methods for GCALEs.

For this we employ the Extended and Rational Krylov Subspace Methods (EKSM and RKSM) [209, 82] with the same stopping criterion based on the scaled residual norm as before. A basic implementation of RKSM can be found in Algorithm A.1 in the appendix. All of the examples used before have $r > 1$ and, therefore, block versions of EKSM and RKSM are employed. RKSM demands, similar to the ADI iteration, a number of shift parameters $\xi_1, \ldots, \xi_k \in \mathbb{C}_+$ and the adaptive strategy proposed in [83] is used. Complex shift parameters are dealt with along the lines of [195] (see Line 8 in Algorithm A.1).

Since Krylov subspaces are usually defined by a single matrix and the initial vector (or block column), some remarks are in order on how to deal with the matrix $E \neq I_n$ for generalized problems. Because E is nonsingular, Krylov subspace methods can be implicitly build from $\hat{A} := E^{-1}A$ and $\hat{B} := E^{-1}B$. For EKSM also matrix vector products with $\hat{A}^{-1} = A^{-1}E$ are required such that in each iteration step two linear systems have to be solved: one with A and one with E. The linear systems required in each step of RKSM can, in complete analogy to Algorithm 3.2, be expressed by $(A - \xi_i E)^{-1}E$. Alternatively, one can implicitly work with $\hat{A} = L^{-1}AL^{-T}$, $\hat{B} = L^{-T}B$ if $0 \prec E = LL^T$, where L is the lower, triangular Cholesky factor of E which has to be constructed only once in the beginning. Since in this case EKSM also works with $\hat{A}^{-1} = L^T A^{-1} L$, it requires additional linear solves as well as matrix vector products with L and L^T. The shifted and inverted matrices of RKSM are given by $L(A+\xi_i E)^{-1}L^T$ see, e.g., [209, 83]. It is noteworthy that the G-LR-ADI iteration (Algorithm 4.3) does not require any solves with E or its Cholesky factors. For EKSM it can also be advantageous to compute and store a factorization of A before the actual iteration and reuse its factors in each iteration step. For the examples *chain* and *bips*, modifications of EKSM and RKSM are used which employ similar structure exploiting techniques as in the SO-LR-ADI and SLRCF-ADI iteration.

For example *FDM*, we will also take the opportunity to compare to the original LR-ADI iteration (3.10) from Section 3.2 without any of the improvements developed though the Chapters 3–5. That is, no exploitation of the low-rank structure (Theorem 6.1) of $\mathcal{L}_j$ and therefore computing $\|\mathcal{L}_j\|$ via a Lanczos process, no handling of complex shift parameters (cf. Section 4.1), and using only the heuristic shift parameter approach.

The results are summarized in Table 5.2 which gives the column dimension d of the computed low-rank solution factor. For EKSM and RKSM, this is also the dimension of the used extended or rational Krylov subspace. Furthermore, computational timings $t_{\text{sol.}}$, $t_{\text{orth.}}$, t_{small}, t_{shift}, and $t_{\text{res.}}$ for different stages of the algorithms corresponding to linear system solves, orthogonalization, solving the small, projected GCALE, computing the residual norm, and generating shift parameters, respectively, are given as well the total computation time t_{total}. Notice that since we do not include further computational stages, for instance, initializing data structures, augmenting the low-rank solution factors, or constructing certain auxiliary quantities, the actual total computation time t_{total} can be slightly larger than the sum of $t_{\text{sol.}}$, $t_{\text{orth.}}$, t_{small}, t_{shift}, and $t_{\text{res.}}$. The smallest value of t_{total} is written in bold letters. Due to the nature of particular algorithms, some of the single timings are not present. Since the G-LR-ADI iteration uses orthogonalization routines exclusively for the shift generation and not as part of the intrinsic iteration, the

corresponding value t_{shift} includes any possible orthogonalization timings. In addition, Figure 5.4 shows for two examples the progress of the scaled Lyapunov residual norms of all three low-rank solvers as the column dimension of the low-rank solution factor grows.

Table 5.2.: Comparison between G-LR-ADI iteration, EKSM and RKSM.

Example	algorithm	dim.	$t_{\text{sol.}}$	$t_{\text{orth.}}$	t_{small}	t_{shift}	$t_{\text{res.}}$	t_{total}
FDM $\tau = 10^{-10}$	old LR-ADI, heur.	770	411.1	–	–	10.2	213.8	636.8
	LR-ADI, $V(3)$-res.min.	385	112.2	–	–	7.4	0.7	**125.7**
	EKSM	930	127.6	95.0	134.9	–	0.02	363.6
	RKSM	**355**	101.9	42.5	10.9	14.1	8.1	178.7
rail79k $\tau = 10^{-10}$	G-LR-ADI, $V(1)$-res.min.	343	52.9	–	–	2.7	0.6	**60.3**
	EKSM	980	83.6	51.4	112.3	–	0.02	251.6
	RKSM	**294**	45.7	13.8	17.6	5.6	5.4	89.2
ifiss66k $\tau = 10^{-10}$	G-LR-ADI, $V(2)$-res.min.	450	76.2	–	–	3.3	0.4	**82.4**
	EKSM	1030	110.5	49.2	188.3	–	0.03	352.1
	RKSM	**400**	66.4	24.6	20.3	17.1	5.1	134.1
chain $\tau = 10^{-8}$	SO-LR-ADI, $V(1)$-shifts	835	5.0	–	–	0.4	0.1	**6.9**
	EKSM	no convergence						
	RKSM	**775**	7.1	58.2	69.7	85.3	12.2	233.4
bips $\tau = 10^{-8}$	SLRCF-ADI, $V(1)$-shifts	340	5.5	–	–	0.1	0.1	**5.7**
	EKSM	no convergence						
	RKSM	**320**	4.9	1.1	5.2	11.8	0.2	23.3

In all examples, the G-LR-ADI iteration achieves the desired accuracy in the smallest amount of time. RKSM requires slightly smaller dimensions of the low-rank solution factors which directly translates to slightly less iteration steps. Because of the inherent orthogonalization processes and the solution of the small, projected GCALEs, the computation times are noticeable larger compared to the G-LR-ADI iteration. The adaptive shift generation within RKSM appears to be also more costly than the used shift strategies for the G-LR-ADI iteration. In contrast, EKSM cannot compete with either RKSM or the G-LR-ADI iteration as it exhibits a much slower residual norm reduction leading to significantly higher times $t_{\text{orth.}}$ and t_{small}. EKSM does fail to deliver an approximate solution for the examples *chain* and *bips*. It should be noted that for EKSM and RKSM, solving the small GCALE only at every couple of steps, e.g., every 5th step, is a very obvious procedure for reducing t_{small}. A further development using a minimal residual type extraction [166] instead of the Galerkin approach points towards improvements of the convergence speed of EKSM and RKSM. It is, however, also mentioned in [166] that this minimal residual extraction becomes expensive for $r > 1$. Employing tangential directions [84] can help to reduce the overall computational effort, especially regarding the orthogonalization, in EKSM and RKSM. There, the basis of the subspace is, e.g. in RKSM, augmented by $s = (I - A/\xi)^{-1}Ud$ for $U \in \mathbb{C}^{n\times m}$ and an appropriately chosen

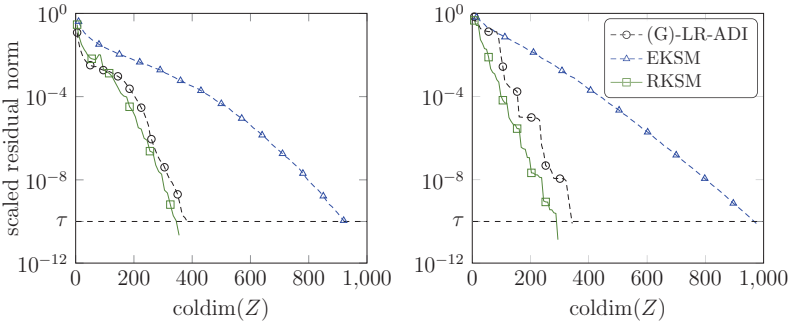


Figure 5.4.: Scaled residual norm against the column dimension of the low-rank solution factors generated by different low-rank methods for the *FDM* (left) and *rail79k* (right) examples.

tangential direction vector $d \in \mathbb{C}^m$. A first concept to introduce tangential directions into the G-LR-ADI iteration can be found in [236]. For this, however, there is no efficient generation strategy for the tangential directions d available so far. It is noteworthy that except *rail79k*, none of the examples satisfies the condition

$$E^{-1}A + A^T E^{-T} \prec 0 \quad \Leftrightarrow \quad AE^T + EA^T \prec 0 \tag{5.20}$$

which ensures that the coefficient matrices defining the projected GCALE (cf. the description in Section 2.3.4) are asymptotically stable. Apparently EKSM or RKSM were not negatively affected by this violation in the majority of experiments. Only for the *bips* example, EKSM encountered severe convergence problems caused by a violation of (5.20). This resulted in a frequent construction of unstable coefficient matrices defining the projected GCALEs and, consequently, to defective results. Such unstable projected matrices also appeared with significantly less occurrences in RKSM and, for *FDM*, for both RKSM and EKSM. In these cases, however, the algorithms seem to recover from this and still produce meaningful results. For *chain*, the convergence speed of EKSM was comparably very slow as it achieved a scaled residual norm of $\varepsilon_{j_{\text{it}}} \approx 3.6 \cdot 10^{-3}$ in 150 iterations and a subspace dimension of 1500 after which we terminated the routine.

To conclude, for the used examples the G-LR-ADI iteration is definitely competitive to RKSM and EKSM. The good performance of the G-LR-ADI iteration is to a substantial amount also achieved by the efficiency enhancements introduced prior in Sections 3.2 and 4.1. Without those modifications, any superiority compared to EKSM, RKSM would be lost, as we see from the large timings of the old version of the LR-ADI iteration for example *FDM*. The newly developed self-generating shift parameters further improve the G-LR-ADI iteration. It is important to keep in mind that both RKSM and the G-LR-ADI iteration critically depend on good shift parameters, and their absence can yield severe losses w.r.t. the performance of both methods.

5.4. Shift Parameters for the Sylvester ADI Iteration

In this section, we discuss shift parameters strategies for the G-fADI iteration for GCASEs. We begin again by reviewing some selected existing shift strategies. After that we generalize the projection based and residual norm-minimizing shifts discussed in Section 5.3 in a straightforward way. Numerical experiments illustrate the performance of the G-fADI iteration under the influence of different shift generation approaches.

5.4.1. Existing Shift Strategies

Similar to the G-LR-ADI iteration for Lyapunov equations, the error reduction in the ADI iteration (3.29) and its low-rank version, the G-fADI iteration (Algorithm 3.4), for generalized Sylvester equations depends by Lemma 3.12 critically on the iteration matrices

$$\begin{aligned} \mathcal{A}^{(j)} &:= \prod_{k=1}^{j} \mathcal{C}_A^{(k)}, \quad \mathcal{C}_A^{(k)} := \mathcal{C}(A, E, -\beta_k,\ \alpha_k) = (A - \beta_k E)^{-1}(A - \alpha_k E), \\ \mathcal{B}^{(j)} &:= \prod_{k=1}^{j} \mathcal{C}_B^{(k)}, \quad \mathcal{C}_B^{(k)} := \mathcal{C}(B, C, -\alpha_k,\ \beta_k) = (B - \alpha_k C)^{-1}(B - \beta_k C). \end{aligned} \tag{5.21}$$

In [230, 202, 162, 43], approaches are presented to obtain shifts for j iteration steps of the G-fADI iteration such that the spectral radii of $\mathcal{A}^{(j)},\ \mathcal{B}^{(j)}$ are as small as possible. This leads to the rational optimization problem

$$\min_{\alpha_j, \beta_j \in \mathbb{C}} \left(\max_{\substack{1 \le \ell \le n \\ 1 \le k \le r}} \prod_{j=1}^{J} \left| \frac{(\lambda_\ell - \alpha_j)(\mu_k - \beta_j)}{(\lambda_\ell - \beta_j)(\mu_k - \alpha_j)} \right| \right), \quad \lambda_\ell \in \Lambda(A,\ E),\ \mu_k \in \Lambda(B,\ C), \tag{5.22}$$

which is also referred to as two-variable ADI parameter problem [230, 233] and is harder to solve than the optimization problem (5.2) for Lyapunov equations. Using (5.22) as starting point for finding shift parameters has the same conceptual drawback as (5.2), i.e., the low-rank structure of the inhomogeneity FG^T is not embraced. In the following, we review existing generalizations of the Wachspress and heuristic shifts for the G-fADI iteration which are based on (5.22). Afterwards, generalizations of the self-generating shifts presented in Section 5.3 are proposed.

Optimal Shifts for the Two-Variable ADI Parameters Problem

Analytic solutions for solving (5.22) are proposed, e.g., in [230],[233, Chapter 2 & 4] and are based on spectral alignment and, as in the Lyapunov case, elliptic integrals. They require knowledge of the smallest and largest real parts $a := \min_i \operatorname{Re}(\lambda_i)$, $b := \max_i \operatorname{Re}(\lambda_i)$, $c := \min_i \operatorname{Re}(\mu_i)$, $d := \max_i \operatorname{Re}(\mu_i)$ and the opening angles (cf. 2.27) $\psi_A := \max_i \arctan\left|\frac{\operatorname{Im}(\lambda_i)}{\operatorname{Re}(\lambda_i)}\right|$, $\psi_B := \max_i \arctan\left|\frac{\operatorname{Im}(\mu_i)}{\operatorname{Re}(\mu_i)}\right|$ for $\lambda_i \in \Lambda(A,\ E)$ and $\mu_i \in$

$\Lambda(B,\ C)$. An implementation of this shift generation strategy is given in the `parsyl`[1] routine provided in [233]. If the spectra $\Lambda(A,\ E)$, $\Lambda(B,\ C)$ are contained in real, disjoint intervals $[a,\ b]$, $[c,\ d]$, another similar approach for generating an equal number J of α- and β-shifts is given in [202, Algorithm 2.1].

As in the Lyapunov case, one might use Arnoldi or Lanczos processes to obtain approximations to $a,\ b,\ c,\ d,\ \psi_A,\ \psi_B$ in the large-scale case for both approaches. We propose to approximate $\Lambda(A,\ E)$ by a set consisting of k_+^A Ritz and k_-^A inverse Ritz values w.r.t. $E^{-1}A$ and $A^{-1}E$. Likewise, $\Lambda(F,\ G)$ is replaced by k_+^B Ritz and k_-^B inverse Ritz values w.r.t. $C^{-1}B$ and $B^{-1}C$. Approximations to the extremal eigenvalues and the spectral angles of $\Lambda(A,\ E)$ and $\Lambda(B,\ C)$ can then be read of easily. However, as for the approximate Wachspress shifts in the GCALE case, the obtained shifts can be sensitive w.r.t. the quality of the approximations of the extremal eigenvalues. This was numerically investigated in [202, Section 2.2.2] for the optimal real shift parameters.

Heuristic Shifts

In [162, 43], a heuristic approach is proposed which generalizes the Penzl shifts (Section 5.2.2) to the solution of Sylvester equations. The spectra $\Lambda(A,\ E)$, $\Lambda(B,\ C)$ are approximated in the same way as for the optimal shifts above. With these sets of Ritz values, one solves (3.38) in an approximate sense to get $J \leq k_+^A + k_-^A$ α- and $L \leq k_+^B + k_-^B$ β-shifts. A detailed implementation can be found in [162, Algorithm 3.1], [43, Algorithm 2]. In [32], just the $k_+^A + k_-^A$ and $k_+^B + k_-^B$ Ritz values are used as shifts which worked sufficiently well. This heuristic approach suffers from the same disadvantages as the heuristic approach for Lyapunov equations discussed in Section 5.2.2: there is no known rule on how to select the predefined numbers J, L, k_+^A, k_-^A, k_+^B, and k_-^B, the quality of the Ritz values (and, hence, of the shifts) depends on the performance of the Arnoldi processes which also introduces additional costs due to the required linear solves. Moreover, there is no known strategy for choosing their initial vectors suitably.

Other Shifts

An overview over several other approaches for generating shifts for the Sylvester ADI iteration, for instance, generalizations of the Leja point based shifts, can be found in [202]. For a generalized version of the iteration (3.28), specialized shift strategies can be found in [158]. Shifts for Sylvester equations occurring in image restoration are proposed in [64].

For symmetric Sylvester equations with $E,\ -C,\ -A,\ -B$ spd, a generalization of IRKA for GCASEs (symmetric Sylvester IRKA, (Sy)^{2}IRKA) is given in [21, Algorithm 3],[22]. Similar to the IRKA approach for GCALEs, the obtained approximate solutions again satisfy an optimality condition w.r.t. their residual in a certain norm. The shifts obtained from (Sy)^{2}IRKA can also be used within the G-fADI iteration leading to equivalent approximate solutions as discussed in [96]. (Sy)^{2}IRKA can be easily

[1] Available at `http://extras.springer.com/2013/978-1-4614-5121-1`.

modified to a general Sylvester IRKA (SyIRKA) to handle GCASEs with nonsymmetric coefficients. This strategy has the same drawbacks as the similar one in GCALE case, especially the high computational cost of SyIRKA makes it a computationally less feasible as it is reported in the numerical experiments with SyIRKA in [39]. As in the GCALE case, they also appear to be highly susceptible regarding the starting data for SyIRKA. The SyIRKA shifts are rather theoretically motivated due to the interesting theoretical properties [96, 21, 22] of the IRKA shifts.

5.4.2. Self-Generating Shifts

In the upcoming subsections, we present generalizations of the $V(u)$-, W-, and residual norm-minimizing shifts from Section 5.3 for the use in the G-fADI iteration. In line with the considerations in Section 4.2, we assume that the used sets of shift parameters are proper (Definition 4.1) and are ordered in the introduced form (cases 1–3)

Shifts Obtained via Projections with ADI Iterates

In [39], the $V(u)$- and W-shifts for GCALEs in Section 5.3.1 are in a straightforward manner generalized to GCASEs. Assume we completed step j of Algorithm 3.4 having the iterates V_j, S_j computed, and look for shift parameters for the subsequent iteration steps. Let Q_j, U_j contain orthonormal columns which span the bases of $\operatorname{span}\{V_j\}$ and, respectively, $\operatorname{span}\{S_j\}$. Then the α- and β-shifts for the next r iteration steps are taken as the eigenvalues of the restrictions corresponding to Ritz Galerkin projections of $(A,\ E)$ and $(B,\ C)$ w.r.t. $\operatorname{span}\{V_j\}$ and $\operatorname{span}\{S_j\}$:

$$\{\alpha_{j+1},\dots,\alpha_{j+r}\} = \Lambda(\tilde{A}_j := Q_j^H A Q_j,\ \tilde{E}_j := Q_j^H E Q_j),$$
$$\{\beta_{j+1},\dots,\beta_{j+r}\} = \Lambda(\tilde{B}_j := U_j^H B U_j,\ \tilde{C}_j := U_j^H C U_j).$$

In analogy to the V-shifts, this approach is referred to as V/S-shifts, [39]. As in the GCALE case it is reasonable to work with orthonormal bases of $[\operatorname{Re}(V_j),\ \operatorname{Im}(V_j)]$ and $[\operatorname{Re}(S_j),\ \operatorname{Im}(S_j)]$ when $V_j,\ S_j$ are complex iterates. Due to the complicated and bulky formula derived in Theorem 4.7, Corollary 4.8 w.r.t. the handling of complex shift parameters, we keep to the complex G-fADI iteration (Algorithm 3.4) as this considerably eases and shortens the representations.

As for the $V(u)$-shifts, we can also incorporate the block columns of the low-rank solution factors $Z,\ Y$ w.r.t. $u > 1$ previous iteration steps which directly leads to $V/S(u)$-shifts. To construct the restrictions $\tilde{A}_j,\ \tilde{B}_j$ more efficiently we can, along the lines of (5.3), modify the relation (3.46) or Corollary 3.16:

$$A\hat{Z}_{j,u} = W_j G_{j,u}^T + E\hat{Z}_{j,u}\boldsymbol{\sigma}_{j,u}^{\alpha}, \quad B^T\hat{Y}_{j,u} = T_j G_{j,u}^T + C^T\hat{Y}_{j,u}\boldsymbol{\sigma}_{j,u}^{\beta}, \tag{5.23}$$

where $\hat{Z}_{j,u} := Z(:,(j-u)r+1:jr)$, $\hat{Y}_{j,u} := Y(:,(j-u)r+1:jr)$, and $\boldsymbol{\sigma}_{j,u}^{\alpha}$, $\boldsymbol{\sigma}_{j,u}^{\beta} \in \mathbb{C}^{ru\times ru}$, $G_{j,u}$ are the appropriate blocks of $\boldsymbol{\sigma}_j^{\alpha}$, $\boldsymbol{\sigma}_j^{\beta}$, G_j.

Incorporating the relations established in Theorem 4.7, Corollary 4.8 will change $\hat{Z}_{j,u}$, $\hat{Y}_{j,u}$ $\boldsymbol{\sigma}_{j,u}^{\alpha}$, and $\boldsymbol{\sigma}_{j,u}^{\beta}$ to real matrices. The formula (5.23) can be used to construct the

restrictions w.r.t. orthonormal bases of $\operatorname{span}(\hat{Z}_{j,u})$ and $\operatorname{span}(\hat{Y}_{j,u})$ without additional matrix vector products with A, B. For the restrictions of B, C, the formulas (3.46), (5.23) suggest to actually work with B^T, C^T instead which does not represent any difficulties since both are real matrices. Some matrix vector products with E, C^T have to be stored or recomputed, however. Similar to the $V(u)$-shifts, we propose to select a smaller number of, e.g., r, α- and β-shifts from the entire ru eigenvalues by using the previously mentioned heuristic strategy.

In [39], also an intuitive generalization of the W-shifts, called the W/T-shifts, is proposed by using orthonormal bases for $\operatorname{span}\{W_j\}$, $\operatorname{span}\{T_j\}$. Again, the restrictions for these W/T-shifts cannot be built without multiplications with A, B.

For both, $V/S(u)$- and W/T-shifts, initial α- and β-shifts can be constructed by using orthonormal bases of F and G, respectively. In our upcoming experiments the orthogonalization process is carried out exactly as in the GCALE case. It can happen in both shift generation strategies that the number of obtained α- and β-shifts is different, and, hence, new α- and β-shifts do not need to be calculated at the same time, but we restrict to this situation here for simplification. Notice that one should ensure that the new shifts satisfy $\alpha \neq \beta$.

Residual Norm-Minimizing Shifts

Motivated by the Lyapunov residual norm-minimizing shifts in Section 5.3.2, one can derive a similar framework for Sylvester equations. At step j of the G-fADI iteration, the (spectral or Frobenius) norm of the Sylvester residual $\mathcal{S}_j$ is by (3.40), (3.45) given as

$$\|\mathcal{S}_j\| = \|W_j T_j^H\| = \sqrt{\lambda_{\max}((T_j^H T_j)(W_j^H W_j))},$$

where the residual factors are

$$\begin{aligned} W_j &= W_{j-1} + (\beta_j - \alpha_j) E V_j = W_{j-1} + (\beta_j - \alpha_j) E \left((A - \beta_j E)^{-1} W_{j-1}\right), \\ T_j &= T_{j-1} - \overline{(\beta_j - \alpha_j)} C^T S_j = T_{j-1} - \overline{(\beta_j - \alpha_j)} C^T \left((B - \alpha_j C)^{-H} T_{j-1}\right). \end{aligned}$$

Since $W_{j-1},\ T_{j-1}$ are known at the beginning of step j, the only unknowns above are the shift parameters α_j, β_j and we may see $\|\mathcal{S}_j\|^2$ as a function

$$h_j(\alpha_j, \beta_j) := \lambda_{\max}((W_j^H W_j)(T_j^H T_j)) \tag{5.24}$$

in two complex variables. By decomposing α, β as $\alpha = \nu + \jmath\xi$, $\beta = \delta + \jmath\eta$, we can also consider h_j as function in four real variables $h_j(\alpha_j, \beta_j) = h_j(\nu_j,\ \delta_j,\ \xi_j,\ \eta_j)$. Finding the optimal shifts w.r.t. to (5.24) motivates the optimization problem

$$[\nu_j,\ \delta_j,\ \xi_j,\ \eta_j] = \underset{\substack{\nu\in\mathbb{R}, \xi\in\mathbb{R}_+ \\ \delta\in\mathbb{R}, \eta\in\mathbb{R}_+}}{\operatorname{argmin}} h_j(\nu_j,\ \delta_j,\ \xi_j,\ \eta_j). \tag{5.25}$$

The remarks from the GCALE case regarding global and local minimizers of the objective function fit here as well and we, therefore, restrict to the detection of local

minima only. Solving (5.25) exactly is again very expensive since each function evaluation of h_j in an optimization routine alone requires to solve two shifted linear systems with r right hand sides. In the following, similar strategies compared to the residual norm-minimizing shifts in the GCALE case are introduced to help with the formidable eigenvalue optimization problem.

Derivatives of the Objective Function The same techniques regarding the derivatives of eigenvalues of parameter dependent matrices [149] which we employed for the residual norm-minimizing shift in the GCALE case can be applied here to (5.24). Neglecting for the moment the iteration index j, the next theorem expresses ∇h in terms of the involved matrices and the shift α, β.

Theorem 5.2 (Gradient of the objective function (5.24)**):**

With A, E, B, C defining a GCASE, W, T the low-rank factors of the residual of the G-fADI iteration, two complex numbers $\alpha = \nu + \jmath\xi$, $\beta = \delta + \jmath\eta$ with $\alpha \neq \beta$, $\alpha, \beta \notin \Lambda(A, E) \cup \Lambda(B, C)$, and let

$$\begin{aligned} W_+ &:= W + \gamma E W_A, \quad W_A := E L_A^{-1} W, \; W_{A^2,+} := E L_A^{-1} W_+, \quad L_A := A - \beta E, \\ T_+ &:= T - \overline{\gamma} C^T T_B, \quad T_B := C^T L_B^{-H} T, \; T_{B^2,+} := C^T L_B^{-H} T_+, \quad L_B := B - \alpha C, \\ \gamma &:= \beta - \alpha = \delta - \nu + \jmath(\eta - \xi). \end{aligned}$$ ◊

Assume that $(W_+^H W_+)(T_+^H T_+)$ has r simple eigenvalues for all α, β and let x_1, y_1 with $y_1^H x_1 = 1$ be the right and left eigenvector corresponding to the largest eigenvalue.

The gradient of $h = h(\nu, \delta, \xi, \eta)$ is given by

$$\nabla h = 2 \begin{bmatrix} -\operatorname{Re}\left(y_1^H W_A^H W_+ T_+^H T_+ x_1\right) \\ \operatorname{Re}\left(y_1^H W_{A^2,+}^H W_+ T_+^H T_+ x_1\right) \\ -\operatorname{Im}\left(y_1^H W_A^H W_+ T_+^H T_+ x_1\right) \\ \operatorname{Im}\left(y_1^H W_{A^2,+}^H W_+ T_+^H T_+ x_1\right) \end{bmatrix} + 2 \begin{bmatrix} \operatorname{Re}\left(y_1^H W_+^H W_+ T_{B^2,+}^H T_+ x_1\right) \\ -\operatorname{Re}\left(y_1^H W_+^H W_+ T_B^H T_+ x_1\right) \\ -\operatorname{Im}\left(y_1^H W_+^H W_+ T_{B^2,+}^H T_+ x_1\right) \\ \operatorname{Im}\left(y_1^H W_+^H W_+ T_B^H T_+ x_1\right) \end{bmatrix}. \quad (5.26)$$

Proof. We can proceed largely similar to the proof of Theorem 5.1, although we now have to deal with two Cayley transformations and in total four variables. With

$$\begin{aligned} \mathcal{H} &= (W^H(\hat{\mathcal{C}}_A)^H \hat{\mathcal{C}}_A W)(T^H(\hat{\mathcal{C}}_B)^H \hat{\mathcal{C}}_B T) = (W_+^H W_+)(T_+^H T_+), \\ \hat{\mathcal{C}}_A &= \hat{\mathcal{C}}_A(A, E, \nu, \delta, \xi, \eta) := I_n + \gamma E L_A^{-1}, \quad \hat{\mathcal{C}}_B = \hat{\mathcal{C}}_B(\nu, \delta, \xi, \eta) := I_m - \overline{\gamma} C^T L_B^{-H}, \end{aligned}$$

it holds again by the results from [149, 173] that $h_\chi = y_1^H \frac{\partial}{\partial \chi} \mathcal{H} x_1$, where χ can be any of the variables ν, δ, ξ, η. The derivative of $\mathcal{H}$ is (using the notation $\frac{\partial}{\partial \chi} = \partial_\chi$)

$$\begin{aligned} \partial_\chi \mathcal{H} &= \left(\partial_\chi (W^H \hat{\mathcal{C}}_A^H \hat{\mathcal{C}}_A W)\right)(T^H \hat{\mathcal{C}}_B^H \hat{\mathcal{C}}_B T) + (W^H \hat{\mathcal{C}}_A^H \hat{\mathcal{C}}_A W)\left(\partial_\chi (T^H \hat{\mathcal{C}}_B^H \hat{\mathcal{C}}_B T)\right) \\ &= W^T \left((\partial_\chi \hat{\mathcal{C}}_A^H)\hat{\mathcal{C}}_A + \hat{\mathcal{C}}_A^H \partial_\chi \hat{\mathcal{C}}_A\right) W T_+^H T_+ + W_+^H W_+ T^T \left((\partial_\chi \hat{\mathcal{C}}_B^H)\hat{\mathcal{C}}_B + \hat{\mathcal{C}}_B^H \partial_\chi \hat{\mathcal{C}}_B\right) T \\ &= \left(W^T(\partial_\chi \hat{\mathcal{C}}_A^H) W_+ + W_+^H (\partial_\chi \hat{\mathcal{C}}_A) W\right) T_+^H T_+ \\ &\quad + W_+^H W_+ \left(T^T (\partial_\chi \hat{\mathcal{C}}_B^H) T_+ + T_+^H (\partial_\chi \hat{\mathcal{C}}_B) T\right). \end{aligned}$$

For instance, the partial derivatives of $\hat{\mathcal{C}}_A$, $\hat{\mathcal{C}}_B$ w.r.t. ν are

$$\partial_\nu \hat{\mathcal{C}}_A = -EL_A^{-1}, \quad \partial_\nu \hat{\mathcal{C}}_B = (I_m - \overline{\gamma} C^T L_B^{-H}) C^T L_B^{-H}$$

such that

$$\partial_\nu \mathcal{H} = -(W_A^H W_+ + W_+^H W_A) T_+^H T_+ + W_+^H W_+ \left(T_{B^2,+}^H T_+ + T_+^H T_{B^2,+} \right).$$

Using similar steps for the partial derivatives w.r.t. δ, ξ, and η yields

$$\begin{aligned} \nabla h = y_1^H & \begin{bmatrix} -(W_A^H W_+ + W_+^H W_A) \\ (W_{A^2,+}^H W_+ + W_+^H W_{A^2,+}) \\ \jmath(W_A^H W_+ - W_+^H W_A) \\ -\jmath(W_{A^2,+}^H W_+ - W_+^H W_{A^2,+}) \end{bmatrix} T_+^H T_+ x_1 \\ & + y_1^H W_+^H W_+ \begin{bmatrix} (T_{B^2,+}^H T_+ + T_+^H T_{B^2,+}) \\ -(T_B^H T_+ + T_+^H T_B) \\ \jmath(T_{B^2,+}^H T_+ - T_+^H T_{B^2,+}) \\ -\jmath(T_B^H T_+ - T_+^H T_B) \end{bmatrix} x_1. \end{aligned} \tag{5.27}$$

Obviously, $h(\nu, \delta, \xi, \eta) = \theta_1 \in \mathbb{R}$ such that the equations for the eigentriplet $(\theta_1,\ x_1,\ y_1)$ are

$$(W_+^H W_+)(T_+^H T_+) x_1 = \theta_1 x_1, \quad (T_+^H T_+)(W_+^H W_+) y_1 = \theta_1 y_1.$$

Under the given assumptions, $\hat{\mathcal{C}}_B$ is nonsingular s.t. $T_+^H T_+$ is positive definite and can be factorized as $T_+^H T_+ = R_T^H R_T$, where $R_T \in \mathbb{C}^{r \times r}$ is an upper triangular Cholesky factor. Then with $\tilde{x}_1 := R_T x_1$, $\tilde{y}_1 := R_T^{-H} y_1$, the above eigenvalue equations can be rewritten as

$$R_T (W_+^H W_+) R_T^H \tilde{x}_1 = \theta_1 \tilde{x}_1, \quad R_T (W_+^H W_+) R_T^H \tilde{y}_1 = \theta_1 \tilde{y}_1$$

which reveals $\tilde{x}_1 = \tilde{y}_1$. Hence, for any matrix K of appropriate size it holds

$$\begin{aligned} y_1^H K T_+^H T_+ x_1 &= y_1^H K R_T^H R_T x_1 = \tilde{y}_1^H R_T K R_T^H \tilde{x}_1 = \overline{\tilde{x}_1^H R_T K^H R_T^H \tilde{y}_1} \\ &= \overline{\tilde{y}_1^H R_T K^H R_T^H \tilde{x}_1} = \overline{y_1^H R_T^{-1} R_T K^H R_T^H R_T x_1} = \overline{y_1^H K^H T_+^H T_+ x_1}. \end{aligned}$$

In the same manner it can be shown that $y_1^H W_+^H W_+ K x_1 = \overline{y_1^H W_+^H W_+ K^H x_1}$. As consequence, (5.27) simplifies to (5.26). □

Similar to the formula (5.7), evaluating the gradient requires that two additional linear systems of equations with $L_A,\ L_B$ have to be solved to obtain $W_{A^2,+},\ T_{B^2,+}$. Similar to the GCALE situation, only $W_{A^2,+} y_1$ and $T_{B^2,+} T_+^H T_+ y_1$ are required in (5.26), such that the right hand sides in the necessary linear systems can be reduced to single columns $W_+ y_1$ and $T_+ W_+^H W_+ y_1$, respectively. Thus, in order to evaluate both h and ∇h at the point $\alpha,\ \beta$, the solution of two linear systems with $L_A,\ L_B$ and $r+1$ right hand sides

each, is required. We refrain from using also exact Hessians of h since this did not lead to improvements in the numerical experiments in Section 5.3.3. As in the GCALE case, the assumption that the largest eigenvalue of $(W_+^H W_+)(T_+^H T_+)$ is simple for all values of ν, δ, ξ, η is difficult to ensure in practice. A violation of this requirement might lead to a non-differentiable objective function. As a simple measure to circumvent this issue we again artificially reduce (5.24) to a scalar problem ($r = 1$) by using Wd_W, Td_T with $d_W,\ d_T \in \mathbb{R}^r$ instead of $W,\ T$, where $d_W,\ d_T$ are the left singular vectors corresponding to the largest singular value of $W,\ T$. With this, evaluating the simplified objective function and its gradient requires the solution of two systems with $L_A,\ L_B$ and two right hand sides each.

Using Projected Data For reducing the numerical effort that comes with the required linear solves for evaluating the objective function and its gradients, we pursue a similar approach as in Section 5.3.2. Let Q_j, U_j be the orthonormal basis matrices corresponding to the proposed $V/S(u)$-shifts, and $\tilde{A}_j = Q_j^T A Q_j$, $\tilde{E}_j = Q_j^T E Q_j$ and $\tilde{B}_j = U_j^T B U_j$, $\tilde{C}_j = U_j^T C U_j$ the restrictions of A, E and, respectively, B, C onto span $\{Q_j\}$, span $\{U_j\}$. With $\tilde{W}_j = Q_j^T W_j$ and $\tilde{T}_j = U_j^T T_j$, one can construct a reduced objective function

$$\begin{aligned} \tilde{h}_j &= \lambda_{\max}(\tilde{W}(\alpha,\beta)^H \tilde{W}(\alpha,\beta)\tilde{T}(\alpha,\beta)^H \tilde{T}(\alpha,\beta)), \\ \tilde{W}(\alpha,\beta) &= \tilde{W}_j + (\beta-\alpha)\tilde{E}_j((\tilde{A}_j - \beta\tilde{E}_j)^{-1}\tilde{W}_j), \\ \tilde{T}(\alpha,\beta) &= \tilde{T}_j - \overline{(\beta-\alpha)}\tilde{C}_j^T((\tilde{B}_j - \alpha\tilde{C}_j)^{-H}\tilde{T}_j) \end{aligned} \tag{5.28}$$

whose evaluation is now cheap due to the small dimension ru of the involved matrices defining the linear systems. The same holds for the gradient of (5.28) which can be constructed by (5.26) using the projected matrices.

Constraints for the Optimization Variables and Initial Guesses Also similar to the GCALE case, we employ boxed constraints for the optimization variables:

$$\begin{aligned} \nu_- \le \nu \le \nu_+, \quad 0 \le \xi \le \xi_+, \\ \delta_- \le \delta \le \delta_+, \quad 0 \le \eta \le \eta_+. \end{aligned} \tag{5.29}$$

If the spectra $\Lambda(A,\ E)$ and $\Lambda(B,\ C)$ can be strictly separated by a vertical line (cf. Corollary 2.41), this separation should be reflected in (5.29). For instance, when $\Lambda(A,\ E) \in \mathbb{C}_-$ and $\Lambda(B, C) \in \mathbb{C}_+$, i.e., the vertical line is positioned at the origin, the used shifts should obey this restriction, i.e., Re (β) − Re $(\alpha) = \delta - \nu > 0$, and we set $\nu_+ < 0$ and $\delta_- > 0$ in (5.29). This also automatically ensures that $\beta \neq \alpha$. We point out that the already used as well as the upcoming GCASE examples fit into this class. We know that for GCALEs defined by coefficients with $\Lambda(A,\ E) \in \mathbb{C}_-$, the G-LR-ADI iteration constitutes a convergent process as long as also the shifts have negative real parts since $\rho(\mathcal{C}(A, E, \alpha)) < 1$. The situation for GCASEs is considerably more complicated. From the viewpoint of (5.21), (5.22), and the reduced objective function (3.39), we may

demand that every pair of shifts α, β satisfies

$$q(\alpha,\beta) := \left| \frac{(\tilde{\lambda}-\alpha)(\tilde{\mu}-\beta)}{(\tilde{\lambda}-\beta)(\tilde{\mu}-\alpha)} \right|^2 < 1, \quad \forall \tilde{\lambda} \in \Lambda(\tilde{A}_j,\ \tilde{E}_j),\ \forall \tilde{\mu} \in \Lambda(\tilde{B}_j,\ \tilde{C}_j), \tag{5.30}$$

which implies $\sup_{\tilde{\lambda},\ \tilde{\mu}} q(\alpha,\beta) < 1$. This represents r^2u^2 conditions to be satisfied by the optimization variables. In terms of $\nu,\ \delta,\ \xi,\ \eta$, the rational function $q(\alpha,\beta)$ can be written as

$$q(\alpha,\beta) = \frac{|\tilde{\lambda}|^2+\nu^2+\xi^2-2(\mathrm{Re}(\tilde{\lambda})\nu+\mathrm{Im}(\tilde{\lambda})\xi)}{|\tilde{\lambda}|^2+\delta^2+\eta^2-2(\mathrm{Re}(\tilde{\lambda})\delta+\mathrm{Im}(\tilde{\lambda})\eta)} \cdot \frac{|\tilde{\mu}|^2+\delta^2+\eta^2-2(\mathrm{Re}\ (\tilde{\mu})\,\delta+\mathrm{Im}\ (\tilde{\mu})\,\eta)}{|\tilde{\mu}|^2+\nu^2+\xi^2-2(\mathrm{Re}\ (\tilde{\mu})\,\nu+\mathrm{Im}\ (\tilde{\mu})\,\xi)}.$$

These additional restrictions can be added as nonlinear constraints of $\nu,\ \delta,\ \xi,\ \eta$ to the optimization problem (5.25). Numerical experiments show that without (5.30) the obtained shift parameters can lead to an oscillatory behavior of $\|\mathcal{S}_j\|$.

If the applied optimization algorithm demands an initial guess, we follow the same strategy as in the GCALE case, i.e., we take the first pair of shifts returned by the heuristic approach [162, 43] on the basis of $\Lambda(\tilde{A}_j,\ \tilde{E}_j)$, $\Lambda(\tilde{B}_j,\ \tilde{C}_j)$. This can also be related to (5.30), where this pair of shifts leads to the smallest nonzero value of $q(\alpha,\ \beta)$ evaluated at $\Lambda(\tilde{A}_j,\ \tilde{E}_j)$, $\Lambda(\tilde{B}_j,\ \tilde{C}_j)$.

5.4.3. Numerical Experiments

In this section, we test some of the proposed shift strategies for the Sylvester ADI iteration. In all examples, the G-fADI iteration is terminated when $\varepsilon_j := \|\mathcal{S}_j\|/\|FG^T\| < \tau$ with $\tau = 10^{-10}$ or after $j^{\max}$ iteration steps.

We take the following test examples. At first, the example *FMD-S* introduced in Section 4.2.3 consisting of two variations of the *FDM* system. We increase the leading dimensions to $n = 40000$, $m = 22500$ (corresponding to $n_0 = 200$ and $m_0 = 150$ grid points) and keep $r = 4$.

As in [21], we construct a GCASE by merging two version of the *rail* example. The matrices A, E, F and $-B, C, -G$ are taken from *rail79* and *rail20k*, respectively, where F, G are the corresponding input matrices of the associated dynamical system (2.5). Since $\|FG^T\| \approx 10^{-14}$ we use $10^4 \cdot F,\ 10^4 \cdot G$ instead to avoid potential numerical instabilities by scalings with $\|FG^T\|$. This GCASE defined by symmetric matrices with $n = 79881$, $m = 20209$, and $r = 7$ is abbreviated by *rail79k/20k*.

In a similar way, the third example is a larger version of the example *ifiss16k/4k* from Sections 3.3.4 and 4.2.3. We merge the matrices from *ifiss66k* and *ifiss16k* to the example called *ifiss66k/16k* with $n = 66049$, $m = 16641$, $r = 5$.

We compare the performance of the G-fADI iteration when different the shift parameter approaches are used. The approximate optimal shifts for solving the two-variable ADI shift parameter problem (5.22) are obtained by the `parsyl` routine [233], which we modified such that (inverse) Arnoldi processes are used to obtain the required approximate spectral data. This is more efficient than using `eigs` as it is done in the original

Table 5.3.: Setup data and results for the Sylvester examples using different shift strategies. The smallest values of t_{total} and j^{iter} are emphasized by bold letters.

Example	shift strategy	t_{shift}	t_{ADI}	t_{total}	j^{iter}	$\varepsilon_{j^{\text{iter}}}$
FDM-S $j^{\max} = 60$ active-set	heur(20, 10, 10, 20, 10, 10)	3.9	26.6	30.5	55	$9.42{\cdot}10^{-10}$
	Wachs(10, 10, 10, 10), J =12	3.7	22.4	26.1	48	$5.74{\cdot}10^{-10}$
	$V/S(2)$-shifts	**0.1**	17.3	**17.5**	**36**	$9.13{\cdot}10^{-10}$
	W/T-shifts	**0.1**	18.7	18.8	42	$1.04{\cdot}10^{-10}$
	$V/S(2)$-res.min.appr.	5.2	16.4	21.6	**36**	$4.69{\cdot}10^{-10}$
rail79k/20k $j^{\max} = 200$ active-set	heur(80, 80, 40, 40, 40, 40)	31.4	217.5	248.9	153	$6.25{\cdot}10^{-10}$
	Sabino(10, 5, 10, 5), $J = 100$	4.9	123.6	128.5	93	$3.36{\cdot}10^{-10}$
	$V/S(4)$-shifts	**2.0**	92.6	94.6	73	$1.29{\cdot}10^{-10}$
	$V/S(2)$-res.min.appr.	10.4	64.5	**74.9**	**53**	$5.36{\cdot}10^{-10}$
ifiss66k/16k $j^{\max} = 150$ interior-point	heur(30, 30, 10, 20, 10, 20)	16.2	136.6	152.9	139	$9.95{\cdot}10^{-10}$
	Wachs(20, 20, 20, 20), J =64	18.9	124.4	143.2	126	$6.78{\cdot}10^{-10}$
	$V/S(2)$-shifts	**1.1**	108.0	**109.1**	106	$6.63{\cdot}10^{-10}$
	$V/S(2)$-res.min.appr.	22.7	93.3	116.0	**92**	$1.45{\cdot}10^{-10}$

`parsyl` implementation. We refer to this strategy by Wachs(k_+^A, k_-^A, k_+^B, k_-^B), where k_+^A, k_-^A, k_+^B, k_-^B denote the orders of the used (inverse) Krylov subspaces. The number $J = L$ of returned shifts is also given. For example *rail79k/20k*, the `parsyl` does not provide good shifts and the method presented in [202, Algorithm 2.1], abbreviated by Sabino(k_+^A, k_-^A, k_+^B, k_-^B), is employed instead. The heuristic shift approach [43, 162] w.r.t. (5.22) is denoted by heur(J, L, k_+^A, k_-^A, k_+^B, k_-^B), employing also (inverse) Arnoldi processes. The generating of the orthonormal bases for the $V/S(u)$- and W/T-shifts is done as in Section 5.3.3. The execution of the optimization process for the projected residual-norm minimizing shifts is also carried out with the help of `fmincon`. However, due to the additional nonlinear constraints (5.30), the trust-region-reflective algorithm, which we used before, is no longer applicable. The active-set or interior-point algorithms are employed instead.

The setup data as well as the results are summarized in Table 5.3, where the first column also indicates which optimization algorithm is used. Additionally, Figure 5.5 shows the scaled residual norm against the iteration number for the examples *FDM-S* and *ifiss66k/16k*.

In part, similar observations can be made as in the GCALE examples. The heuristic shifts manage to steer the G-fADI iteration to the desired accuracy within $j^{\max}$ iterations for all examples. Compared to the heuristic shifts, the Wachspress type shifts require smaller values of k_+^A, k_-^A, k_+^B, k_-^B to get the necessary spectral data, leading, therefore, to smaller times t_{shift}. They also seem to converge similarly to the Wachspress shifts for Lyapunov equations with real spectra. However, the required setup numbers seemed to be highly influential for the performance of both the heuristic and Wachspress shifts. Different values than the ones used here led to a clearly different and often slower

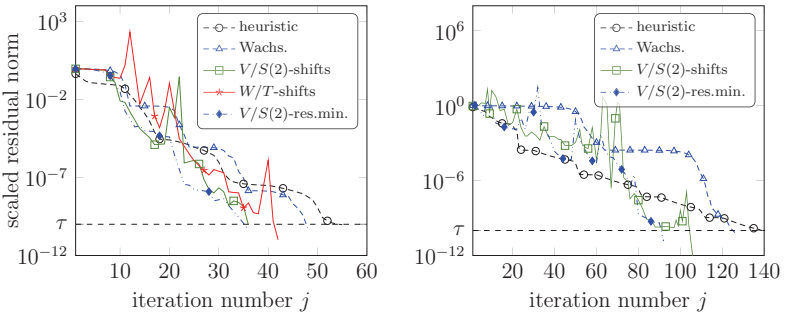


Figure 5.5.: Scaled residual norm against iteration index j of the G-fADI iteration using different shift strategies for the *FDM-S* (left) and *ifiss66k/16k* (right) example.

convergence, especially for the examples *FMD-S* and *ifiss66k/16k* which have complex spectra.

As before, the $V/S(u)$- as well as the W/T-shifts obtained from projections onto spaces spanned by G-fADI iterates or residual factors require the smallest generation times t_{shift}. However, the W/T-shifts only lead to convergence for the example *FDM-S*. The residual history of both, the $V/S(u)$- and W/T-shifts, seems to be highly oscillatory as it is clearly visible in the residual plots in Figure 5.5 (right plot). There are very high spikes in $\|\mathcal{S}_j\|$ which appear to unnecessarily prolong the iteration. An investigation of this phenomenon revealed that, in terms of (5.21), these peaks are the result of pairs of shifts $\alpha_k,\ \beta_k$ with $\rho(\mathcal{C}_A^{(k)})\rho(\mathcal{C}_B^{(k)}) \gg 1$. This indicates that the corresponding computed shifts α_k, β_k are of no good quality. In some situations and other experiments, these spikes were so high that numerical overflow led to unusable results in the end. In the results presented in Table 5.3, the G-fADI iteration manages to recover from these instances. The approximate residual norm-minimizing shifts largely avoid such oscillations thanks to the constraint (5.30). Without (5.30) such peaks are also present for the residual norm-minimizing shifts. The plots in Figure 5.5 reveal that the residual norm-minimizing shifts allow a much smoother and monotonic decrease of $\|\mathcal{S}_j\|$ compared to the $V/S(u)$- and W/T-shifts which also leads to the smallest number of iteration steps j^{iter} for all examples except for *ifiss66k/16k*. There, the residual plot in Figure 5.5 shows some smaller peaks towards the end of the iteration. However, solving the reduced optimization problem (5.25) appears to be more formidable than (5.5) for the G-LR-ADI iteration. This shows in the generation times t_{shift} in Table 5.3 which are sometimes clearly larger compared to the other shift approaches. The example *ifiss66k/16k* appears to be exceptionally difficult for all shift strategies but also w.r.t. the solution of the optimization problem (5.25). Only the interior-point algorithm leads to useful results but still has severe issues in detecting a local minimizer in some iteration steps.

The existing shift approaches are outperformed by the proposed self-generating strate-

gies but there is still room for improvement. Modifying the $V/S(u)$-shift such that the large spectral radii caused by infeasible shifts do not occur, might lead to a further performance improvement. At the current stage they, unfortunately, lead to numerical instabilities for difficult GCASEs. However, their small execution and generations times, as well as the advantage that they are computed in an entirely automatic way, makes the $V/S(u)$-shifts nevertheless a promising approach. The $V/S(u)$-residual norm-minimizing shifts lead to a much smoother convergence behavior and therefore more robust execution of the G-fADI iteration. They can also be computed automatically for the most part, provided the optimization routine is able to detect local minima of (5.25) with low effort. As for the GCALE case, future research effort should be devoted to improving the numerical solution of this optimization problem.

5.5. Summary and Further Research Perspectives

5.5.1. Conclusions

We have discussed shift parameter strategies for low-rank ADI methods for solving large-scale Lyapunov and Sylvester equations. After reviewing some prominent approaches to compute shifts a-priori, two novel strategies have been proposed which generate shifts automatically during the ADI iteration without any setup data needed. The first one uses Galerkin projections onto spaces spanned by the current ADI iterates to obtain a small number of Ritz values as next shifts. The second one is intrinsically designed to compute the new shift such that the residual norm is minimized at each step. The latter approach was enhanced by using derivative information and a projection framework in order to significantly reduce the computation costs compared to the original setting in [39]. The obtained shift parameters showed impressive numerical results that considerably outperformed the existing shift strategies w.r.t. both the required execution time and the required number of ADI iteration steps. The projection based V-, V/S-shifts, and the associated residual norm-minimizing shifts, are definitely competitive to existing shift parameter approaches, especially for problems with complex spectra. With these new shift generation strategies, numerical experiments confirm that the G-LR-ADI iteration for GCALEs compares very well with other low-rank algorithms such as EKSM [209] and RKSM [82]. For Sylvester equations, the proposed dynamically updated shifts can lead to an oscillatory residual behavior which deteriorates the convergence, especially for the V/S-shifts. The (approximate) Wachspress shifts appear to be still a viable choice for problems with real spectra, even though the novel strategies managed to outperform them in some cases. Moreover, in contrast to the existing precomputed shift approaches, the novel strategies admit a largely automatic shift parameter generation in the course of the iteration, which represents a clear user-oriented advantage.

5.5.2. Future Research Possibilities and Outlook

We already mention in the previous section that the optimization process for the residual-norm minimizing shifts can be improved further, especially for the G-fADI iteration for GCASEs.

Here, we propose a few additional strategies that might further enhance the shift generation itself, but also the overall ADI iteration. We restrict to the G-LR-ADI iteration since generalizations to the G-fADI iteration are obvious.

The first extension of the residual-norm minimizing shifts is of conceptual nature. Our motivation for considering (5.5) is to maximally reduce the residual norm from one iteration step to the next by finding an appropriate shift parameter. In other words, the goal is to decrease $\|\mathcal{L}_j\|$ down to $\|\mathcal{L}_{j+1}\|$ as much as possible. It is easy to incorporate more than one, say $\ell \geq 1$, future iteration steps by considering

$$\begin{aligned} W_{j+\ell} &= \mathcal{C}(A, E, \alpha_{j+\ell}) W_{j+\ell-1} = \mathcal{C}(A, E, \alpha_{j+\ell}) \mathcal{C}(A, E, \alpha_{j+\ell-1}) W_{j+\ell-2} \\ &= \ldots = \left(\prod_{i=1}^{\ell} \mathcal{C}(A, E, \alpha_{j+i}) \right) W_j. \end{aligned}$$

After iteration step j is completed, W_j is known and, thus, the next ℓ shift parameters $\alpha_{j+i} = \nu_i + \jmath\xi_i$, $i = 1, \ldots, \ell$ are obtained from the solution of the optimization problem

$$\begin{aligned} [\nu_1, \ldots, \nu_\ell, \xi_1, \ldots, \xi_\ell] &= \operatorname*{argmin}_{\boldsymbol{\nu} \in \mathbb{R}_-^\ell, \boldsymbol{\xi} \in \mathbb{R}_+^\ell} f_{j,\ell}(\boldsymbol{\nu}, \boldsymbol{\xi}), \\ f_{j,\ell}(\boldsymbol{\nu}, \boldsymbol{\xi}) &:= \left\| \left(\prod_{i=1}^{\ell} \hat{\mathcal{C}}(A, E, \boldsymbol{\nu}(i), \boldsymbol{\xi}(i)) \right) W_j \right\|^2 \end{aligned} \tag{5.31}$$

in 2ℓ variables. Here, ℓ can be seen as a horizon defining how many future iterations we look ahead. If the global minimum of (5.31) is found, the resulting residual norm $\|W_{j+\ell}\|$ will be the smallest one possible w.r.t. to all admissible sets of ℓ shift parameters. Assuming that all pairs $(\nu_i,\ \xi_i)$ are distinct and taking the complex conjugates of the obtained shifts into account (i.e., using $-\xi_i$), these residual minimizing-shifts can be used in up to 2ℓ iterations. On the one hand, solving (5.31) is likely more demanding than (5.5) but, on the other hand, it has to be dealt with only every ℓ-th (or 2ℓ-th) iteration step. Preliminary tests showed no significant speed ups of the G-LR-ADI iteration that were worth the additional effort resulting from solving the harder optimization problem (5.31).

A second potential further research topic was already partly discussed when the reduced objective function (5.14) was introduced. The approach based on the shift-invariance of Krylov subspaces can be enhanced if the occurring linear systems of the G-LR-ADI iteration are dealt with by iterative Krylov subspace methods. We could then in principle employ a multi-shift variant, e.g., [103, 85, 226, 2, 102], of a Krylov subspace method for solving all linear system that occur in the optimization routine for (5.5) and, after a local minimizer α_{j+1} has been detected, also for solving the linear system $(A + \alpha_{j+1} I_n) V_{j+1} = W_j$ of the G-LR-ADI iteration. The required iterate V_{j+1} could

also be reused from the final iteration step of the optimization routine. In [160] it is shown that the LR-ADI iteration can be rearranged such that the right hand sides of the linear systems in every iteration step are always equal to F. Hence, the use of multi-shift Krylov methods can even be extended to the whole algorithm. However, the drawbacks as mentioned in Section 5.3.2 prevent the success of this strategy: the absence of a priori knowledge of the occurring shifts and the difficulties arising in the situation $E \neq I$. Also note that, as remarked in [202], finding good ADI shift parameters might compete with the goal to obtain a fast convergence of the employed Krylov solver. If these issues can be solved in future research, this might lead to a highly efficient low-rank ADI iteration incorporating preconditioned iterative solves and high quality residual norm-minimizing shifts.

CHAPTER 6

LOW-RANK NEWTON METHODS FOR ALGEBRAIC RICCATI EQUATIONS

Contents

This chapter is concerned with the computation of factorized solutions of low-rank of continuous-time and non-symmetric algebraic Riccati equations. For this, low-rank implementations of a Newton's scheme will be considered, where the occurring Lyapunov and Sylvester equations are dealt with by the respective low-rank ADI iteration. The previously established enhancements of the low-rank ADI iteration are incorporated as well.

6.1. Continuous-time Algebraic Riccati Equations

We consider generalized, continuous-time, algebraic Riccati equations (GCARE)

$$\mathcal{R}(X) = A^T X E + E^T X A - E^T X F F^T X E + C^T C = 0 \tag{6.1}$$

with $A,\ E,\ X \in \mathbb{R}^{n\times n}$, E nonsingular, $C \in \mathbb{R}^{p\times n}$, and $F \in \mathbb{R}^{n\times r}$ (cf. Definition 2.34). Here, in contrast to the GCALEs (3.12), the right most coefficient is A^T because such

GCAREs often arise in the design of linear-quadratic-regulators (LQR) to stabilize or control generalized state-space systems of the form (2.5). There, one is interested in the stabilizing solution X_* such that $\Lambda(A - FK^T, E) \subset \mathbb{C}_-$, where $K := E^T X_* F \in \mathbb{R}^{n\times r}$ is usually referred to as stabilizing feedback matrix. In contrast to GCALEs, this also means that the spectrum $\Lambda(A,\ E)$ is allowed to have unstable eigenvalues. We assume in the following that the conditions of Lemma 2.35 are fulfilled which ensure the existence and uniqueness of X_*. Moreover, with $r, p \ll n$ one can expect that a fast decay of the singular values of X_* which, as in the GCALE and GCASE cases, motivates to approximate X_* by a low-rank solution $X_* \approx ZZ^T$ with $Z \in \mathbb{R}^{n\times\ell}$, $\operatorname{rank}(Z) = \ell \ll n$. The singular value decay of X_* as well as the existence of such a low-rank approximation is investigated in [23].

6.1.1. Newton Methods for GCAREs

Because the Riccati operator $\mathcal{R}(X)$ is nonlinear in X, one can apply a general Newton iteration to find X. Starting from an initial guess X_0, the step k of this iteration is, following, e.g., [5, 150], given by

$$X^{(k)} = X^{(k-1)} + N^{(k-1)} \tag{6.2a}$$

with the update $N^{(k-1)}$ obtained from the Newton step

$$\mathcal{R}'|_{X^{(k-1)}}(N^{(k-1)}) = -\mathcal{R}(X^{(k-1)}). \tag{6.2b}$$

There, $\mathcal{R}'|_X$ is the Fréchet derivative of the Riccati operator (6.1) at X and is given by

$$\mathcal{R}'|_X \ :\ N \mapsto (A - FF^T XE)^T NE + E^T N(A - FF^T XE). \tag{6.2c}$$

Hence, obtaining $N^{(k-1)}$ demands the solution of the GCALE

$$(A^{(k)})^T N^{(k)} E + E^T N^{(k)} A^{(k)} = -\mathcal{R}(X^{(k-1)}) \tag{6.3}$$

with the closed-loop system matrix $A^{(k)} := A - F(K^{(k-1)})^T$. By using the Kleinman trick [145], it is possible to solve for $X^{(k)}$ directly which changes (6.2b) to

$$\mathcal{R}'|_{X^{(k-1)}}(X^{(k)}) = -\hat{F}^{(k)}(\hat{F}^{(k)})^T, \quad \hat{F}^{(k)} := [C^T, K^{(k-1)}] \tag{6.4a}$$

with $K^{(k-1)} := E^T X^{(k-1)} F$. In terms of the corresponding GCALE to be solved this becomes

$$(A^{(k)})^T X^{(k)} E + E^T X^{(k)} A^{(k)} = -\hat{F}^{(k)}(\hat{F}^{(k)})^T. \tag{6.4b}$$

The resulting Newton-Kleinman iteration is illustrated in Algorithm 6.1. Both formulations (6.2) and (6.4) are in fact equivalent and produce the same iterates $X^{(k)}$ provided $K^{(0)} = E^T X^{(0)} F$ in Algorithm 6.1. It is shown in [150, 145] that, under the conditions

Algorithm 6.1: Newton-Kleinman method for GCAREs

Input : Matrices $A,\ E,\ F,\ C$ defining (6.1), initial guess $X^{(0)}$ or $K^{(0)}$.
Output: Approximate solution X.

1 **for** $k = 1, \ldots, k_{\max}$ **do**
2 $\quad \hat{F}^{(k)} := [C^T, K^{(k-1)}]$.
3 $\quad$ Solve the GCARE

$$(A - F(K^{(k-1)})^T)^T X^{(k)} E + E^T X^{(k)} (A - F(K^{(k-1)})^T) + \hat{F}^{(k)} (\hat{F}^{(k)})^T = 0$$

$\quad$ for $X^{(k)}$.
4 $\quad$ Set $K^{(k)} := E^T X^{(k)} F,$.

of Theorem 2.35 and provided $X^{(0)}$ is stabilizing, the sequence of iterates converges quadratically towards X_*. Moreover, it satisfies

$$X^{(1)} \succeq X^{(2)} \succeq \ldots \succeq X_* \quad \text{and} \quad \Lambda(A - FF^T X^{(k)} E) \subset \mathbb{C}_- \quad \forall k > 0.$$

A further improvement of the convergence behavior by using a line-search strategy can, e.g., be found in [25, 16]. If $(A,\ E)$ is Hurwitz, $X_0 = 0$ is a viable stabilizing initial guess. Otherwise, if $\Lambda(A,\ E)$ contains also anti-stable eigenvalues, finding a good initial guess is a formidable task. Some strategies are given in [150, Section 5.3],[55, Section 3.3.1], and the references therein. We will later on mention an approach that is also applicable for large-scale problems.

Comparing the GCALEs to be solved in both the original Newton and the Newton-Kleinman iteration, one observes that (6.3) and (6.4b) only differ in the right hand side. In the following we investigate the handling of the occurring GCALEs by the G-LR-ADI iteration which was studied in the chapters before. The Newton-Kleinman iteration is the better choice for this because it holds

$$\operatorname{rank}\left(\hat{F}^{(k)}(\hat{F}^{(k)})^T\right) \leq \operatorname{rank}\left(\hat{F}^{(k)}\right) \leq p + r \ll n,$$

i.e., the right hand side has a very small rank. The right hand sides $-\mathcal{R}(X^{(k-1)})$ of the original Newton iteration (6.2b) do, in general, not share this property.

6.1.2. The Low-Rank Newton-ADI for GCAREs

Introducing the G-LR-ADI (Algorithm 3.2) into the Newton-Kleinman method (Algorithm 6.1) is straightforward and has already been considered in, e.g., [42, 201, 135, 94, 49, 50, 10, 52]. The main goal of this section is to also insert the improvements for the G-LR-ADI iteration established in the previous three chapters into this framework. Using the G-LR-ADI iteration to solve (6.4b) for each k, transforms the Newton-Kleinman method into an inner-outer iteration, consisting of an outer, the Newton iteration, and an inner iteration, the G-LR-ADI iteration. To distinguish these two stages we will from

now on use the notation that subscripts $_j$ and bracketed superscripts $^{(k)}$ will refer to quantities associated to the inner and outer iteration, respectively. The inner-iteration in our novel reformulation (Section 3.2, Algorithm 3.2) follows in each outer iteration k the scheme

$$V_j^{(k)} = (A^{(k)} + \alpha_j^{(k)} E)^{-H} W_{j-1}^{(k)}, \quad W_j^{(k)} = W_{j-1}^{(k)} - 2\,\mathrm{Re}\,(\alpha_j)\, E^T V_j^{(k)}, \tag{6.5}$$
$$Z_j^{(k)} = [Z_{j-1}^{(k)},\ \sqrt{-2\,\mathrm{Re}\,(\alpha_j)} V_j^{(k)}], \quad j = 1, \ldots, j_{\max},$$

where $W_0^{(k)} := [C^T, K^{(k-1)}]$, $Z_0^{(k)} = [\,]$, and $\alpha_j^{(k)}$ are shift parameters corresponding to $(A^{(k)},\ E)$. As before we indent to use the residual norm based criterion

$$\|\mathcal{L}_j^{(k)}\| \le \tau_{\mathrm{ADI}} \|W_0^{(k)}\|^2, \quad 0 < \tau_{\mathrm{ADI}} \ll 1$$

for terminating the inner iteration. Including also the relations (4.8) regarding the treatment of complex shifts as in G-LR-ADI-r (ALgorithm 4.3) leads to the low-rank Newton-ADI iteration for GCAREs (LR-NADI-C) illustrated in Algorithm 6.2. We naturally assume that the sets of shift parameters $\{\alpha_1^{(k)}, \ldots, \alpha_J^{(k)}\}$ are proper. Hence, the LR-NADI-C iteration will produce real low-rank solution factors and, consequently, also real feedback matrices $K^{(k)}$. Some numerical evidence that this leads to computational savings can be found in [38, Section 5.2].

In the following we go through some further aspects and subproblems of Algorithm 6.2.

Solving the Linear Systems

The coefficient matrices $A^{(k)} + \alpha_j^{(k)} E$ in the linear systems in Line 6 are the sum of a sparse matrix $A + \alpha_j^{(k)} E$ and a low-rank update $F(K^{(k)})^T$. Hence, if sparse-direct solvers are employed, the Sherman-Morrison-Woodbury formula [111] might be applied:

$$[V_S,\ V_K] = (A + \alpha_j^{(k)} E)^{-H} [W_{j-1}^{(k)},\ K^{(k-1)}], \tag{6.6}$$
$$V_j^{(k)} = V_S + V_K \left(I_p - F^T V_K\right)^{-1} (F^T V_S). \tag{6.7}$$

Hence, $2r + p$ linear systems with the sparse coefficient matrix $A + \alpha_j^{(k)} E$ have to be solved in each inner iteration. Iterative solvers [199, 225] that work only with matrix vector products of the coefficient matrices might be applied directly to $A^{(k)} + \alpha_j^{(k)} E$. However, the low-rank updates might severely increase the condition number such that the Sherman-Morrison-Woodbury formula might still be preferable.

Implicit Updates of the Feedback Matrices

In each inner iteration, the intermediate feedback matrices can be updated recursively by

$$K_j^{(k)} = -2 \sum_{i=1}^{j} E^T\,\mathrm{Re}\,(\alpha_i)\, V_i^{(k)} (V_i^{(k)})^H F = K_{j-1}^{(k)} - 2\,\mathrm{Re}\,(\alpha_j)\, E^T V_j^{(k)} (V_j^{(k)})^H F$$

Algorithm 6.2: Low-rank Newton-ADI for GCAREs (LR-NADI-C)

Input : Matrices $A,\ E,\ F,\ C$ defining (6.1), initial feedback $K^{(0)}$, and stopping tolerances $0 < \tau_{\text{ADI}}, \tau_{\text{NM}} \ll 1$.

Output: $Z^{k_{\max}} \in \mathbb{R}^{n\times(r+p)j}$ such that $Z^{k_{\max}}(Z^{k_{\max}})^T \approx X$, stabilizing feedback matrix $K^{k_{\max}} \in \mathbb{R}^{n\times r}$ with $\Lambda(A - F(K^{k_{\max}})^T,\ E) \subset \mathbb{C}_-$.

1 **for** $k = 1, \ldots, k_{\max}$ **do**
2 Determine shifts $\{\alpha_1^{(k)}, \ldots, \alpha_J^{(k)}\}$ w.r.t. $(A^{(k)} := A - F(K^{(k-1)})^T,\ E)$.
3 $W_0^{(k)} = [C^T, K^{(k-1)}]$, $Z_0^{(k)} = [\,]$, $K_0^{(k)} = 0$, $j = 0$.
4 **while** $\|W_j^{(k)}\|^2 > \tau_{ADI}\|W_0^{(k)}\|^2$ **do**
5 $j = j + 1$
6 Solve $(A^{(k)} + \alpha_j^{(k)}E)^H V_j^{(k)} = W_{j-1}^{(k)}$ for $V_j^{(k)}$.
7 **if** $\operatorname{Im}\left(\alpha_j^{(k)}\right) = 0$ **then**
8 $\gamma_j^{(k)} = \sqrt{-2\alpha_j^{(k)}}$, $V_+ := (\gamma_j^{(k)})^2 E^T V_j^{(k)}$.
9 $W_j^{(k)} = W_{j-1}^{(k)} + V_+$, $Z_j^{(k)} = [Z_{j-1}^{(k)},\ \gamma_j^{(k)} V_j^{(k)}]$.
10 $K_j^{(k)} = K_{j-1}^{(k)} + V_+(V_j^{(k)})^T F$.
11 **else**
12 $\gamma_j^{(k)} = 2\sqrt{-\operatorname{Re}\left(\alpha_j^{(k)}\right)}$, $\quad \delta_j^{(k)} = -\frac{\operatorname{Re}\left(\alpha_j^{(k)}\right)}{\operatorname{Im}\left(\alpha_j^{(k)}\right)}$.
13 $W_{j+1}^{(k)} = W_{j-1}^{(k)} + (\gamma_j^{(k)})^2 E^T \left(\operatorname{Re}\left(V_j^{(k)}\right) + \delta_j^{(k)} \operatorname{Im}\left(V_j^{(k)}\right)\right)$.
14 $Z_+ := \gamma_j^{(k)} \left[\operatorname{Re}\left(V_j^{(k)}\right) + \delta_j^{(k)} \operatorname{Im}\left(V_j^{(k)}\right),\ \sqrt{((\delta_j^{(k)})^2 + 1)} \cdot \operatorname{Im}\left(V_j^{(k)}\right)\right]$.
15 $Z_{j+1}^{(k)} = \left[Z_{j-1}^{(k)},\ Z_+\right]$, $K_{j+1}^{(k)} = K_{j-1}^{(k)} + E^T Z_+ (Z_+^T F)$.
16 $j = j + 1$.
17 $K^{(k)} = K_j^{(k)}$, $Z^{(k)} = Z_j^{(k)}$.

as in Line 10 and, in the case of a complex shift, Line 15. The matrix vector products with E^T can be reused from the construction of $W_j^{(k)}$. Therefore, storing the low-rank solution factors $Z_j^{(k)}$ is not needed at all if only the feedback matrices are of interest. This leads to significant savings regarding the memory requirements of Algorithm 6.2 and was already exploited in the implicit low-rank Newton method [42, Algorithm 6].

Computing Initial Guesses

The outer Newton iteration might benefit from a nonzero $K^{(0)}$ as initial guess, especially if $K^{(0)} \approx E^T X_* F$. If $\Lambda(A,\ E)$ contains unstable eigenvalues, a nonzero $K^{(0)}$ is even necessary. Here, we briefly mention one often used approach to generate such $K^{(0)}$ for large-scale problems [10]. Let $(\lambda_{\text{us},i},\ q_{\text{us},i},\ y_{\text{us},i})$, $i = 1, \ldots, n_{\text{us}} \ll n$, be the eigentriples

corresponding to the unstable eigenvalues $\lambda_{\mathrm{us},i} \in \Lambda(A,\ E) \cap \mathbb{C}_+$ and define

$$Q_{\mathrm{us}} = [q_{\mathrm{us},1}, \ldots, q_{\mathrm{us},n_{\mathrm{us}}}], \quad Y_{\mathrm{us}} = [y_{\mathrm{us},1}, \ldots, y_{\mathrm{us},n_{\mathrm{us}}}] \in \mathbb{C}^{n \times n_{\mathrm{us}}}.$$

Then let $\tilde{X}_{\mathrm{ABE}}$ be the solution of the generalized algebraic Bernoulli equation (GABE)

$$\begin{aligned} \tilde{A}_{\mathrm{us}}^H \tilde{X}_{\mathrm{ABE}} \tilde{E}_{\mathrm{us}} + \tilde{E}_{\mathrm{us}}^H \tilde{X}_{\mathrm{ABE}} \tilde{A}_{\mathrm{us}}^H - \tilde{E}_{\mathrm{us}}^H \tilde{X}_{\mathrm{ABE}} \tilde{B}_{\mathrm{us}} \tilde{B}_{\mathrm{us}}^H \tilde{X}_{\mathrm{ABE}} \tilde{E}_{\mathrm{us}} = 0, \\ \tilde{A}_{\mathrm{us}} := Y_{\mathrm{us}}^H A Q_{\mathrm{us}}, \quad \tilde{E}_{\mathrm{us}} := Y_{\mathrm{us}}^H E Q_{\mathrm{us}}, \quad \tilde{B}_{\mathrm{us}} := Y_{\mathrm{us}}^H B. \end{aligned} \tag{6.8}$$

The initial feedback matrix is then

$$K^{(0)} = E^T Y_{\mathrm{us}} \tilde{X}_{\mathrm{ABE}} \tilde{B}_{\mathrm{us}} \in \mathbb{C}^{n \times r} \tag{6.9}$$

and it can be shown that $-\lambda_{\mathrm{us},i} \in \Lambda(A - B(K^{(0)})^H,\ E)$. The GABE (6.8) is of dimension n_{us} and can be solved by methods for small, dense Bernoulli equations, e.g., [13]. The required eigentriples can be computed by iterative algorithms for large-scale eigenvalue problems, see, e.g., [198, 6, 192]. In practice, however, it can be difficult to guarantee that said eigenvalue methods have found all unstable eigenvalue of $\Lambda(A,\ E)$. For the use in Algorithm 6.2, it is reasonable to consider a real initial feedback matrix $K^{(0)}$ by constructing Q_{us}, Y_{us} using the real and imaginary parts of the right and left eigenvectors corresponding to complex unstable eigenvalues.

The Riccati Residual Matrix and Stopping Criteria

By Theorem 3.5, the Lyapunov residual matrix for each j is given by $\mathcal{L}_j^{(k)} = W_j^{(k)} (W_j^{(k)})^H$ and, hence, $\operatorname{rank}\left(\mathcal{L}_j^{(k)}\right) \leq p + r$. Granted by this low-rank factorization of $\mathcal{L}_j^{(k)}$, a similar result can be established for the GCARE residual matrix. For brevity, we keep the complex formulation of the G-LR-ADI iteration since the incorporation of the relation (4.8) for generating real residual factors $W_j^{(k)}$ is obvious.

Theorem 6.1 (Generalization of Theorem 3.5, [37, Theorem 4.1]):

The GCARE residual matrix w.r.t. the solution $X_j^{(k)} = Z_j^{(k)} (Z_j^{(k)})^H$ produced at outer iteration step k and inner iteration step j has at most rank $2r + p$ and is given by

$$\mathcal{R}_j^{(k)} := \mathcal{R}(X_j^{(k)}) = \mathcal{L}_j^{(k)} - \Delta K_j^{(k)} (\Delta K_j^{(k)})^H = L_j^{(k)} D (L_j^{(k)})^H, \tag{6.10a}$$

where

$$\Delta K_j^{(k)} := K_j^{(k)} - K^{(k-1)} \in \mathbb{C}^{n \times r}, \quad L_j^{(k)} := [W_j^{(k)},\ \Delta K_j^{(k)}] \in \mathbb{C}^{n \times 2r+p}, \tag{6.10b}$$

$$D := \operatorname{diag}\left(I_{p+r}, -I_r\right), \tag{6.10c}$$

with $K_j^{(k)} = E^T X_j^{(k)} F$. ◊

Proof. Writing out the GCARE residual matrix and including the definitions of $K^{(k)}$, $K_j^{(k)}$ and $W_0^{(k)}$ yields

$$\begin{aligned}
\mathcal{R}(X_j^{(k)}) &= A^T X_j^{(k)} E + E^T X_j^{(k)} A - E^T X_j^{(k)} F F^T X_j^{(k)} E + C^T C \\
&= (A - F(K^{(k-1)})^H)^H X_j^{(k)} E + E^T X_j^{(k)} (A - F(K^{(k-1)})^H) \\
&\quad + K^{(k-1)} F^T X_j^{(k)} E + E^T X_j^{(k)} F (K^{(k-1)})^H - E^T X_j^{(k)} F F^T X_j^{(k)} E \\
&\quad + C^T C + K^{(k-1)} (K^{(k-1)})^H - K^{(k-1)} (K^{(k-1)})^H \\
&= (A^{(k)})^H X_j^{(k)} E + E^T X_j^{(k)} A^{(k)} + W_0^{(k)} (W_0^{(k)})^H \\
&\quad + K^{(k-1)} (K_j^{(k)})^H + K_j^{(k)} (K^{(k-1)})^H - K_j^{(k)} (K_j^{(k)})^H - K^{(k-1)} (K^{(k-1)})^H \\
&= W_j^{(k)} (W_j^{(k)})^H + [K_j^{(k)} - K^{(k-1)}][K^{(k-1)} - K_j^{(k)}]^H
\end{aligned}$$

from which (6.10) follows since $\mathcal{L}_j^{(k)} = W_j^{(k)} (W_j^{(k)})^H$ by Theorem 3.5. □

It holds $\operatorname{rank}(\mathcal{R}_j^{(k)}) = 2r + p$ only if $L_j^{(k)}$ has full column rank and there are some possibilities when this is not the case. Obviously, if $K^{(0)} = 0$, then $\operatorname{rank}(W_j^{(1)}) \leq p$ and, thus, $\operatorname{rank}(L_j^{(1)}) \leq p + r$. Also, by the same discussion as in the proof of Theorem 3.5, if $\alpha_j \in \Lambda(A^{(k)},\ E)$, $(A - \alpha_j^{(k)} E)$ is singular and $W_j^{(k)}$ can be of rank smaller than $r + p$. Of course, solving the inner GCALE equation exactly, i.e., $W_j^{(k)} (W_j^{(k)})^H = 0$, will lead to $\operatorname{rank}(\mathcal{R}_j^{(k)}) \leq r$.

The result of the above theorem can be equivalently proven by using a Taylor expansion of $\mathcal{R}(X)$ of order two [135, 52]. In contrast to the GCALE residual matrix $\mathcal{L}_j^{(k)}$, the GCARE residual matrix $\mathcal{R}_j^{(k)}$ is in general indefinite since $\Lambda(D) = \{\pm 1\}$. If $\mathcal{L}_j^{(k)} \succeq \Delta K_j^{(k)} (\Delta K_j^{(k)})^H$, it is positive semidefinite and if $\mathcal{L}_j^{(k)} = 0$, it is negative semidefinite. The outer iteration can be stopped, e.g., when

$$\varepsilon_j^{(k)} := \|\mathcal{R}_j^{(k)}\| / \|C\|^2 \leq \tau_{\mathrm{NM}} \|C\|^2, \quad 0 < \tau_{\mathrm{NM}} \ll 1. \tag{6.11}$$

This criterion can actually also be monitored during each single or every couple of steps of the inner iteration. This allows to stop G-LR-ADI when (6.11) is satisfied, regardless of the magnitude of $\mathcal{L}(X_j)$. Especially in the last outer iteration step this can lead to a reduction of the number of carried out inner iteration steps and, thus, to computational savings. Motivated by the expected quadratic convergence rate of the Newton process, we propose to start monitoring the inner GCARE residual norm when $\varepsilon^{(k-1)} \leq \sqrt{\tau_{\mathrm{NM}}}$. For a given τ_{NM}, one typically chooses $\tau_{\mathrm{ADI}} < \tau_{\mathrm{NM}}$ to ensure that the required accuracy can be achieved. From the theory of inexact Newton methods it might conceptually be possible to solve the Lyapunov equation in each outer iteration less accurately and still maintain quadratic convergence of the Newton process, see, e.g., [77, 88]. In the context of algebraic Riccati equations this appears to be less straightforward because one has to additionally ensure that the Newton method still converges to the desired stabilizing solution. This issue is subject of ongoing research initiated, e.g., by [94, 135]. In [125], a novel theoretical and numerical framework for the LR-NADI-C iteration with

inexact inner solves is established. We will not further consider inexact solves of the inner GCALEs in the sense $\tau_{\mathrm{ADI}} > \tau_{\mathrm{NM}}$.

The norm of the GCARE residual matrix can be computed very efficiently in a similar fashion as, due to the indefiniteness, for $\|\mathcal{S}_j\|$ in the fADI iteration for Sylvester equations:

$$\begin{aligned}\|\mathcal{R}_j^{(k)}\| &= \left|\lambda_{\max}\left(L_j^{(k)} D (L_j^{(k)})^H\right)\right| = \left|\lambda_{\max}\left((L_j^{(k)})^H L_j^{(k)} D\right)\right| \\ &= \left|\lambda_{\max}\left(\begin{bmatrix} \left(W_j^{(k)}\right)^H W_j^{(k)} & -\left(W_j^{(k)}\right)^H \Delta K_j^{(k)} \\ \left(\Delta K_j^{(k)}\right)^H W_j^{(k)} & -\left(\Delta K_j^{(k)}\right)^H \Delta K_j^{(k)} \end{bmatrix}\right)\right|.\end{aligned}$$

The matrix $(W_j^{(k)})^H W_j^{(k)}$ from the computation of $\|\mathcal{L}_j^{(k)}\|$ can be reused here. Hence, computing $\|\mathcal{R}_j^{(k)}\|$ boils down to finding the largest eigenvalue of a matrix of size $2r+p$. By the same reasoning as in Section 3.2.4, this is considerably less costly than estimating $\|\mathcal{R}_j^{(k)}\|$ via a Lanczos process applied to $\mathcal{R}_j^{(k)}$ or using an orthogonal factorization of $\mathcal{R}_j^{(k)}$ [42, 52] similar to (3.16) for $\mathcal{L}_j$.

6.1.3. Shift Parameter Strategies for the Inner Iteration

The inner G-LR-ADI(-r) iteration requires a set of (proper) $\{\alpha_1^{(k)}, \ldots, \alpha_J^{(k)}\} \in \mathbb{C}_-$ shift parameters corresponding to $(A^{(k)} := A - F(K^{(k-1)})^T,\ E)$. Here we discuss various strategies for their choices that are based on the approaches mentioned in Chapter 5.

Shifts Computed Before Each Inner Iteration

In Line 2 in each outer iteration step, shift parameters are exemplarily computed before the G-LR-ADI iteration is started. As proposed in [201], this could be relaxed by keeping the same sets of shifts for $k_{\mathrm{s}} > 1$ outer iterations. Moreover, the shift computation can be moved outside the Newton-loop such that only the shift parameters w.r.t. $(A^{(1)},\ E)$ are used in the whole process. This might, depending on the magnitude of the changes in $K^{(k)}$, slow down the convergence speed of the inner G-LR-ADI iteration, but save some execution time due to skipping the shift generation. Basically, all the shift strategies mentioned in Section 5.2 can be applied here. Generating the Wachspress or heuristic shifts at the beginning of each inner iteration will require linear solves with $A^{(k)}$ for obtaining the inverse Ritz values. For this the Sherman-Morrison-Woodbury formula (cf. (6.6)) can be used. Notice that even if $A = A^T$, the matrix $A^{(k)}$ is nonsymmetric such that the spectrum $\Lambda(A^{(k)},\ E)$ might contain complex eigenvalues even if $E = E^T \succ 0$. Let $E = I_n$, for simplicity, in which case

$$\begin{aligned}\mathcal{K}_h(A^{(k)},\ F^{(k)}) &= \operatorname{span}\left\{F^{(k)},\ A^{(k)}F^{(k)},\ (A^{(k)})^2F^{(k)}, \ldots, (A^{(k)})^{h-1}F^{(k)}\right\} \\ &= \operatorname{span}\left\{C^T, K^{(k-1)}, AC^T, K^{(k-1)}F^TC^T, AK^{(k-1)}, K^{(k-1)}F^TK^{(k-1)}, \ldots\right\} \\ &= \ldots = \mathcal{K}_h(A,\ F^{(k)}),\end{aligned}$$

a property that is typically referred to as *feedback invariance of Krylov subspaces*. This observation can be seen as justification for keeping the same heuristic of Wachspress

shifts generated from the Krylov subspaces w.r.t. $\mathcal{K}(A,\ F^{(k)})$ for several outer iteration steps, especially if the feedback matrices $K^{(k)}$ only change marginally in the course of the outer iteration.

Self-Generating Shifts

Motivated by the promising results with the proposed self-generating shifts in Section 5.3, we will adapt this strategies to the LR-NADI-C iteration. The self-generating $V(u)$-shift parameters can be included right away into LR-NADI-C by carrying out the projections using the matrix pair $(A^{(k)},\ E)$. The same remarks concerning the generation of stable shifts apply here as well and Corollary 4.6 generalizes as follows.

Corollary 6.2 (Generalization of Corollary 4.6):

The real low-rank solution factor $Z_j^{(k)}$ at outer iteration step k and after j inner iteration steps corresponding to a proper set of j shift parameters satisfies the GCASEs

$$(A^{(k)})^T Z_j^{(k)} - E^T Z_j^{(k)} B_{\text{ADI-r}} = W_j^{(k)} G_{\text{ADI-r}}^T, \tag{6.12a}$$

$$A^T Z_j^{(k)} - E^T Z_j^{(k)} B_{\text{ADI-r}} = W_j^{(k)} G_{\text{ADI-r}}^T + K^{(k)} B^T Z_j^{(k)}. \tag{6.12b}$$

There, $B_{\text{ADI-r}}$, $G_{\text{ADI-r}}^T$ are defined as in (4.12) but w.r.t. the complex conjugates of the used complex shifts. ◊

Proof. The result (6.12a) follows directly from Corollary 4.6 by taking care of the fact that the inner iteration is applied to $(A^{(k)})^T,\ E^T$. The definition of the closed loop matrix $A^{(k)}$ yields (6.12b). □

Similar to (5.3), no additional matrix vector products with $(A^{(k)})^T$ are necessary to build the restriction required for the $V(u)$-shift parameters.

An alternative but similar approach proposed in [23] is based on the following proposition which is established very easily.

Proposition 6.3 (The residual GCARE [23, Theorem 5(c)]):

The error $\mathcal{E}_j^{(k)} := X - X_j^{(k)}$ at the inner and outer iteration steps j and k of Algorithm 6.2 is the solution of the *residual GCARE*

$$(A_j^{(k)})^T \mathcal{E}_j^{(k)} E + E^T \mathcal{E}_j^{(k)} A_j^{(k)} - E^T \mathcal{E}_j^{(k)} C^T C \mathcal{E}_j^{(k)} E + L_j^{(k)} D (L_j^{(k)})^T = 0, \tag{6.13}$$

where $A_j^{(k)} := A - K_j^{(k)} Q^T$ and $L_j^{(k)}$, D are the factors of $\mathcal{R}_j^{(k)}$ from Theorem 6.1. ◊

The matrix pair

$$\mathcal{H}_j^{(k)} := \begin{bmatrix} A_j^{(k)} & C^T C \\ L_j^{(k)} D (L_j^{(k)})^T & -(A_j^{(k)})^T \end{bmatrix}, \quad \mathcal{G} := \begin{bmatrix} E & 0 \\ 0 & E^T \end{bmatrix} \in \mathbb{R}^{2n \times 2n}$$

is associated with (6.13). Let $\hat{U}^H \mathcal{H}_j^{(k)} \hat{Q} = \tilde{T}_1$, $\hat{U}^H \mathcal{K} \hat{Q} = \tilde{T}_2$ be the generalized Schur form of $(\mathcal{H}_j^{(k)},\ \mathcal{G})$ with $\hat{U}^H \hat{U} = \hat{Q}^H \hat{Q} = I_{2n}$. Partition the right Schur vectors as $\hat{q}_i = \begin{bmatrix} \hat{q}_i^t \\ \hat{q}_i^b \end{bmatrix}$,

$\hat{q}_i^t,\ \hat{q}_i^b \in \mathbb{C}^n$ for $i = 1 \ldots, 2n$. As Algorithm 6.2 converges, the lower left block of $\mathcal{H}_j^{(k)}$ will become smaller and smaller, and in the limit (6.13) will have a trivial solution. Hence, the norms $\|E\hat{q}_i^b\|$ for the right Schur vectors $\hat{q}_i$ corresponding to the stable eigenvalues of $(\mathcal{H},\ \mathcal{G})$ will also converge towards zero. Assuming the conditions of Theorem 2.35 hold for (6.13), then by [150, Theorem 7.2.8], $(\mathcal{H}_j^{(k)},\ \mathcal{G})$ has exactly n stable and n anti-stable eigenvalues which are also identical to the spectra $\pm\Lambda(A_j^{(k)} - FF^T\mathcal{E}_j^{(k)}E,\ E)$. Following the approach mentioned in [23, Section 3, Example 6-7], one selects the next shift α_{j+1} as the stable eigenvalue of $(\mathcal{H}_j^{(k)},\ \mathcal{G})$ for which $E\hat{q}_i^b$ has the largest norm. However, since $(\mathcal{H}_j^{(k)},\ \mathcal{G})$ is a high-dimensional matrix pair, the computation of this eigenvalue is not feasible. Instead we propose to work with the reduced matrices

$$\tilde{\mathcal{H}}_j^{(k)} := \begin{bmatrix} \tilde{A}_j^{(k)} & Q_j^T C^T C Q_j \\ Q_j^T L_j^{(k)} D (L_j^{(k)})^T Q_j & -(\tilde{A}_j^{(k)})^T \end{bmatrix}, \quad \tilde{\mathcal{G}}_j^{(k)} := \begin{bmatrix} \tilde{E}_j^{(k)} & 0 \\ 0 & (\tilde{E}_j^{(k)})^T \end{bmatrix} \tag{6.14}$$

where $\tilde{A}_j^{(k)} = Q_j^T A_j^{(k)} Q_j$, $\tilde{E}_j^{(k)} = Q_j^T E Q_j$ and Q_j are the matrices from the $V(u)$-shift approach. We will call the shift parameters obtained with this strategy $\mathcal{H}(u)$-shifts. In the first occurrence of this approach [23], all columns of the current low-rank solution factor are used.

The discussed residual norm-minimizing shifts can also be applied here on the basis of $(A^{(k)},\ E)$ and $W_j^{(k)}$. The use of the reduced objective function (5.14) constructed by, e.g., $\tilde{A}_j^{(k)}$, $\tilde{E}_j^{(k)}$ and $Q_j^T W_j^{(k)}$ from the $V(u)$-shift approach, is also advised here to lower the evaluation of the function and the gradient.

6.1.4. Accelerating the Outer Iteration by a Galerkin Projection

In [201, 50], the outer iteration of the low-rank Newton-ADI method for CAREs is accelerated by performing a Galerkin projection onto the space spanned by the current low-rank solution factor. This often leads to an impressive convergence boost where the number of outer iterations is reduced to one or two. For this, let $Q_Z^{(k)} \in \mathbb{R}^{n\times g}$ be a rectangular, orthonormal matrix for the spaces spanned by the low-rank solution factor $Z^{(k)}$ after the inner iteration in the outer iteration step k has been finished, i.e., right after the while-loop in Algorithm 6.2 is terminated. A clever orthogonalization routine should neglect nearly linearly independent columns in $Z^{(k)}$ such that $g \leq (r+p)j_{\text{it}}^{(k)}$, where $j_{\text{it}}^{(k)}$ is the number of executed inner iteration steps. Along the lines of [50] we look for an approximate solution of (6.1) of the form $X_{\text{pr}} = Q_Z^{(k)} \tilde{X} (Q_Z^{(k)})^T$ for which the residual $\mathcal{R}(X_{\text{pr}})$ satisfies a Galerkin condition. In that case, $\tilde{X} \in \mathbb{C}^{g\times g}$ has to be the solution of the projected GCARE

$$\tilde{A}^T \tilde{X} \tilde{E} + \tilde{E}^T \tilde{X} \tilde{A} - \tilde{X} \tilde{F} \tilde{F}^T \tilde{X} + \tilde{C}^T \tilde{C} = 0 \tag{6.15}$$

with the restrictions $\tilde{A}$, $\tilde{E}$ of $A,\ E$ and $\tilde{F} := (Q_Z^{(k)})^T F$, $\tilde{C} := C Q_Z^{(k)}$.

Corollary 6.2 and, in particular, the relation (6.12b) can be used to construct the restrictions without additional matrix vector products with A^T. Since the inner GCALE is

defined by $(A^{(k)})^T,\ E^T$, it is reasonable to work with the restriction $\tilde{A} = (Q_Z^{(k)})^T A^T Q_Z^{(k)}$, $\tilde{E} = (Q_Z^{(k)})^T E^T Q_Z^{(k)}$ such that the projected GCARE is given by

$$\tilde{A}\tilde{X}\tilde{E}^T + \tilde{E}\tilde{X}\tilde{A}^T - \tilde{X}\tilde{F}\tilde{F}^T\tilde{X} + \tilde{C}^T\tilde{C} = 0. \tag{6.16}$$

The projected GCARE is of much smaller dimension than (6.1). It can be solved directly by, e.g., the Newton-Kleinman method in Algorithm 6.1, the Schur vector method [55, Listing 3.4], [154, 155], structure preserving methods for the Hamiltonian eigenvalue problem [67, 63], or by the sign function iteration [191].

Once $\tilde{X}$ is computed, the next outer iteration step $k+1$ in Algorithm 6.2 starts with the updated feedback matrix $K_{\text{pr}} := E^T X_{\text{pr}} F = E^T Q_Z^{(k)} \tilde{X} \tilde{F}$. The GCARE residual $\mathcal{R}(X_{\text{pr}})$ does not have the low-rank structure (6.10) from Theorem 6.1. Hence, $\|\mathcal{R}(X_{\text{pr}})\|$ has the be computed differently, e.g., via applying a Lanczos process to $\mathcal{R}(X_{\text{pr}})$ to get an approximation of the largest eigenvalue value in magnitude. Numerical experiments [201, 50] confirm that this Galerkin acceleration indeed decreases the required number of outer iteration steps significantly. However, in contrast to the projection for the self-generating shift parameters above, computing orthonormal basis of $Z^{(k)}$ is usually more expensive because more columns have to be orthogonalized.

Remark 6.4:

In [201, 50], a very similar Galerkin projection framework is used within the inner G-LR-ADI iteration of Algorithm 6.2. We refrain from this idea because of the following reasons. At first, since orthonormal spaces for the low-rank solutions factors of the inner GCALE are required, one could have used a projection method as inner iteration from the start. Popular representatives of these methods are Krylov subspace methods, e.g., [137, 209, 218, 83, 82, 84], which produce the required orthonormal matrices anyway. Moreover, in [82, 235] it is shown that the G-LR-ADI iteration generates a rational Krylov subspace, but it works without any orthogonalization and solutions of projected GCAREs. Hence, from this viewpoint adding explicit orthogonal bases for the low-rank solution factors is not expected to give significantly better results than the plain G-LR-ADI iteration. Moreover, as mentioned in [201, 50], adding a Galerkin projection within the G-LR-ADI iteration destroys some structural properties of the method. Most notably, the associated GCALE residual does no longer have the low-rank factorization (3.15). From the view point of the used reformulated iteration (6.5), it is then not clear what to use as right hand side for the linear systems. ◊

6.1.5. Numerical Experiments

Here, we evaluate and compare the performance of the LR-NADI-C iteration regarding different shift generation strategies and the Galerkin acceleration.

As mention before we employ stopping criteria based on the scaled residual norm for both outer and inner iteration using thresholds τ_{ADI} and τ_{NM}. Alternatively, the inner and outer iteration is stopped after $j_{\max}$ or $k_{\max} = 10$ iteration steps. We use the matrices $A,\ E,\ B$ of the test examples *FDM* ($n = 122500$), *rail79k* ($n = 79841$, $r = 7$)

and *ifiss66k* ($n = 66049$), from Section 5.3.3. For example *FDM*, C in (6.1) is chosen randomly with $p = r = 5$. The *rail79k* example provides a matrix C with $p = 6$ and ϕC with $\phi = 10$ is chosen as factor of the constant term in (6.1). The scaling parameter ϕ is used to make this problem more demanding for the outer iteration as in [10]. Finally, the constant term for *ifiss66k* is set to $C = [C_1, 0]$ with a random $C_1 \in \mathbb{R}^{r\times r}$, $r = 5$. With general C, the outer Newton iteration encountered severe problems for this, but also for other examples not reported here. These difficulties could be solved by the enhancements of Algorithm 6.2 proposed in [125]. Since this would be beyond the scope of this work, these strategies are not investigated here. We restrict the discussion to the effects of different shift parameters and the outer Galerkin projection.

Influence of Different Shift Generation Strategies for the Inner Iteration

At first, we test the performance of the LR-NADI-C iteration when different shift strategies mentioned in Section 6.1.3 for the inner G-LR-ADI iteration are used. The different approaches are carried out with the same settings as in Section 5.3.3. For the heuristic and Wachspress shifts we also investigate the case when these shifts are only computed every $k_\mathrm{s} > 1$ outer iterations. Since for all examples $K^{(0)} = 0$, such that $K^{(1)}$ is the first nonzero feedback matrix, this relaxation is started when $k > 1$, i.e., new shifts are generated at the steps $k = 1,\ k_\mathrm{s},\ 2k_\mathrm{s}, \ldots$ of the outer iteration. Table 6.1 summarizes the setup parameters and the results. There, t_shift and t_total refer to the time consumed in generating shift parameters and in the complete execution of Algorithm 6.2, k_it is the number of required outer Newton steps, $j_\sum = \sum_{i=1}^{k_\mathrm{it}} j_k$ is the total number of inner G-LR-ADI iteration steps, the average over all outer steps is $j_\mathrm{avg.} = \lceil j_\sum / k_\mathrm{it} \rceil$, and $\varepsilon^{(k_\mathrm{it})}$ is the final obtained scaled norm of the GCARE residual.

Obviously, the different shift parameter approaches do not influence the outer iteration, provided they manage to steer the inner iteration towards the desired accuracy. Similar to the experiments in Section 5.3.3, the generation times t_shift of the heuristic and Wachspress shifts are considerably large portions of the complete execution times t_total. These generation times could be reduced by the setting $k_\mathrm{s} = 2$ which resulted in slightly more inner iteration steps. This effect is more pronounced for the heuristic shifts: only a slight reduction of t_total is achieved by $k_s = 2$. Since the first nonzero feedback matrix occurs in outer iteration $k = 2$, setting $k_s > 2$ is not recommended as this resulted in drastically more inner iteration steps for the outer iteration steps $1 < k < k_s$. In terms of $j_\sum$, $j_\mathrm{avg.}$, the approximate Wachspress shifts appear to be a good choice for the example *rail79k* with symmetric matrices A, E. Among the self-generating shift approaches, the $V(u)$-shift lead to the smallest generating times t_shift and also to good results w.r.t. $j_\sum$, $j_\mathrm{avg.}$ for the examples *FDM* and *ifiss66k*. For *rail79k*, *ifiss66k* they also lead to the smallest times t_total. For the *FDM* and *ifiss66k* the best performance of the inner iteration is achieved by the $V(u)$-residual norm-minimizing shifts whose generation times t_shift are, however, comparably large. This is especially the case for example *ifiss66k*. The $\mathcal{H}$-shifts managed only for example *FDM* to steer the inner iteration towards convergence. For example *rail79k*, this approach generated the same shift over and over again, whereas for *ifiss66k* the extended Hamiltonian pencil

Table 6.1.: Setup parameters and results for the examples using different shift strategies. The smallest values of t_{shift}, t_{total} and $j_{\sum}$, $j_{\text{avg.}}$ for each example are emphasized using bold letters.

Example	shift strategy	t_{shift}	t_{total}	$j_{\sum}$, $j_{\text{avg.}}$	k_{it}	$\varepsilon^{(k_{\text{it}})}$
FDM $\tau_{\text{ADI}} = 10^{-10}$, $\tau_{\text{NM}} = 10^{-9}$, $j_{\max} = 200$	heur(10, 20, 10)	40.6	1353.6	527, 176	3	$6.72{\cdot}10^{-11}$
	heur(10, 20, 10), $k_s = 2$	25.0	1335.8	527, 176	3	$6.84{\cdot}10^{-11}$
	Wachs(10^{-10}, 20, 20)	78.9	810.0	305, 102	3	$8.74{\cdot}10^{-11}$
	Wachs(10^{-10}, 20, 20), $k_s = 2$	49.9	797.1	305, 102	3	$8.91{\cdot}10^{-11}$
	$V(3)$-shifts	**5.4**	648.1	275, 92	3	$4.34{\cdot}10^{-11}$
	$\mathcal{H}(1)$-shifts	13.6	814.8	317, 106	3	$3.38{\cdot}10^{-11}$
	$V(3)$-res.min.	32.3	**557.9**	**235, 79**	3	$7.22{\cdot}10^{-11}$
rail79k $\tau_{\text{ADI}} = 10^{-10}$, $\tau_{\text{NM}} = 10^{-9}$ $j_{\max} = 100$	heur(20, 40, 40)	244.82	916.05	271, 55	5	$4.56{\cdot}10^{-11}$
	heur(20, 40, 40), $k_s = 2$	131.77	788.64	271, 55	5	$4.56{\cdot}10^{-11}$
	Wachs(10^{-10}, 20, 20)	116.12	683.18	221, 45	5	$5.09{\cdot}10^{-11}$
	Wachs(10^{-10}, 20, 20), $k_s = 2$	63.91	631.02	221, 45	5	$5.09{\cdot}10^{-11}$
	$V(3)$-shifts	**4.35**	545.25	230, 46	5	$1.14{\cdot}10^{-11}$
	$V(2)$-res.min.	20.8	**481.5**	**207, 42**	5	$9.54{\cdot}10^{-11}$
ifiss66k $\tau_{\text{ADI}} = 10^{-10}$, $\tau_{\text{NM}} = 10^{-9}$ $j_{\max} = 50$	heur(20, 30, 20)	160.3	371.4	135, 20	7	$6.36{\cdot}10^{-10}$
	heur(20, 30, 20), $k_s = 2$	89.5	400.3	194, 28	7	$6.76{\cdot}10^{-10}$
	wachs(10^{-10}, 20, 10)	87.1	341.0	160, 23	7	$6.58{\cdot}10^{-10}$
	wachs(10^{-10}, 20, 10), $k_s = 2$	48.7	305.2	162, 24	7	$6.50{\cdot}10^{-10}$
	$V(1)$-shifts	13.8	304.5	175, 25	7	$6.64{\cdot}10^{-10}$
	$V(1)$-res.min.	61.1	265.9	**122, 18**	7	$6.72{\cdot}10^{-10}$

(6.14) had exact or almost purely imaginary eigenvalues which resulted in inappropriate shift parameters. This was also observed in several further experiments not reported here. To conclude, similar recommendations regarding the shift generation approach as for GCALEs can be given for GCAREs: the approximate Wachspress shift are a good choice in case of real spectra and the $V(u)$- or $V(u)$-residual minimizing shifts otherwise.

Galerkin Acceleration of the Outer Iteration

We repeat the above experiments but perform the acceleration approach via the Galerkin projection in every outer iteration after the inner iteration satisfies the termination criterion. For this, we do not consider the simplification of the heuristic approach and the approximate Wachspress shifts with $k_{\text{s}} > 1$. The construction of the orthonormal basis matrix for the low-rank solution factor is, similar to the $V(u)$-shift approach, carried out using a singular value decomposition of $(Z^{(k)})^T Z^{(k)}$ with $Z^{(k)} \in \mathbb{R}^{n \times n_Z}$ and exploiting (6.12). All singular values with $\sigma_i/\sigma_1 < \mathbf{u}_{\text{mach.}} n_Z$ are neglected. The results are listed in Table 6.2, where now the time t_{GP} needed for performing the Galerkin projection is

Table 6.2.: Results for the previous examples using the Galerkin projection after each outer iteration. Here, t_{GP} denotes the execution time spent in performing the Galerkin projection.

Example	shift strategy	t_{GP}	t_{total}	$j_{\sum}$, $j_{\text{avg.}}$	k_{it}	$\varepsilon^{(k_{\text{it}})}$
FDM $\tau_{\text{ADI}} = 10^{-10}$, $\tau_{\text{NM}} = 10^{-9}$, $j_{\max} = 200$,	heur(10, 20, 10)	5.2	364.1	187, 187	1	$4.86{\cdot}10^{-11}$
	wachs(10^{-10}, 20, 10)	3.1	244.1	101, 101	1	$3.04{\cdot}10^{-11}$
	$V(3)$-shifts	2.8	**162.7**	89, 89	1	$3.56{\cdot}10^{-11}$
	$\mathcal{H}(1)$-shifts	3.5	214.8	106, 106	1	$2.34{\cdot}10^{-11}$
	$V(3)$-res.min.	2.4	166.2	**87, 87**	1	$4.03{\cdot}10^{-11}$
rail79k $\tau_{\text{ADI}} = 10^{-10}$, $\tau_{\text{NM}} = 10^{-9}$, $j_{\max} = 100$	heur(20, 40, 40)	1.5	92.3	51, 51	1	$1.80{\cdot}10^{-10}$
	wachs(10^{-10}, 20, 20)	3.4	87.7	**41, 41**	1	$6.94{\cdot}10^{-11}$
	$V(3)$-shifts	2.7	80.5	52, 52	1	$1.60{\cdot}10^{-10}$
	$V(2)$-res.min.	2.0	**61.9**	43, 43	1	$1.62{\cdot}10^{-10}$
ifiss66k $\tau_{\text{ADI}} = 10^{-10}$, $\tau_{\text{NM}} = 10^{-9}$, $j_{\max} = 50$	heur(20, 30, 20)	1.7	96.4	41, 21	2	$1.43{\cdot}10^{-10}$
	wachs(10^{-10}, 20, 10)	1.4	79.9	41, 21	2	$1.49{\cdot}10^{-10}$
	$V(1)$-shifts	1.2	67.8	44, 22	2	$2.75{\cdot}10^{-13}$
	$V(1)$-res.min.	1.0	**60.1**	**33, 17**	2	$9.56{\cdot}10^{-11}$

given. This includes constructing the orthonormal basis as well as solving the projected GCARE (6.15) by the `care` command. Apparently, the Galerkin projection reduces the number of required outer iteration steps down to one or two. This yields a significant reduction of the computation times compared to the results in Table 5.1. The ranking regarding the different shift approaches is mainly not changed. The clear reduction of the required number of outer iteration steps by the Galerkin projection was also observed in several other experiments, e.g., [50, 201]. The Galerkin acceleration appears to be worthwhile when the outer Newton iteration requires several steps before the quadratic convergence phase takes place which is the case for example *ifiss66k*.

Comparison to Other Low-Rank Methods for CAREs

Here, we will compare the G-LR-NADI-C (Algorithm 6.2) with two other low-rank methods for GCAREs. Because of the comparison in Section 5.3.3, we will not consider the case when the inner G-LR-ADI iteration is replaced by EKSM or RKSM since similar results can be expected. Instead, the extended block Arnoldi method for CAREs (EBA-CARE) proposed in [127] will be used. This is a straightforward modification of the EKSM for GCALEs [209], where in each iteration step a projected GCARE similar to (6.15) has to be solved. The MATLAB routine `care` is employed for this task. In the same way RKSM can be modified to deal with GCAREs [211] and we refer to this modification as RKSM-CARE. We do not include the iterative subspace method proposed in [167] as the examples there suggest that it cannot compete with RKSM-CARE although closely related. The shifts for RKSM-CARE are again computed

adaptively using the strategy in [83]. The results are summarized in Table 6.3, where dim. stands for the column dimension of the computed low-rank solution factor before any rank truncation, $t_{\text{sol.}}$, $t_{\text{orth.}}$, t_{small}, t_{shift} and t_{res} denote the times (in seconds) spent for solving the occurring linear systems, the orthogonalization processes, the solution of the projected GCAREs, the shift generation, and the residual norm computation, respectively. The total time of the method is t_{total}. As explained in Section 5.3.3, t_{total} can be slightly larger than the sum of the single times. For the G-LR-NADI-C iteration the subspace dimension is the column dimension of the low-rank solution factor in the last outer iteration. The G-LR-NADI-C iteration is carried out with Galerkin acceleration and with the shift parameters which lead to the best results in Table 6.2 w.r.t. both the execution time and required iteration steps.

Table 6.3.: Comparison between the G-LR-NADI-C iteration, EBA-CARE and RKSM-CARE.

Example	algorithm	dim.	$t_{\text{sol.}}$	$t_{\text{orth.}}$	t_{small}	t_{shift}	$t_{\text{res.}}$	t_{total}
FDM	LR-NADI-C, $V(3)$-res.min.	435	124.9	2.4	0.9	9.4	22.5	**166.2**
	EBA-CARE	910	148.0	88.2	303.7	–	0.05	541.1
	RKSM-CARE	340	101.3	37.0	19.3	13.3	7.5	179.5
rail79k	LR-NADI-C, $V(2)$-res.min.	258	44.0	1.2	0.8	4.4	7.0	61.9
	EBA-CARE	stagnation at $\varepsilon \approx 10^{-5}$						
	RKSM-CARE	186	30.5	6.0	12.6	3.9	2.5	**56.4**
ifiss66k	LR-NADI-C, $V(1)$-res.min.	170	39.5	0.9	0.02	12.7	4.6	60.1
	EBA-CARE	80	8.9	5.4	0.3	–	0.001	**15.3**
	RKSM-CARE	110	18.7	16.5	9.1	3.6	2.9	51.4

The LR-NADI-C method is only the fastest one for the example *FDM*. It is outperformed by RKSM-CARE for the other two examples, and, for *ifiss66k* also by EBA-CARE. Comparing to the GCALE results in Section 5.3.3, this is somewhat surprising because there, EKSM could not compete for the *ifiss66k* example. Apparently, the altered inhomogeneity present here leads to a significantly different behavior of EKSM. For the example *rail79k*, EBA-CARE dropped into stagnation phase which was caused by problems in solving the projected GCAREs by `care`. In fewer instances such difficulties could also be observed in RKSM-CARE which, however, still managed to produce an accurate low-rank solution. Replacing `care` by a different GCARE solver did not solve these issues. Compared to the results in Table 6.1, the LR-NADI-C iteration is far behind if the Galerkin acceleration is not used. Similar to the short discussion after the comparative experiments in Section 5.3.3 it should, on the one hand, be mentioned again that the RKSM and EKSM type approach can be further improved by basic simplifications (solving the projected GCARE only every $k_s > 1$ iterations) and the techniques proposed in [166, 84]. On the other hand, recent results in [125] promise significant improvements of the LR-NADI-C iteration by employing inexact GCALE solves of the inner iteration as well as a clever implementation of a line-search strategy [25, 26]. For

instance, we observed severe difficulties of the outer Newton iteration without these techniques for arbitrarily chosen matrices C in the *ifiss66k* example. These issues appeared to be also independent of the employed algorithm for solving the inner GCALEs.

6.2. Nonsymmetric Algebraic Riccati Equations

In this section, we consider non-symmetric algebraic Riccati equations (NARE) of the form

$$\mathcal{R}(X) = AX + XB - XFG^TX + UP^T = 0 \tag{6.17}$$

with $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{m\times m}$, $F \in \mathbb{R}^{m\times r}$, $G \in \mathbb{R}^{n\times r}$, $U \in \mathbb{R}^{n\times p}$, $P \in \mathbb{R}^{m\times p}$, and the sought solution $X \in \mathbb{R}^{n\times m}$. Applications where NAREs arise include, e.g., total least square problems [76], fluid queue models [187], the numerical treatment of transport equations [140, 163], the study of open loop Nash games [1], and methods to compute or refine invariant subspaces of matrices or pencils [78, 62]. Often, the block matrix

$$\mathcal{M} := \begin{bmatrix} B & FG^T \\ -UP^T & A \end{bmatrix}$$

is an M-matrix (cf. Definition 2.37). These MNAREs (Definition 2.37) were recently subject of extensive research in both theoretical and computational aspects, see, e.g., [57, 55, 140, 121, 122, 120, 164, 241] and the references therein. We focus on numerical methods for large-scale NAREs with low-rank matrices F, G, P and U, i.e., $r, p \ll \min\{n, m\}$. For instance, NAREs arising in transport theory [140, 163] involve matrices F, G, P, and U of low-rank. Often, the minimal, nonnegative solution $X^{(\min)} \geq 0$ which satisfies $X^{(\min)} \leq X$ for all possible solutions X of (6.17) if of interest. Recall that $\leq$, $\geq$ refer to the element wise partial orderings of matrices. By Theorem 2.38, [55, Theorem 2.9], this minimal, nonnegative solution always exists if $\mathcal{M}$ is a nonsingular, or an irreducible, singular M-matrix.

The goal of this section is, without using M-matrix properties, to present a generalization of the LR-NADI-C method (Algorithm 6.2) for large-scale NAREs which generates a low-rank approximation X_h. The factored alternating directions implicit iteration (fADI) see, e.g., Section 3.3, [43] is used to deal with the involved Sylvester equations.

6.2.1. Newton Methods for NAREs

As for CAREs and DAREs, a Newton's scheme can be applied to (6.17) which yields

$$X^{(k)} = X^{(k-1)} + N^{(k)}, \quad N^{(k)} := -(\mathcal{R}'_{X^{(k-1)}})^{-1}\mathcal{R}(X^{(k-1)}),$$

where $\mathcal{R}'_X$ is again the Fréchet derivative of $\mathcal{R}$ at X. The increment $N^{(k)}$ is the solution of the linear matrix equation

$$\mathcal{R}'|_{X^{(k-1)}}(N^{(k)}) = -\mathcal{R}(X^{(k-1)})$$

Algorithm 6.3: Newton-Kleinman method for NAREs

Input : Matrices $A,\ B,\ F,\ G,\ P,\ U$ defining (6.17), initial guesses $K^{(0)},\ H^{(0)}$, stopping tolerance $0 < \tau_{\mathrm{NM}} \ll 1$.

Output: Approximate solution X.

while $\|\mathcal{R}(X^{(k-1)})\| > \tau_{NM}\|UP^T\|$ **do**

  $\hat{F}^{(k)} := [U,\ K^{(k-1)}]$, $\hat{G}^{(k)} := [P,\ H^{(k-1)}]$.

  Solve the Sylvester equation

$$(A - K^{(k-1)}G^T)X^{(k)} + X^{(k)}(B - F\,{H^{(k-1)}}^T) + \hat{F}^{(k)}\,{\hat{G}^{(k)}}^T = 0 \tag{6.19}$$

  for $X^{(k)}$. Set $K^{(k)} := X^{(k)}F$, $H^{(k)} := (X^{(k)})^T G$.

which turns out to be a Sylvester equation. The resulting iteration for $k \geq 1$ and an initial guess $X^{(0)} \in \mathbb{R}^{n\times m}$ is

$$\mathcal{R}(X^{(k-1)}) = UP^T + AX^{(k-1)} + X^{(k-1)}B - X^{(k-1)}FG^TX^{(k-1)}, \tag{6.18a}$$

$$0 = (A - X^{(k-1)}FG^T)N^{(k)} + N^{(k)}(B - FG^TX^{(k-1)}) + \mathcal{R}(X^{(k-1)}), \tag{6.18b}$$

$$X^{(k)} = X^{(k-1)} + N^{(k)}, \tag{6.18c}$$

see, e.g. [55, Listing 3.11]. It is shown in [122, 120] that if $\mathcal{M}$ is a nonsingular or singular and irreducible M-Matrix, then the iteration (6.18) initialized with $X^{(0)} = 0$ produces a sequence of nonnegative matrices $X^{(0)} \leq X^{(1)} \leq \ldots \leq X^{(\min)}$ that converge to the minimal nonnegative solution $X^{(\min)}$. The convergence is quadratic provided $\mathcal{M}$ is a nonsingular M-matrix, where additional assumptions are needed when $\mathcal{M}$ is singular and irreducible. In some cases, the convergence may be only linear, but there are approaches that cure this remedy: one can either apply a shifting technique to the NARE or start the iteration with an appropriate initial guess X_0 [55]. For brevity we will not include these techniques as they can be easily implemented.

Similar to the Newton-Kleinman [145] and Newton-Hewer method [126] for CAREs and, respectively, DAREs, inserting $N^{(k)} = X^{(k)} - X^{(k-1)}$ into (6.18b) gives a reformulated Newton iteration for NAREs illustrated in Algorithm 6.3. There, we introduced the matrices $K^{(k)} := X^{(k)}F \in \mathbb{R}^{n\times r}$, $H^{(k)} := {X^{(k)}}^T G \in \mathbb{R}^{m\times r}$ which might be considered as analogue to the feedback matrices in the CARE / DARE case. The basic Newton iteration (6.18) and Algorithm 6.3 are mathematically equivalent if $K^{(0)} = X^{(0)}F$, $H^{(0)} = (X^{(0)})^T G$, and produce the same results in that case.

The Sylvester equations in (6.18b) and (6.19) are defined by the same coefficient but different right hand side matrices. It holds $\operatorname{rank}(\hat{F}^{(k)}(\hat{G}^{(k)})^T) \leq r + p$ in (6.19), which will enable us to formulate a low-rank version of Algorithm 6.3 provided $r + p \ll n, m$. Solving the Sylvester equation in each iteration step is the main computational effort for both algorithms. Here, we employ the factored ADI iteration (Section 3.3, [43, 32]) for this purpose since this combination has already been successfully applied for CAREs [42, 201, 135, 94] as well as DAREs [27]. For special cases of NAREs arising in transport

Algorithm 6.4: Low-rank Newton-ADI iteration for NAREs (LR-NADI-N)

Input : Matrices $A,\ B,\ F,\ G,\ U,\ P$ defining (6.17), initial guesses $K^{(0)},\ H^{(0)}$, and stopping tolerances $\tau_{\mathrm{NM}}, \tau_{\mathrm{ADI}} \ll 1$.

Output: $Z_{k_{\max}} \in \mathbb{C}^{n\times(r+p)j}$, $Y_{k_{\max}} \in \mathbb{C}^{m\times(r+p)j}$, $\Gamma_{k_{\max}} \in \mathbb{C}^{(r+p)j\times(r+p)j}$ such that $Z_{k_{\max}}\Gamma_{k_{\max}}Y_{k_{\max}}^H \approx X$.

while $\|\mathcal{R}(X^{(k-1)})\| > \tau_{NM}\|UP^T\|$ **do**

  Determine shifts $\{\alpha_1^{(k)},\dots,\alpha_J^{(k)}\}$, $\{\beta_1^{(k)},\dots,\beta_J^{(k)}\}$ w.r.t. $A^{(k)} := A - K^{(k-1)}G^T$ and $B^{(k)} := B - F\,H^{(k-1)\,H}$.

  $W_0^{(k)} = [U, K^{(k-1)}]$, $T_0^{(k)} = [P, H^{(k-1)}]$, $Z_0^{(k)} = \Gamma_0^{(k)} = Y_0^{(k)} = [\,]$, $j = 0$

  while $\|W_j^{(k)}(T_j^{(k)})^H\| > \tau_{ADI}\|W_0^{(k)}(T_0^{(k)})^H\|$ **do**

    $j = j + 1$

    Solve

$$(A^{(k)} + \beta_j^{(k)} I_n)V_j^{(k)} = W_{j-1}^{(k)}, \quad (B^{(k)} + \alpha_j^{(k)} I_m)^H S_j^{(k)} = T_{j-1}^{(k)}$$

    for $V_j^{(k)},\ W_j^{(k)}$.

    Update low-rank factors of Sylvester residual

    $W_j^{(k)} = W_{j-1}^{(k)} + \gamma_j V_j^{(k)}$, $T_j^{(k)} = T_{j-1}^{(k)} + \overline{\gamma_j} S_j^{(k)}$, $\gamma_j = -(\beta_j^{(k)} + \alpha_j^{(k)})$.

    Augment the low-rank solution factors

$$Z_j^{(k)} = [Z_{j-1}^{(k)},\ V_j^{(k)}],\ Y_j^{(k)} = [Y_{j-1}^{(k)},\ S_j^{(k)}],\ \Gamma_j^{(k)} = \operatorname{diag}\left(\Gamma_{j-1}^{(k)},\ \gamma_j I_r\right).$$

  $K^{(k)} = Z_j^{(k)}\Gamma_j^{(k)}((Y_j^{(k)})^H F)$, $H^{(k)} = Y_j^{(k)}(\Gamma_j^{(k)})^H(Z_j^{(k)})^H G$.

theory, this has also been done in [241] but here we focus on the general situation and also include the enhancements for the fADI iteration discussed in the Sections 3.3, 4.2, and 5.4.

6.2.2. Low-Rank Newton-ADI for NAREs

Applying the fADI iteration (Algorithm 3.4) to solve the Sylvester equations (6.19) in the Newton-Kleinman method for NAREs (Algorithm 6.3) directly yields the low-rank Newton-ADI for NAREs (LR-NADI-N) which is illustrated in Algorithm 6.4.

In Algorithm 6.4 we use the original fADI iteration from Algorithm 3.4 which inhibits complex arithmetic operations if some of the shifts are complex numbers and, thus, the resulting low-rank factors $Z_{k_{\max}}$, $\Gamma_{k_{\max}}$, $Y_{k_{\max}}$ are also complex. Provided the shift parameters obey the conditions set in Section 4.2, Algorithm 4.4 for generating real low-rank solution factor is also applicable here. Since the involved formulas of Algorithm 4.4 are rather long and complicated, but not crucial for this section, we keep the simpler

but possibly complex representation given in Algorithm 6.4 as well as in the upcoming investigations. In the appendix, a version of the LR-NADI-N method incorporating the fADI-r iteration (Algorithm 4.4) for generating real solution factors $Z_j^{(k)} \in \mathbb{R}^{n\times(r+p)j}$, $Y_j^{(k)} \in \mathbb{R}^{m\times(r+p)j}$, $\Gamma_j^{(k)} \in \mathbb{R}^{(r+p)j\times(r+p)j}$, as well as real $K^{(k)}$, $H^{(k)}$, can be found. This algorithm will be used in the upcoming examples whenever complex shift parameters occur.

In analogy to the discussion regarding the LR-NADI-C iteration (Algorithm 6.2) for CAREs, we will now consider some of the major steps of Algorithm 6.4 in more detail.

Solving the Linear Systems

The coefficient matrices of the linear systems in Line 6 will in general be dense, even if A and B are sparse matrices. In that case $A^{(k)} + \beta_j^{(k)} I_n$ and $B^{(k)} + \alpha_j^{(k)} I_m$ are given as a sum of a sparse matrix and a low-rank update. In complete analogy to the LR-NADI-C method, the Sherman-Morrison-Woodbury formula [111] can be used for obtaining $V_j^{(k)}$ via

$$\begin{aligned}[V_W,\ V_K] &= (A + \beta_j^{(k)} I_n)^{-1}[W_{j-1}^{(k)},\ K^{(k-1)}],\\ V_j^{(k)} &= V_W + V_K \left(I_p - P^T V_K\right)^{-1} (P^T V_W).\end{aligned}$$

The equations for generating $S_j^{(k)}$ are similar. Similar to the GCARE case, $r + 2p$ linear systems with the sparse coefficient matrices $A + \beta_j^{(k)} I_n$, $B + \alpha_j^{(k)} I_m$ have to be solved.

In certain applications, e.g. [140], the matrices A, B are given by

$$A = \Psi - st^T, \quad B = \hat{\Psi} - xy^T \tag{6.20a}$$

with diagonal matrices $\Psi \in \mathbb{R}^{n\times n}$, $\hat{\Psi} \in \mathbb{R}^{m\times m}$ and $s,\ t \in \mathbb{R}^n$, $x,\ y \in \mathbb{R}^m$. In this situation, the solutions of the linear system are, by another use of the Sherman-Morrison-Woodbury formula, given by

$$\begin{aligned}V_j^{(k)} &= \tilde{V}_S + \tilde{V}_K (I_{p+1} - [t,T]^T \tilde{V}_K)^{-1} [t,T]^T \tilde{V}_S,\\ \tilde{V}_S &:= (\Psi + \beta_j I_n)^{-1} W_{j-1}^{(k)}, \quad \tilde{V}_K := (\Psi + \beta_j I_n)^{-1}[s,K],\end{aligned} \tag{6.20b}$$

and similarly for $S_j^{(k)}$. The inversions of $\Psi + \beta_j I_n$ and $\hat{\Psi} + \alpha_j I_m$ come basically for free, leading to a very low computational effort for solving both the linear systems and, consequently, also for the overall algorithm. More precisely, the complexity for solving a linear system defined by a diagonal matrix of dimension n and r right hands sides can be described adequately by $2nr$ floating point operations. Hence, the complete algorithm will basically have a linear complexity. Tricks similar to (6.20b) have also been applied in other methods for MNAREs, see, e.g., [122, 140, 57, 172, 241]. For the case $r = p = 1$, explicit and even cheaper to compute solutions of (6.20b) are proposed in [241] which leads to a specially tailored version of Algorithm 6.4. The implementation for these special NAREs employed here uses similar techniques for solving the linear systems.

Implicit Updates of $K^{(k)}$, $H^{(k)}$

The approximate low-rank solution of the Sylvester equations is generated via

$$X_j^{(k)} = Z_j^{(k)}\Gamma_j^{(k)}(Y_j^{(k)})^H = \sum_{i=1}^{j}\gamma_i V_i^{(k)}(S_i^{(k)})^H = X_{j-1}^{(k)} + \gamma_j V_j^{(k)}(S_j^{(k)})^H$$

in each inner iteration step. It is again possible to recursively update the matrices $K^{(k)}$, $H^{(k)}$ in every inner iteration step as well:

$$\begin{aligned} K_j^{(k)} &:= X_j^{(k)}U = K_{j-1}^{(k)} + \gamma_j V_j^{(k)}(S_j^{(k)})^H U, \\ H_j^{(k)} &:= (X_j^{(k)})^H P = H_{j-1}^{(k)} + \overline{\gamma_j} S_j^{(k)}(V_j^{(k)})^H P. \end{aligned} \tag{6.21}$$

Similar to the implicit low-rank Newton-ADI method for CAREs [42, Algorithm 6], if only $K^{(k)}$, $H^{(k)}$ are of interest, accumulating and storing the low-rank solution factors in Line 9 in Algorithm 6.4 is not required which can reduce the overall storage requirements of the LR-NADI-N method.

Stopping Criteria

As in the GCARE case we stop both the inner and outer iteration using the scaled residual norm, cf. Line 4 of Algorithm 6.4.

The following theorem from [41] generalizes the Theorems 3.5, 3.14, 6.1 to the LR-NADI-N method.

Theorem 6.5 (Low-rank structure of the NARE residual [41]):

The NARE residual matrix w.r.t. the solution $X_j^{(k)} = Z_j^{(k)}\Gamma_j^{(k)}(Y_j^{(k)})^H$ produced at outer iteration step k and inner iteration step j in Algorithm 6.4 has at most rank $2r + p$ and is given by

$$\mathcal{R}(X_j^{(k)}) = \mathcal{S}(X_j^{(k)}) - \Delta K_j^{(k)}\Delta(H_j^{(k)})^H = L_j^{(k)}(D_j^{(k)})^H, \tag{6.22a}$$

where

$$\Delta K_j^{(k)} := K_j^{(k)} - K^{(k-1)} \in \mathbb{C}^{n\times r}, \quad \Delta H_j^{(k)} := H_j^{(k)} - H^{(k-1)} \in \mathbb{C}^{m\times r}, \tag{6.22b}$$

$$L_j^{(k)} := [W_j^{(k)},\ \Delta K_j^{(k)}], \quad D_j^{(k)} := [T_j^{(k)},\ \Delta H_j^{(k)}] \tag{6.22c}$$

with $K_j^{(k)}$, $H_j^{(k)}$ defined as in (6.21) and $W_j^{(k)}$, $T_j^{(k)}$ are the low-rank factors of the Sylvester residual matrix $\mathcal{S}(X_j^{(k)})$. ◊

Proof. Writing out the NARE residual and including the definitions of $K^{(k)}$, $H^{(k)}$ as well as of $K_j^{(k)}$, $H_j^{(k)}$ yields

$$\begin{aligned}
\mathcal{R}(X_j^{(k)}) =& AX_j^{(k)} + X_j^{(k)}B - X_j^{(k)}FG^TX_j^{(k)} + UP^T \\
=& (A - K^{(k-1)}G^T)X_j^{(k)} + X_j^{(k)}(B - F(H^{(k-1)})^H) \\
&+ K^{(k-1)}G^TX_j^{(k)} + X_j^{(k)}F(H^{(k-1)})^H - X_j^{(k)}FG^TX_j^{(k)} \\
&+ UP^T + K^{(k-1)}(H^{(k-1)})^H - K^{(k-1)}(H^{(k-1)})^H \\
=& A^{(k)}X_j^{(k)} + X_j^{(k)}B^{(k)} + W_0^{(k)}(T_0^{(k)})^H \\
&+ K^{(k-1)}(H_j^{(k)})^H + K_j^{(k)}(H^{(k-1)})^H - K_j^{(k)}(H_j^{(k)})^H - K^{(k-1)}(H^{(k-1)})^H \\
=& W_j^{(k)}(T_j^{(k)})^H + [K^{(k-1)} - K_j^{(k)}][H_j^{(k)} - H^{(k-1)}]^H
\end{aligned}$$

from which (6.22) follows. □

Similar discussion as for the GCARE (Section 6.1) and the CASE residual matrix (Section 3.3.2) w.r.t. a possible rank deficiency of $L_j^{(k)}$, $D_j^{(k)}$ can done here [41]. The result of the above theorem can again be equivalently proven as in [135, 52] by using a Taylor expansion of $\mathcal{R}(X)$ of order two.

For using (6.11) to stop the outer iteration, the spectral norm of $\mathcal{R}(X_j^{(k)})$ can then be computed in the same fashion as (3.45) for the Sylvester residual matrix:

$$\|\mathcal{R}(X_j^{(k)})\| = \|L_j^{(k)}(D_j^{(k)})^H\| = \sqrt{\lambda_{\max}((L_j^{(k)})^HL_j^{(k)}(D_j^{(k)})^HD_j^{(k)})}.$$

Some of the matrices from the computation of $\|\mathcal{S}_j^{(k)}\| = \|W_j^{(k)}(T_j^{(k)})^H\|$ can be stored and reused for forming $(L_j^{(k)})^HL_j^{(k)}(D_j^{(k)})^HD_j^{(k)})$. Therefore, computing $\|\mathcal{R}(X_j^{(k)})\|$ reduces to finding the largest eigenvalue of a matrix of size $2r + p$.

Shift Parameters

Two sets $\{\alpha_1^{(k)}, \ldots, \alpha_J^{(k)}\}$, $\{\beta_1^{(k)}, \ldots, \beta_J^{(k)}\}$ of shift parameters corresponding to the matrices $A^{(k)} := A - K^{(k-1)}G^T$ and $B^{(k)} := B - F(L^{(k-1)})^H$ are required by the fADI iteration. In Line 2 these shifts are, in each outer iteration step, computed before the fADI iteration is started. In case of an MNARE, after convergence of the method the spectra of the matrices $A^{(k)}$, $B^{(k)}$ will be located in the right half plane [55, Theorem 2.11.] such that it might be reasonable to neglect or negate the occasionally computed anti-stable eigenvalues of the projected matrices.

The self-generating $V/S(u)$-shift parameters and their residual norm-minimizing version from Section 5.4.2, [39] can be easily included into LR-NADI-N by performing the projections using the matrices $A^{(k)}$, $B^{(k)}$.

An approach similar to the $\mathcal{H}(u)$-shifts for the LR-NADI-C iteration is based on the following proposition which can be proved very easily along the lines of [21, Theorem 5(c)].

Proposition 6.6 (Generalization of [21, Theorem 5(c)]):

The error $\mathcal{E}_j^{(k)} := X - X_j^{(k)}$ at the inner and outer iteration steps j and k of the LR-NADI-N method (Algorithm 6.4) solves the *residual NARE*

$$A_j^{(k)}\mathcal{E}_j^{(k)} + \mathcal{E}_j^{(k)}B_j^{(k)} - \mathcal{E}_j^{(k)}FG^T\mathcal{E}_j^{(k)} + L_j^{(k)}(D_j^{(k)})^H = 0, \tag{6.23}$$

where $A_j^{(k)} := A - K_j^{(k)}G^T$ and $B_j^{(k)} := B - F(H_j^{(k)})^H$. ◊

The $m+n \times m+n$ matrix

$$\mathcal{H}_j^{(k)} := \begin{bmatrix} B_j^{(k)} & FG^T \\ L_j^{(k)}(D_j^{(k)})^H & -A_j^{(k)} \end{bmatrix}$$

is associated to (6.23). Consider the Schur form $\hat{Q}^H\mathcal{H}_j^{(k)}\hat{Q} = \tilde{T}$ of $\mathcal{H}$ with $\hat{Q}^H\hat{Q} = I_{m+n}$ and partition the Schur vectors as $\hat{q}_h = \left[\begin{smallmatrix}\hat{u}_h \\ \hat{v}_h\end{smallmatrix}\right]$, $\hat{u}_h \in \mathbb{C}^m$, $\hat{v}_j \in \mathbb{C}^n$ for $h = 1\ldots, m+n$. If Algorithm 6.4 converges, the norm of the lower left block of $\mathcal{H}_j^{(k)}$ will decrease and in the limit (6.23) will have a trivial solution. Therefore, the norms $\|\hat{v}_h\|$ and $\|\hat{u}_h\|$ corresponding to the stable and, respectively, anti-stable eigenvalues of $\mathcal{H}_j^{(k)}$ will also decrease towards zero. In particular, if (6.23) is a MNARE associated to a nonsingular M-matrix, then by [55, Theorem 2.11], $\mathcal{H}_j^{(k)}$ has exactly n stable and m anti-stable eigenvalues which are also identical to $-\Lambda(A_j^{(k)} - \mathcal{E}_j^{(k)}FG^T)$ and $\Lambda(B_j^{(k)} - FG^T\mathcal{E}_j^{(k)})$, respectively. In case of a singular M-matrix, there will be at least one zero eigenvalue in one of these spectra. In order to modify the approach mentioned in [21, Section 3, Example 6-7] for GCAREs, a straightforward idea is to choose the next shift α_{j+1} as the negative of the stable eigenvalue of $\mathcal{H}_j^{(k)}$ whose $\hat{v}_h$ has the largest norm. Likewise, the shift β_{j+1} is set to the anti-stable eigenvalue of $\mathcal{H}_j^{(k)}$ with the largest norm of $\tilde{u}_h$. Since $\mathcal{H}_j^{(k)}$ is a high-dimensional matrix, this generation strategy of α_{j+1}, β_{j+1} is too expensive, and we propose to work instead with the reduced, at most $u(r+p)$ dimensional matrix

$$\tilde{\mathcal{H}}_j^{(k)} := \begin{bmatrix} U_B^H B_j^{(k)} U_B & U_B^H FG^T U_A \\ U_A^H L_j^{(k)}(D_j^{(k)})^H U_B & -U_A^H A_j^{(k)} U_A \end{bmatrix},$$

where U_A and U_B are the orthogonal matrices associated to the $V/S(u)$-shifts. The shift parameters obtained with this strategy will be referred to as $\mathcal{H}(u)$-shifts.

Accelerating the Outer Iteration

We have seen in the experiments w.r.t. the LR-NADI-C iteration for GCAREs that performing a Galerkin projection onto the space spanned by the low-rank solution factor often yields an impressive convergence boost, where the number of outer iteration steps is reduced to one or two. This approach can also be inserted into Algorithm 6.4. For this, assume $Q_Z^{(k)} \in \mathbb{C}^{n\times g}$ and $Q_Y^{(k)} \in \mathbb{C}^{m\times h}$ are orthonormal matrices for the spaces spanned by the low-rank solution factors $Z^{(k)}$ and $Y^{(k)}$ after the inner iteration in the

outer iteration step k has been stopped. We look for an approximate solution of (6.17) of the form $X_{\rm pr} = Q_Z^{(k)} \tilde{X} (Q_Y^{(k)})^H$ for which we impose a Galerkin condition on the residual $\mathcal{R}(X_{\rm pr})$, i.e., it holds $\tilde{X} \in \mathbb{C}^{g\times h}$ solves the projected NARE

$$\tilde{A}\tilde{X} + \tilde{X}\tilde{B} - \tilde{X}\tilde{F}\tilde{G}^T\tilde{X} + \tilde{U}\tilde{P}^T = 0 \tag{6.24}$$

defined by $\tilde{A} := (Q_Z^{(k)})^H A Q_Z^{(k)}$, $\tilde{B} := (Q_Y^{(k)})^H B Q_Y^{(k)}$, $\tilde{F} := (Q_Y^{(k)})^H F$, $\tilde{G} := (Q_Z^{(k)})^H G$, $\tilde{U} := (Q_Z^{(k)})^H U$, and $\tilde{P} := (Q_Y^{(k)})^H P$. Because of the much smaller dimension of (6.24) direct methods are applicable, e.g., the Newton methods in (6.18), Algorithm 6.3, or the Schur vector method [55, Listing 3.5]. The latter method is susceptible to numerical problems when (6.24) is associated to a singular M-matrix. A modified and more stable variant of the Schur vector method proposed in [121] should be used for this situation.

With the solution $\tilde{X}$ of (6.24), the next outer iteration step $k+1$ of Algorithm 6.4 is initialized by $K_{\rm pr} := X_{\rm pr} S = Q_Z^{(k)} \tilde{X} \tilde{F}$ and $L_{\rm pr} := X_{\rm pr}^H T = Q_Y^{(k)} \tilde{X}^H \tilde{G}$. However, Theorem 6.5 does not apply to the NARE residual matrix $\mathcal{R}(X_{\rm pr})$ and, thus, $\|\mathcal{R}(X_{\rm pr})\|$ has to be computed differently, e.g., by computing an approximation of the largest singular value by a Lanczos process applied to $\mathcal{R}(X_{\rm pr})^H \mathcal{R}(X_{\rm pr})$. As before, computing the orthonormal bases for $Z^{(k)}$, $Y^{(k)}$ can become expensive. Hence, regarding the numerical costs the acceleration of the outer iteration will only pay off if the effort for the orthogonalization of $Z^{(k)}$, $Y^{(k)}$ is not too high. To achieve this one should keep the number of processed inner iteration steps small, e.g., by using high quality shift parameters.

Remark 6.7:

The presented acceleration via a Galerkin projection is a straightforward generalization of the idea in [43, Section 4], where the authors use basically the same approach for accelerating the convergence of the fADI iteration. Hence, one could also implement the Galerkin projection within the inner fADI iteration of Algorithm 6.4. We refrain from this inner Galerkin projection because of similar reasons as given in Remark 6.4 concerning the LR-NADI-C iteration. ◊

Generalized Equations

Everything discussed in this section so far can be modified to handle NAREs of the form

$$\mathcal{R}(X) = UP^T + AXC + EXB - EXFG^TXC = 0 \tag{6.25}$$

which we shall call generalized NAREs (GNARE). The matrices $E \in \mathbb{R}^{n\times n}$ and $C \in \mathbb{R}^{m\times m}$ are assumed to be nonsingular. In Algorithm 6.5 only the necessary changes in the LR-NADI-N method (Algorithm 6.4) are given which can be derived easily by following, e.g., the manipulations done in Section 3.3, [32] for generalized Sylvester equations.

6.2.3. Numerical Experiments

We evaluate the performance of the LR-NADI-N iteration, where, similar to previous experiments, the outer iteration is terminated when $\varepsilon^{(k)} := \|\mathcal{R}(X^{(k)})\|/\|UP^T\| \leq \tau_{\rm NM}$

Algorithm 6.5: Low-rank Newton-ADI iteration for GNAREs

2' Determine shifts $\{\alpha_1^{(k)}, \ldots, \alpha_J^{(k)}\}$, $\{\beta_1^{(k)}, \ldots, \beta_J^{(k)}\}$ w.r.t. the matrix pairs $(A^{(k)},\ E)$ and $(B^{(k)},\ C)$.

6' Solve $(A^{(k)} + \beta_j^{(k)} E)V_j^{(k)} = W_{j-1}^{(k)}, \quad (B^{(k)} + \alpha_j^{(k)} C)^H S_j^{(k)} = T_{j-1}^{(k)}$ for $V_j^{(k)},\ S_j^{(k)}$.

8' $W_j^{(k)} = W_{j-1}^{(k)} + \gamma_j E V_j^{(k)}$, $T_j^{(k)} = T_{j-1}^{(k)} + \overline{\gamma_j} C^T S_j^{(k)}$, $\gamma_j = -(\beta_j^{(k)} + \alpha_j^{(k)})$.

11' $K^{(k)} = E Z_j^{(k)} \Gamma_j^{(k)} ((Y_j^{(k)})^H F)$, $L^{(k)} = C^T Y_j^{(k)} (\Gamma_j^{(k)})^H (Z_j^{(k)})^H G$.

and, likewise, the inner iteration is stopped using the criterion employed in Algorithm 6.4 with τ_{ADI} or when $j_{\max}$ iteration steps have been performed. If not stated otherwise, the initial guess $X^{(0)} = 0$ is used, i.e., $K^{(0)} = 0$ and $H^{(0)} = 0$.

We use the following examples:

mnare1: As an adaptation of [55, Example 3.18] we take

$$A = \begin{bmatrix} 3 & -1 & & \\ \ddots & \ddots & \ddots & \\ & 3 & -1 & \\ -1 & & & 1.9 \end{bmatrix} \in \mathbb{R}^{n\times n}, \qquad B = \begin{bmatrix} 2 & -1 & & \\ & 3 & \ddots & \\ & & \ddots & -1 \\ -1 & & & 3 \end{bmatrix} \in \mathbb{R}^{m\times m},$$

$$U = \begin{bmatrix} U_1 \\ 0 \end{bmatrix} \in \mathbb{R}^{n\times r}, \qquad U_1 = \begin{bmatrix} -1 & -1 & & \\ & \ddots & \ddots & \\ & & -1 & -1 \\ & & & -0.9 \end{bmatrix} \in \mathbb{R}^{r\times r}, \qquad P = I_{m,r},$$

$$F = \begin{bmatrix} F_1 \\ 0 \end{bmatrix} \in \mathbb{R}^{m\times p}, \qquad F_1 = \begin{bmatrix} 1 & & & \\ 1 & 1 & & \\ & \ddots & \ddots & \\ & & 1 & 1 \end{bmatrix} \in \mathbb{R}^{p\times p}, \qquad G = I_{n,p}.$$

with $n = 30000$, $m = 20000$, $r = 3$, and $p = 5$. This example constitutes a MNARE and we use the abbreviation *mnare1*.

transp: We take the setting from [140] which inherits the structure (6.20a) with

$$\begin{aligned} &\Psi = \operatorname{diag}(\delta_1, \ldots, \delta_n), \quad \hat{\Psi} = \operatorname{diag}\left(\hat{\delta}_1, \ldots, \hat{\delta}_n\right), \\ &t = x = U = P = (q_1, \ldots, q_n)^T, \quad s = y = F = -G = \mathbf{1}_n, \\ &\delta_i = \frac{1}{c\xi_i(1+\nu)}, \quad \hat{\delta}_i = \frac{1}{c\xi_i(1-\nu)}, \quad q_i = \frac{\omega_i}{2\xi_i} \quad \text{for} \quad i = 1, \ldots, n. \end{aligned}$$

There, $0 < c \le 1$, $0 \le \nu < 1$, ξ_i, ω_i are the quadrature nodes and weights corresponding to a Gaussian quadrature on $[0, 1]$, and $\mathbf{1}_h$ denotes the column vector of length h with all entries equal to one. It obviously holds $r = p = 1$ and the other dimension is set to $n = m = 20000$. We set the other defining constants to $c = 0.5$ and $\nu = 0.3$. The relation (6.20b) is used to solve the occurring shifted linear systems. The obtained NARE is an MNARE of the structure arising in transport theory [140] and therefore abbreviated as *trans*.

Table 6.4.: Setup parameters and results for the NARE examples using different shift strategies.

Example	shift strategy	t_{shift}	t_{total}	k_{it}	$j_{\sum}$, $j_{\text{avg.}}$	$\varepsilon^{(k_{\text{it}})}$
mnare1 $\tau_{\text{ADI}} = 10^{-12}$, $\tau_{\text{NM}} = 10^{-10}$, $j_{\max} = 40$	heur(10,10)	5.41	15.34	5	77, 16	$1.44{\cdot}10^{-12}$
	$V/S(1)$-shift	0.12	5.96	5	45, 9	$1.41{\cdot}10^{-11}$
	$\mathcal{H}(1)$-shift	0.61	8.14	5	55, 11	$4.13{\cdot}10^{-13}$
	$V/S(1)$-res.min	9.02	16.60	5	59, 12	$2.06{\cdot}10^{-11}$
trans $\tau_{\text{ADI}} = 10^{-10}$, $\tau_{\text{NM}} = 10^{-9}$, $j_{\max} = 300$	heur(50,40)	12.86	22.67	3	413, 138	$9.26{\cdot}10^{-10}$
	$V/S(1)$-shift	0.36	4.24	3	230, 77	$5.30{\cdot}10^{-11}$
	$\mathcal{H}(1)$-shift	0.73	5.49	3	242, 81	$2.44{\cdot}10^{-10}$
	$V/S(2)$-res.min	11.06	13.96	3	177, 59	$6.98{\cdot}10^{-10}$
FDMNARE $\tau_{\text{ADI}} = 10^{-10}$, $\tau_{\text{NM}} = 10^{-8}$, $j_{\max} = 100$	heur(10,10)	19.30	148.66	11	408, 38	$5.23{\cdot}10^{-9}$
	$V/S(1)$-shift	0.72	113.88	11	362, 33	$2.84{\cdot}10^{-9}$
	$\mathcal{H}(1)$-shift	6.44	207.53	11	626, 57	$1.78{\cdot}10^{-9}$
	$V/S(1)$-res.min	54.63	161.03	11	345, 32	$3.88{\cdot}10^{-9}$

FDMNARE: Similar to the examples *FDM-S* in Sections 4.2.3 and 5.4.3, we take two versions of the FDM example (cf. Section 2.4). The settings for A are $n_0 = 110$, $f_1 = \mathrm{e}^{\xi_1\xi_2}$, $f_2 = \sin(\xi_1\xi_2)$, and $f_3 = (\xi_2^2 - \xi_1^2)$. Likewise, the matrix B is obtained from using $n_0 = 90$ and $f_1 = 100\mathrm{e}^{\xi_1}$, $f_2 = 10(\xi_1 + \xi_2)$, and $f_3 = \sqrt{\xi_2^1 + \xi_2^2}$. These settings are close to the Sylvester equations used in [60, Example 1]. The matrices F, G, P, U are random matrices with $p = 5$ and $r = 10$ columns of uniformly distributed entries. This entirely for testing purposes constructed example will be referred to as *FDMNARE*.

Influence of Different Shift Generation Strategies

At first, we test the performance of LR-NADI-N for these examples when different shift strategies for the inner fADI iteration are used. We employ the heuristic Ritz value based shifts [162, 43], the $V/S(u)$-, and $V/S(u)$-residual norm-minimizing shifts from Section 5.4.2, [39], as well as the $\mathcal{H}(u)$-shifts introduced above. The optimization problem for the residual norm-minimizing shift dealt with the same settings as in Section 5.4.2, except that the active-set algorithm is used within `fmincon`.

Other approaches, e.g., the one proposed in [202, Algorithm 2.1] or the `parsyl`[1] routine provided in [233] were not able to compete with those strategies. The setup parameters as well as the results are summarized in Table 6.4. There, heur(k_+, k_-) refers to the heuristic shifts [162, 43] which are generated by using $k_+ = k_+^A = k_+^B$ and $k_- = k_-^A = k_-^B$ Ritz and inverse Ritz values w.r.t. $A^{(k)}$ and $B^{(k)}$. As before also the total and average numbers, $j_{\sum}$ and $j_{\text{avg.}} = \lceil j_{\sum}/k_{\text{it}} \rceil$, of inner iteration steps are given.

For all examples, the LR-NADI-N algorithm is able to compute low-rank solutions of the large-scale NAREs in a small amount of time. Apparently, as for the LR-NADI-C

[1] Available at `http://extras.springer.com/2013/978-1-4614-5121-1`.

method for GCAREs, the progress of the outer Newton iteration is largely not effected by the different shift generation strategies, as is it can be seen by the constant number of required outer iteration steps k_{it}.

For all examples, the $V/S(u)$-shifts lead to the smallest executions times t_{total}, closely followed by the $\mathcal{H}(u)$-shifts.

For the *trans* example, the residual norm-minimizing shifts lead to the smallest number of inner iterations, but due to their computationally more demanding generation, the resulting time t_{total} is larger. The heuristic shift approach required high numbers $k_+,\ k_-$ of Ritz values in this example to provide convergence of the inner iteration within $j_{\max}$ steps. Although the generation of the heuristic shifts uses (6.20b), the generation time is larger than for the other shift approaches. We point out that for this MNARE, the obtained low-rank approximations

$$X^{(k)} = Z^{(k)}_{j^{(k)}_{\text{it}}} \Gamma^{(k)}_{j^{(k)}_{\text{it}}} \big(Y^{(k)}_{j^{(k)}_{\text{it}}}\big)^T, \quad k = 1, \ldots, k_{\text{it}}$$

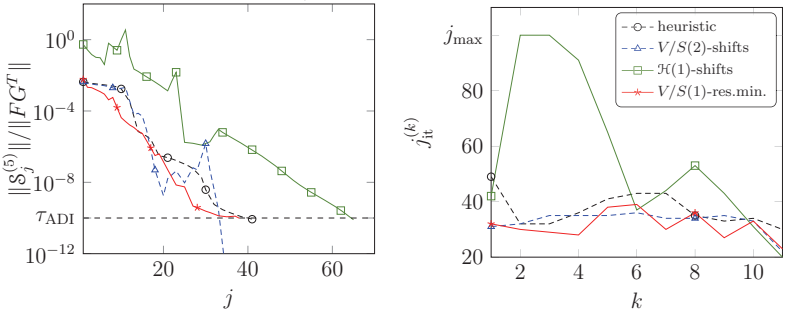
were indeed nonnegative matrices, regardless of the used shift strategy. Hence, as for the Newton methods (6.18) and Algorithm 6.3, the LR-NADI-N iteration produces a sequence of nonnegative approximations. The low-rank solution factors $Z^{(k)}_{j^{(k)}_{\text{it}}}$, $\Gamma^{(k)}_{j^{(k)}_{\text{it}}}$, $Y^{(k)}_{j^{(k)}_{\text{it}}}$ alone were, however, not nonnegative.

The example *FDMNARE* appears to be the hardest one for the LR-NADI-N method resulting in the largest number of outer iteration steps k_{it}. The usage of an appropriately chosen initial guess $X^{(0)}$ could be of great value. Here, the heuristic shifts lead to a similar progress of the inner iteration compared to the $V/S(u)$-shifts. The latter ones can, however, be generated cheaper such that the computation time t_{total} is still smaller than for the heuristic shifts. Moreover, the optimization process for residual norm-minimizing shifts experienced difficulties very similar to those mentioned in Section 5.4.3. We observe that the $V/S(u)$- and $\mathcal{H}(u)$-shifts lead in some outer iteration steps to an oscillatory behavior of the inner Sylvester residual norm. The observation is similar to the experiments regarding the G-fADI iterations for GCASEs in Section 5.4.3 (cf. Figure 5.5) and [32, Section 3.6]. The $\mathcal{H}(u)$-shifts cannot compete with the other approach since they had difficulties to steer the inner iteration towards convergence within $j_{\max}$ steps. This was especially apparent in the early stage of the outer iteration when the approximate solution is still too far from the real solution. Hence, the $\mathcal{H}$-shifts lead to the largest number of total inner iterations j_{it} as well as the highest total execution times t_{total}.

In Figure 6.1, we illustrate these issues, where the left plot shows the progress of the scaled inner Sylvester residual norm at the outer iteration step $k = 5$ for all three shift approaches. The right picture shows the number of total required inner iteration steps $j^{(k)}_{\text{it}}$ against the outer iteration step k as the outer iteration proceeds.

From the given results in Table 6.4 we suggest to use the $V/S(u)$-shifts unless better strategies are available. This choice appears to be a good trade-off between shift generation effort and the achieved speed of the inner iteration. As for the GCASE experiments in Section 5.4.3, the optimization process of the residual norm-minimizing shift should be further enhanced.

Figure 6.1.: Problematic behavior of the inner iteration for the example *FDMNARE*. Left plot: History of scaled Sylvester residual norm $\|\mathbb{S}_j^{(5)}\|/\|FG^T\|$ of the fADI iteration at the fifth outer iteration step. Right plot: number of required inner iteration steps $j_{\text{it}}^{(k)}$ against outer iteration index k.



Effect of the Galerkin Acceleration in the Outer Iteration

We repeat the above examples, but employ the Galerkin acceleration after each outer iteration step. The small, projected NAREs (6.24) are dealt with by the basic Newton-Kleinman method (Algorithm 6.3). The settings w.r.t. stopping criteria and the shift parameter generation are kept unchanged. The results are given in Table 6.5. It is again remarkable that, similar to the LR-NADI-C iteration for GCAREs (Section 6.1.5, [50]), the Galerkin projection reduces the number of required outer iteration steps down to one or two. This leads to a reduction of the computation time compared to the results in Table 6.4. For the example *trans* the already fast outer iteration is only accelerated by one iteration step which does not yield a huge reduction of t_{total} because of the numerical costs of the required orthogonalization procedure and the extraordinarily cheap solution of the linear systems by (6.20b). In summary, the Galerkin acceleration appears to be a worthwhile strategy, especially when the outer Newton iteration requires several steps before the quadratic convergence is reached, which is the case for example *FDMNARE*.

Comparison with Other Methods

Here we briefly compare the LR-NADI-C method (Algorithm 6.4) to two existing approaches for large-scale NAREs. The first method is the Newton type algorithm proposed in [57] which is intrinsically designed for problems of the form (6.20). The second competitive method is the structure preserving doubling algorithm given in [164], where the implementation we gratefully received from the authors is also only applicable to problems of the form (6.20). Hence, only the example *trans* can be used for a comparison with these two approaches. The LR-NADI-N iteration is carried out with the same stopping criteria as before, the $V/S(1)$-shifts, and with the Galerkin projection.

The Newton type method by [57] led to a residual norm of $8.63 \cdot 10^{-11}$ after 4 iteration

Table 6.5.: Results for the examples with Galerkin projection in each outer iteration step. Here, the execution time spent in performing the Galerkin projection is denoted by t_{GP}.

Example	shift strategy	t_{GP}	t_{total}	k_{it}	$j_{\sum}$, $j_{\mathrm{avg.}}$	$\varepsilon^{(k_{\mathrm{it}})}$
mnare1 $\tau_{\mathrm{ADI}} = 10^{-12}$, $\tau_{\mathrm{NM}} = 10^{-10}$, $j_{\max} = 40$	heur(10,10)	0.08	1.83	1	11, 11	$1.44{\cdot}10^{-12}$
	$V/S(1)$-shift	0.06	2.00	2	17, 9	$1.41{\cdot}10^{-11}$
	$\mathcal{H}(1)$-shift	0.08	1.09	1	10, 10	$4.13{\cdot}10^{-13}$
	$V/S(1)$-res.min	0.06	5.61	2	21, 11	$2.06{\cdot}10^{-11}$
trans $\tau_{\mathrm{ADI}} = 10^{-10}$, $\tau_{\mathrm{NM}} = 10^{-9}$, $j_{\max} = 300$	heur(10,10)	0.84	15.12	2	265, 133	$9.26{\cdot}10^{-10}$
	$V/S(1)$-shift	0.25	3.08	2	145, 73	$5.30{\cdot}10^{-11}$
	$\mathcal{H}(1)$-shift	0.33	2.88	2	142, 71	$2.44{\cdot}10^{-10}$
	$V/S(2)$-res.min	0.36	8.61	2	119, 60	$6.98{\cdot}10^{-10}$
FDMNARE $\tau_{\mathrm{ADI}} = 10^{-10}$, $\tau_{\mathrm{NM}} = 10^{-8}$, $j_{\max} = 100$	heur(10,10)	1.36	21.84	2	79, 40	$5.23{\cdot}10^{-9}$
	$V/S(2)$-shift	0.71	5.13	1	31, 31	$2.84{\cdot}10^{-9}$
	$\mathcal{H}(1)$-shift	0.96	7.60	1	42, 42	$1.78{\cdot}10^{-9}$
	$V/S(1)$-res.min	0.70	19.37	2	56, 28	$3.88{\cdot}10^{-9}$

steps and approximately 256 seconds. It required around 3 GB of memory during its runtime. For this algorithm we used the original Fortran 90 implementation provided by the authors compiled (using gfortran-4.8.2 and the compiler options -O2, -march=native) with `netlib.org` reference implementations of LAPACK95, LAPACK, and BLAS. We expect even better results with optimized LAPACK and BLAS, especially when all CPU Cores are exploited. However, the gain to be expected from such an optimization is unlikely to make up for the performance gap to the LR-NADI-N iteration.

After roughly one hour of computation time, the doubling algorithm [164] did not manage to provide an approximate solutions of the desired accuracy. The major part of time was spent in the necessary rank truncation steps and in evaluating the inherent recursive computations. It also required significantly more memory compared to Algorithm 6.4.

Consequently, compared to the timings in Tables 6.4 and 6.5, the LR-NADI-N iteration is clearly superior to both alternative methods for large-scale NAREs.

6.3. Conclusions

We investigated the numerical solution for large-scale GCARES and NAREs by a low-rank Newton iteration. Motivated by similar approaches for linear matrix equations, the core idea is to approximate the ARE solution by a factorization of very low rank. In each step of the Newton iteration, a large Lyapunov or Sylvester equation has to be solved, for which we employed the relevant low-rank ADI iterations from Chapter 3, [183, 161, 162, 43]. The improvements discussed in Chapters 3–5, [38, 36, 37, 39, 32]

were incorporated into this combination of Newton scheme and low-rank ADI iteration which lead to improved low-rank Newton-ADI methods for AREs [42, 201, 135, 94, 49, 50, 10, 52, 41]. This allowed, for instance, to express the residual of the ARE w.r.t. the approximate solution at any stage of the iterative process explicitly as low-rank factorization which allows the cheap computations of its norm. Since the performance of the low-rank ADI iteration heavily relies on the shift parameters, we adapted existing strategies known for the inner ADI iteration [183, 43, 39], but also approaches which are closer connected to AREs [21]. An acceleration strategy for the Newton iteration based on a Galerkin projection [50] has also been presented and shows remarkable speedups in the execution time at essentially no accuracy loss. With this acceleration, the proposed low-rank Newton-ADI methods are competitive to other, existing low-rank algorithms for AREs, e.g., [127, 211, 164]. In the NARE case the proposed low-rank Newton-ADI iteration was able to highly outperform existing methods for large NAREs [57, 164]. As for the the low-rank ADI iteration for GCALEs and GCASEs, although the employed shift parameters also worked satisfactory here, there is still room for improvement. Apart from the shift parameters, further research effort should also be put w.r.t. speeding up the Newton iteration, e.g., by the promising techniques in [125]. Carrying over the improvements of the G-LR-ADI iteration to the low-rank qADI iteration [238, 21], which are specially tailored low-rank ADI type methods directly applicable to GCAREs, is currently investigated [24].

CHAPTER 7

APPLICATIONS TO MODEL ORDER REDUCTION

Contents

7.1. Concepts and Goals of Model Order Reduction

Linear dynamical control systems of the form (2.5) are used in many fields of applications, e.g., for simulation and stabilization purposes. In many applications the still increasing demand for more realistic models can lead to large system order n, i.e., the size of $x(t)$ and the leading coefficients A, E. For instance, (2.5) can be the result of a spatial discretization of a partial differential equation. These large degrees of freedom yield high computational effort for solving the differential equation in (2.5). Model order reduction is the process of approximating the original large-scale system by a system

having a drastically reduced state space dimension that accurately describes the dynamical behavior of the original full-size system. These reduced systems, called reduced order models, enable more cost efficient simulations.

Most model order reduction approaches can be interpreted as a Petrov-Galerkin style projection using $x(t) \approx T_R\tilde{x}(t),\ \tilde{x} \in \mathbb{R}^r$ with $r \ll n$, and a right projection matrix $T_R \in \mathbb{C}^{n\times r}$ whose columns span a low-dimensional subspace $\mathcal{T}_R \subset \mathbb{C}^n$. In the following, let $x(0) = 0$. The imposed Petrov-Galerkin condition [3] is

$$E\tilde{x}(t) - A\tilde{x}(t) - Bu(t) \perp \mathcal{T}_L \tag{7.1}$$

with a second low-dimensional subspace $\mathcal{T}_L \subset \mathbb{C}^n$. Using a basis matrix $T_L \in \mathbb{C}^{n\times r}$ for $\mathcal{T}_L$ and adding the appropriately projected output equation, (7.1) leads to the reduced order model

$$\tilde{E}\tilde{x}(t) = \tilde{A}\tilde{x}(t) + \tilde{B}u(t), \quad \tilde{y}(t) = \tilde{C}\tilde{x}(t), \tag{7.2}$$

defined by the reduced matrices $\tilde{E} := T_L^H E T_R$, $\tilde{A} := T_L^H A T_R \in \mathbb{C}^{r\times r}$, $\tilde{B} := T_L^H B \in \mathbb{C}^{r\times m}$, $\tilde{C} := CT_R \in \mathbb{C}^{p\times r}$. Since $r \ll n$, the reduced system (7.2) can be simulated with much less computational effort. Of course, $\tilde{y}(t)$ should approximate the original output $y(t)$ accurately, such that the error $\tilde{y} - y$ is small in an appropriate norm.

Let $x(s),\ u(s),\ y(s)$ denote the Laplace transforms of $x(t),\ u(t),\ y(t)$. Then $y(s) = H(s)u(s)$, where

$$H(s) = C(sE - A)^{-1}B \in \mathbb{C}^{p\times m},\ s \in \mathbb{C} \tag{7.3}$$

is the transfer function matrix of (2.5). The $\mathcal{H}_\infty$-norm of (7.3) is given by

$$\|H\|_{\mathcal{H}_\infty} = \sup_{\omega\in\mathbb{R}_+} \sigma_{\max} H(\jmath\omega) = \sup_{\omega\in\mathbb{R}_+} \|H(\jmath\omega)\|_2. \tag{7.4}$$

If $\tilde{H}$ is the transfer function matrix of (7.2) it holds [44, Section 1.2] that

$$\|\tilde{y} - y\|_2 \le \|H - \tilde{H}\|_{\mathcal{H}_\infty}\|u\|_2 \tag{7.5}$$

such that finding a small error $\tilde{y} - y$ can be recast into obtaining a small error $H - \tilde{H}$ in the $\mathcal{H}_\infty$-norm. Due to the Paley-Wiener theorem [240], (7.5) holds in time as well as frequency domain, and $\|\cdot\|_2$ stands for the corresponding 2-induced norm w.r.t. the function spaces for u, y. The preservation of important system properties, e.g., stability ($\Lambda(\tilde{A}, \tilde{E}) \subset \mathbb{C}_-$), is also an often desired goal.

Different model order reduction approaches [3, 205, 44] differ in the way the subspaces $\mathcal{T}_R,\ \mathcal{T}_L$, or more precisely, the projection matrices $T_R,\ T_L$, are generated. Here, we will exclusively consider model reduction via the balanced truncation [221, 175] framework which is topic of the next section. There, we will investigate the usage of the G-LR-ADI iteration for computing low-rank approximations of the involved Lyapunov equations. We also discuss some stopping criteria adapted to the use in this model reduction context. A modification of balanced truncation for second order systems (2.32) is also briefly

explained. The presented balanced truncation framework is intrinsically designed to find reduced systems that are accurate for all values of ω in (7.5). In Section 7.3, a version of balanced truncation is investigated where this goal is relaxed to ω from smaller intervals $\Omega \subset \mathbb{R}_+$. This restricted form of BT, referred to as frequency-limited BT, will lead to different Lyapunov equations which incorporate also matrix valued functions, which makes computing approximate low-rank solutions more demanding. Specialized methods for computing approximate low-rank solutions of these equations are developed after a short investigation concerned with the eigenvalue decay of their solution.

7.2. Balanced Truncation Model Order Reduction

Balanced truncation (BT) introduced in [221, 175] has shown to be a reliable system theoretic method for model order reduction (MOR), see, e.g., [3, 205, 44]. Consider the generalized state space system (2.5), where $E,\ A \in \mathbb{R}^{n\times n}$, E is assumed to be invertible, and $\Lambda(A,\ E) \subset \mathbb{C}_-$, i.e., the system $(E; A, B, C)$ is asymptotically stable.

The key ingredients of BT are the reachability and observability Gramians [72, 129, 3] which are given by

$$P = \int_0^\infty (\exp\left(E^{-1}At\right)E^{-1}BB^TE^{-T}\exp\left(E^{-1}At\right)^T \mathrm{d}t, \tag{7.6a}$$

$$E^TQE = E^T\int_0^\infty (\exp\left(E^{-1}At\right)^T C^TC\exp\left(E^{-1}At\right)\mathrm{d}tE. \tag{7.6b}$$

Recalling Lemma 2.25, P, Q solve

$$APE^T + EPA^T = -BB^T, \tag{7.7a}$$

$$A^TQE + E^TQA = -C^TC, \tag{7.7b}$$

which we also refer to as reachability and observability GCALEs. The magnitude of the eigenvalues of P and E^TQE constitutes a measure of how well states can be controlled and, respectively, observed. Following this reasoning, the square roots of singular values σ_i of the product of both Gramians, PE^TQE, represents a joint measure for controllability and observability. The σ_i are also called *Hankel singular values* of the system and are system invariants w.r.t. state space transformations. The aim of balanced truncation [221, 175] is to identify and truncate components that are weakly controllable and observable. This is achieved by a state space transformation via a nonsingular matrix $T \in \mathbb{R}^{n\times n}$ that simultaneously diagonalizes both Gramians through congruence, i.e.,

$$T^TPT = T^{-1}E^TQET^{-T} = \Sigma = \mathrm{diag}\,(\sigma_1, \ldots, \sigma_n)$$

and, hence, it holds for their product $T^TPE^TQET^{-T} = \Sigma^2$. In other words, the system is brought into a balanced realization. This balanced realization allows to easily identify weakly controllable and observable state components which correspond to diagonal

entries of Σ^2 that are very small in magnitude. As the system is assumed to be stable and, thus, P and E^TQE are positive (semi-)definite, there exist Cholesky(-like) factorizations $P = Z_P Z_P^T$ and $Q = Z_Q Z_Q^T$. The balancing transformation is then easily found by computing the singular value decomposition $Z_Q^T E Z_P = R\Sigma L^T$, $R^T R = L^T L = I_n$ and defining $T := Z_P L \Sigma^{-\frac{1}{2}}$. It can be shown that the inverse of T can be written as $T^{-1} = Z_Q R \Sigma^{-\frac{1}{2}}$ provided $P,\ E^TQE > 0$ which holds if (2.5) is controllable and observable (cf. Definition 2.21, Lemma 2.25). The truncation is carried out by using only the r largest singular values $\sigma_1, \ldots, \sigma_r$ and corresponding singular vectors for constructing T and T^{-1}. This procedure, often referred to as the *square-root balanced truncation* (SRBT) method, was introduced in [221, 156]. One can show [3] that the reduced system generating by balanced truncation is, like the original system, asymptotically stable and that the error is bounded by

$$\|H - \tilde{H}\|_{\mathcal{H}_\infty} = \max_{\omega \in \mathbb{R}} (\|H(\jmath\omega) - \tilde{H}(\jmath\omega)\|_2) \leq 2 \sum_{j=r+1}^{n} \sigma_j \tag{7.8}$$

without taking the multiplicities of the σ_j into account. The error bound (7.8) admits to automatically determine the reduced dimension r, for instance, one could adjust r such that

$$2 \sum_{j=r+1}^{n} \sigma_j \leq \tau_{\mathrm{BT}}, \tag{7.9}$$

where $0 < \tau_{\mathrm{BT}} \ll 1$ indicates the desired accuracy of the reduced order model. For large-scale systems with $m,\ p \ll n$, the Gramians are typically approximated by low-rank factorizations $P \approx Z_P Z_P^T$, $Q \approx Z_Q Z_Q^T$ with $Z_P \in \mathbb{R}^{n \times r_1}$, $Z_Q \in \mathbb{R}^{n \times r_2}$, $r_1,\ r_2 \ll n$ which can be computed with suitable low-rank algorithms. In that case the GCALEs (7.7a), (7.7b) are only solved approximately and one occasionally speaks of the low-rank or approximate square-root balanced truncation method which is summarized in Algorithm 7.1. By construction of the transformations, it always holds that $\tilde{E} = T_L^T E T_R = I$ and, moreover, LR-SRBT constitutes a Petrov-Galerkin model order reduction method. It is important to note that both the stability preservation as well as the error bound (7.8) are intrinsically only proven for the case when Z_P, Z_Q are exact solution factors, i.e. $Z_P Z_P^T = P$, $Z_Q Z_Q^T = Q$ solving the GCALEs in (7.6) exactly. The effects of using inexact Gramians in balanced truncation have been investigated in, e.g., [119, 3, 118, 237], but to the author's knowledge, precise ramifications of this inexactness are still not well understood.

The next section is concerned with the computation of Z_P, Z_Q by a variant of the G-LR-ADI iteration.

7.2.1. Numerical Solution of the Gramians with the Dual LR-ADI Iteration

Obviously, the G-LR-ADI iteration can be readily applied to solve (7.7a) and (7.7b) in two separate runs. However, one can also establish a variant for a simultaneous solution,

Algorithm 7.1: Square root balanced truncation for generalized state-space systems using low-rank factors (LR-SRBT)

Input : System matrices E, A, B, C defining the dynamical system (2.5), truncation tolerance $\varepsilon_{\mathrm{BT}}$

Output: Matrices $\tilde{E}$, $\tilde{A}$, $\tilde{B}$, $\tilde{C}$ of reduced system

1 Compute low-rank solution factors Z_P, Z_Q of the solutions of (7.7) such that $P \approx Z_P Z_P^T$, $Q \approx Z_Q Z_Q^T$.

2 Compute and partition a (thin) singular value decomposition

$$R\Sigma L^T = \begin{bmatrix} R_1 & R_2 \end{bmatrix} \begin{bmatrix} \Sigma_1 & 0 \\ 0 & \Sigma_2 \end{bmatrix} \begin{bmatrix} L_1 & L_2 \end{bmatrix}^T = Z_Q^T E Z_P, \tag{7.10}$$

where Σ_1 contains the largest (approximate) Hankel singular values.

3 Construct transformation matrices T_R and T_L

$$T_R := Z_P L_1 \Sigma_1^{-\frac{1}{2}}, \quad T_L := Z_Q R_1 \Sigma_1^{-\frac{1}{2}}. \tag{7.11}$$

4 Generate reduced order model

$$\tilde{E} := T_L^T E T_R, \quad \tilde{A} := T_L^T A T_R, \quad \tilde{B} := T_L^T B, \quad \tilde{C} := C T_R. \tag{7.12}$$

motivated by the adjoint nature of both GCALEs (7.7). In the following, the subscripts $_P$ and $_Q$ denote quantities that belong to the iteration for P and, respectively, Q. Recall that for the reachability GCALEs (7.7a), the G-LR-ADI iteration proceeds as

$$V_{P,j} = \left(A + \alpha_j E\right)^{-1} W_{P,j-1}, \tag{7.13}$$

$$W_{P,j} = W_{P,j-1} - 2\,\mathrm{Re}\,\left(\alpha_j\right) E V_{P,j}, \tag{7.14}$$

$$Z_{P,j} = [Z_{P,j-1},\ \sqrt{-2\,\mathrm{Re}\,\left(\alpha_j\right)} V_{P,j}], \tag{7.15}$$

where $W_{P,0} := B$. If we now use the same shifts $\alpha_1, \ldots, \alpha_j$ for the observability GCALE (7.7b), then this translates to

$$V_{Q,j} = \left(A^T + \alpha_j E^T\right)^{-1} W_{Q,j-1}, \tag{7.16a}$$

$$W_{Q,j} = W_{Q,j-1} - 2\,\mathrm{Re}\,\left(\alpha_j\right) E^T V_{Q,j}, \tag{7.16b}$$

$$Z_{Q,j} = [Z_{Q,j-1},\ \sqrt{-2\,\mathrm{Re}\,\left(\alpha_j\right)} V_{Q,j}] \tag{7.16c}$$

with $W_{Q,0} := C^T$. As before, we assume that the set of shifts is proper, such that a complex shift is followed by its complex conjugate. In that case, interchanging α_j

Algorithm 7.2: The dual G-LR-ADI iteration [33]

Input : System matrices E, A, B, C defining (7.7), shift parameters $\{\alpha_1, \ldots, \alpha_{j_{\max}}\} \subset \mathbb{C}_-$, and tolerance $0 < \tau_{\mathcal{L}} \ll 1$

Output: $Z_P = Z_{P,j_{\max}} \in \mathbb{R}^{n \times m j_{\max}}$, $Z_Q = Z_{Q,j_{\max}} \in \mathbb{R}^{n \times p j_{\max}}$ such that $Z_P Z_P^T \approx P$, $Z_Q Z_Q^T \approx Q$.

1 $W_{P,0} = B$, $W_{Q,0} = C^T$ $Z_{P,0} = Z_{Q,0} = [\,]$, $j = 1$.

2 **while** $\|(W_{P,j-1})^T W_{P,j-1}\| \geq \tau_{\mathcal{L}} \|B^T B\|$ *or* $\|(W_{Q,j-1})^T W_{Q,j-1}\| \geq \tau_{\mathcal{L}} \|CC^T\|$ **do**

3 Solve

$$(A + \alpha_j E) V_{P,j} = W_{P,j-1}, \quad (A + \alpha_j E)^H V_{Q,j} = W_{Q,j-1} \tag{7.17}$$

for $V_{P,j}$, $V_{Q,j}$.

4 **if** $\mathrm{Im}\,(\alpha_j) = 0$ **then**

5 $[W_{P,j},\ W_{Q,j}] = [W_{P,j-1},\ W_{Q,j-1}] - 2\,\mathrm{Re}\,(\alpha_j)\,[EV_{P,j},\ E^T V_{Q,j}]$.

6 $Z_{P,j} = [Z_{P,j-1},\ \sqrt{-2\alpha_j} V_{P,j}]$, $Z_{Q,j} = [Z_{Q,j-1},\ \sqrt{-2\alpha_j} V_{Q,j}]$.

7 $j = j + 1$.

8 **else**

9 $\gamma_j = 2\sqrt{-\mathrm{Re}\,(\alpha_j)}$, $\delta_j = \frac{\mathrm{Re}\,(\alpha_j)}{\mathrm{Im}\,(\alpha_j)}$.

10 $[W_{P,j+1}, W_{Q,j+1}] =$
$[W_{P,j-1}, W_{Q,j-1}] + \gamma_j^2 \Big(\mathrm{Re}\,\big([EV_{P,j}, E^T V_{Q,j}]\big) + \delta_j\,\mathrm{Im}\,\big([EV_{P,j}, -E^T V_{Q,j}]\big)\Big)$.

11 $Z_{P,j+1} = \big[Z_{P,j-1},\ \gamma_j\big(\mathrm{Re}\,\big(V_{P,j}\big) + \delta_j\,\mathrm{Im}\,\big(V_{P,j}\big)\big),\ \gamma_j\sqrt{(\delta_j^2+1)}\cdot\mathrm{Im}\,\big(V_{P,j}\big)\big]$.

12 $Z_{Q,j+1} = \big[Z_{Q,j-1},\ \gamma_j\big(\mathrm{Re}\,\big(V_{Q,j}\big) - \delta_j\,\mathrm{Im}\,\big(V_{Q,j}\big)\big),\ \gamma_j\sqrt{(\delta_j^2+1)}\cdot\mathrm{Im}\,\big(V_{Q,j}\big)\big]$.

13 $j = j + 2$.

and $\overline{\alpha_j}$ in the iteration (7.16) yields $\big(A^T + \overline{\alpha_j} E^T\big)^{-1} = \big(A + \alpha_j E\big)^{-H}$. The relations for $V_{P,j+1}$, $W_{P,j+1}$, $V_{Q,j+1}$, $W_{Q,j+1}$ w.r.t. a pair of complex conjugated shifts can be easily deduced from (4.8b). The resulting dual G-LR-ADI iteration for simultaneously solving both GCALEs (7.7) is illustrated in Algorithm 7.2. As for a single GCALE, the most expensive step is the solution of the linear systems of equations in Line 3. Since the linear systems are adjoint to each other one can, similar to the fADI iteration (3.48) for cross-Gramian equations (3.47), compute a LU factorization of $(A + \alpha_j E) = L_j U_j$ in the P-iteration and reuse it in the Q-iteration via $(A + \alpha_j E)^H = U_j^H L_j^H$, i.e., $V_{P,j} = U_j^{-1}(L_j^{-1} W_{P,j-1})$, $V_{Q,j} = L_j^{-H}(U_j^{-H} W_{Q,j-1})$. As we have seen in Section 3.3.4, solving both linear system separately, e.g., by the MATLAB backslash, can be faster than computing and storing the triangular LU factors. Again, which way is faster appears to be highly dependent on the problem and the computing environment. In the case of a separate solution there is no necessity to use the same set of shift parameters for both the P- and Q-iteration, although generating a single set might save some computations.

For symmetric matrices $A = A^T$ and $E = E^T$, both iterates can be obtained by solving

one single linear system with an augmented right hand side:

$$[V_{P,j},\ V_{Q,j}] = (A + \alpha_j E)^{-1}[W_{P,j-1},\ W_{Q,j-1}].$$

This is the same as the situation in the fADI iteration (3.50) for GCALEs with unsymmetric inhomogeneities. As reported in Section 3.3.4, this will indeed be faster than solving for $V_{P,j},\ V_{Q,j}$ separately. Obviously, the associated Algorithm 4.6 can be readily applied here with $F = B$, $G = -C$. The low-rank solution factors for P and Q are given by the computed factors Z and Y. Hence, after the same number of steps, Algorithm 4.6 and Algorithm 7.2 construct identical results in this case.

Remark 7.1:

For the case $E = I_n$, a conceptually very different approach that tries to deal with both CALEs in (7.7) at the same time is proposed in [243]. The approximate implicit subspace iteration with alternating directions (AISIAD) successively computes dominant, invariant subspaces of P and Q and uses the subspace for P to reduce the amount of computations in the next step for Q and vice versa. In each iteration step of AISIAD, the key ingredient is to transform the two CALEs in (7.7) into specially structured Sylvester equations. These can then solved efficiently by Arnoldi type methods or specialized algorithms for CASEs [215, 30]. For (7.7a), the CASE to be solved at step $k \geq 1$ is of the form

$$A\hat{P}_k + \hat{P}_k\hat{A}_k + \hat{M}_k = 0, \quad \hat{P}_k, \hat{M}_k \in \mathbb{R}^{n\times k},\ \hat{A}_k \in \mathbb{R}^{k\times k}.$$

Since the dimension of $\hat{A}_k \in \mathbb{R}^{k\times k}$ increases with k in the course of AISIAD, this approach is likely to be numerically more expensive compared to our LR-SRBT approach employing the dual G-LR-ADI iteration. ◊

7.2.2. Stopping the Dual Iteration

As we discussed in the previous chapters, the G-LR-ADI iteration can be terminated once the norm of the scaled Lyapunov residual is smaller than some tolerance. For the dual G-LR-ADI iteration, where we have to monitor two residuals $\mathcal{L}_{P,j},\ \mathcal{L}_{Q,j}$ for (7.7a),(7.7b), this gives a termination criterion

$$\|\mathcal{L}_{P,j}\| = \|W_{P,j}^T W_{P,j}\| < \tau_{\mathcal{L}}\|B^T B\| \quad \wedge \quad \|\mathcal{L}_{Q,j}\| = \|W_{Q,j}^T W_{Q,j}\| < \tau_{\mathcal{L}}\|CC^T\| \tag{7.18}$$

which is already included in Algorithm 7.2.

In [33, 37], some arguments against this residual based stopping in the context of BT are given. The residual of the GCALEs is often not directly related to the accuracy of the reduced order model that we are actually interested in the model order reduction. On the one hand, one frequently observes very good reduced order model accuracies even when the G-LR-ADI iteration does not converge at all. On the other hand, there are examples where one of the ADI iterations converges much faster than the other one. Since the dimension of the reduced order model is at most as large as the smaller of the

two column dimensions of Z_P and Z_Q, this can limit the accuracy of the reduced order model and prevent us from fulfilling prescribed error bounds. Therefore, we discuss alternative stopping criteria ideas next that try to overcome these issues in balanced truncation model order reduction when the dual G-LR-ADI iteration, but also any other dual Lyapunov solver, is employed to solve (7.7).

Goal-Oriented Stopping via the Approximate Hankel Singular Values

The actual properties of interest in the (LR)-SRBT method are the Hankel singular values of the original system. To be precise, we want to capture the leading Hankel singular values as accurate as possible since these describe the dominant dynamics of the system and thus need to be reflected in the reduced order model. The two solution factors Z_P, Z_Q can be employed to compute (7.10) during the dual iteration for (7.7). In the following, superscripts $^{(j)}$ denote quantities of the SVD (7.10) at iteration step j of Algorithm 7.2. Hence, at step j we have $j \cdot \min(m,\ p)$ nonzero approximate Hankel singular values. We will consider two stopping criteria based on these values.

At first, let us assume we are searching for a reduced order model of a fixed dimension $r \in \mathbb{N}$. Following [33, 37], we monitor the change of the leading r singular values in $\Sigma_1^{(j)} \in \mathbb{R}^{r \times r}$. Once these singular values stagnate we have matched the corresponding subsystem and, thus, get a good reduced order model evaluating (7.12). This way we find a reliable criterion when to stop the iteration in contrast to the Lyapunov residuals that may not tell us anything about the approximation. In finite arithmetic it is sufficient to drive the relative change of the leading Hankel singular values below a certain tolerance. Here, relative is to be understood as relative to the largest singular value. This suggests a termination criterion of the form

$$\Delta_\infty \boldsymbol{\sigma} := \|\boldsymbol{\sigma}_r^{(j)} - \boldsymbol{\sigma}_r^{(j-1)}\|_\infty / \sigma_1^{(j)} \quad < \quad \tau_\sigma, \quad 0 < \tau_\sigma \ll 1, \tag{7.19}$$

where $\boldsymbol{\sigma}_r^{(j)} := (\sigma_1^{(j)}, \ldots, \sigma_r^{(j)})^T$. We can use a start-up phase in which we do not evaluate the stopping criterion at all. This phase is pre-determinated by the size of the matrices B and C. The dual G-LR-ADI iteration should at least be running as long as Z_P and Z_Q do not have a minimum of r linearly independent columns each. Moreover, after the start-up phase one can also evaluate this measure only every couple of steps, e.g., every 5th-step, of the dual G-LR-ADI iteration.

An alternative but similar approach which does not depend on a pre-specified reduced order r is motivated by the theoretical error bound (7.8). We can also monitor the relative change in the sum of all $j \cdot \min(m,\ p)$ computed nonzero singular values. Let $\boldsymbol{\sigma}^{(j)} := (\sigma_1^{(j)}, \ldots, \sigma_{j \cdot \min(m,\ p)}^{(j)})^T$ such that the stopping criterion becomes

$$\Delta_1 \boldsymbol{\sigma} := \left| \|\boldsymbol{\sigma}^{(j)}\|_1 - \|\boldsymbol{\sigma}^{(j-1)}\|_1 \right| / \|\boldsymbol{\sigma}^{(j)}\|_1 < \tau_\sigma, \quad \boldsymbol{\sigma}^{(j)} := (\sigma_1^{(j)}, \ldots, \sigma_{j \cdot \min(m,\ p)}^{(j)})^T. \tag{7.20}$$

If the change in $\|\boldsymbol{\sigma}\|_1$ is only marginal, we may stop the iteration since the largest HSVs are found and the smallest ones will be neglected in the truncation step anyway. Notice that $\|\boldsymbol{\sigma}^{(j)}\|_1$ is also the Ky-Fan and Schatten-1 norm [131] of the matrix $G_j = Z_{Q,j}^T E Z_{P,j}$.

The quantitative analysis of the convergence of the Hankel singular values is still under investigation [118]. In [33], some basic qualitative ideas to motivate the stopping criteria (7.19),(7.20) above can be found. From a practical point of view it is obvious that the criteria are reasonable quantities to look at if the Hankel singular values converge. Unfortunately, as we will see later, both criteria (7.19) and (7.20) often show an irregular behavior in practice as the dual G-LR-ADI iteration proceeds. It may happen that these criteria might terminate the dual G-LR-ADI iteration too early which can destroy the stability preservation of BT [3] due to utilizing crude low-rank approximations of P, Q. Some of the upcoming numerical experiments indeed show such effects.

The tolerance τ_σ can in both cases be related to the desired accuracy τ_{BT}. Since only approximate singular values are used in (7.19), (7.20) one should use $\tau_\sigma < \tau_{\mathrm{BT}}$.

Implementational Aspects

The stopping criteria above require in each step of Algorithm 7.2 the computation of a thin SVD of the matrix $G_j = Z_{Q,j}^T E Z_{P,j}$ which can become more expensive than computing the norms of the GCALE residuals. Here, we mention some basic strategies to make this step less costly. The matrix G_j does not need to be computed completely from scratch since it can be accumulated efficiently, assuming that the shift α_j is real, via

$$G_j = Z_{Q,j}^T E Z_{P,j} = \begin{bmatrix} G_{j-1} & \gamma Z_{Q,j-1}^T E V_{P,j} \\ \gamma (V_{Q,j})^T E Z_{P,j-1} & \gamma^2 (V_{Q,j})^T E V_{P,j} \end{bmatrix},$$

where $\gamma := \sqrt{-2\alpha_j}$. In the complex case this augmentation can be carried out similarly by adding $2p$ rows and $2m$ columns w.r.t. the the double step. Since G_j is obtained by adding new columns $\gamma (Z_{Q,j-1})^T E V_{P,j}$ and new rows $[\gamma (V_{Q,j})^T E Z_{P,j-1},\ \gamma^2 (V_{Q,j})^T E V_{P,j}]$ to G_{j-1}, updating strategies for the SVD [115, 61] can be employed to reduce the cost for computing the SVD of G_j. There, however, either new rows or columns are added to a matrix with an already known SVD but not both as it is the situation here, such that the proposed updating strategies in [115, 61] have to be applied consecutively, e.g, once for the new rows and afterwards for the new columns. For most of our examples, the encountered SVDs are comparably small and only minor savings are expected from these ideas such that we do not pursue them further. Once the dual G-LR-ADI terminates using the stopping criterion (7.19) or (7.20), the SVD of G_j can be reused for the construction of T_L, T_R in (7.11). Further savings can be made if $Z_{P,j} = Z_{Q,j}$ which is the case when A and E are symmetric and $B = \pm C^T$.

7.2.3. Shift Parameters for the Dual Iteration

We briefly discuss the adaptation of the shift parameters generation strategies from Chapter 5 to Algorithm 7.2. The a-priori computed Wachspress or heuristic shift parameters from Section 5.2 can be used right away since these are typically generated from Ritz values w.r.t. $(A,\ E)$ resulting in a single set of shift parameters. Regarding

the self-generating $V(u)$-shifts from Section 5.3.1, we propose to merge the Ritz values obtained from applying this approach to both the P- and Q-iteration. For the latter one it holds, similarly to (6.12a), by Corollary 4.6 that $A^T Z_{Q,j} = W_{Q,j} G_{\text{ADI-r}}^T + E^T Z_{Q,j} \hat{B}_{\text{ADI-r}}$ with $\hat{B}_{\text{ADI-r}}$ constructed as $B_{\text{ADI-r}}$ w.r.t. the conjugated set of shift parameters. Hence, (5.3) can be used twice to obtain the restrictions of A and A^T without additional matrix vector products. This will yield a set of cardinality up to $(u+1)(m+p)$ from which a smaller number, e.g., $\max(p, m)$, is chosen using the heuristic shift approach.

For the residual minimizing extension of the $V(u)$-shifts, we might look for the single shift α_{j+1} that simultaneously minimizes both GCALE residual norms. This can be easily achieved by considering, e.g., the joint objective function

$$f_j^{\text{dual}}(\nu,\ \xi) := \frac{1}{2}\left(\|\mathcal{C}(A, E, \alpha) W_{P,j}\|^2 + \|\mathcal{C}(A, E, \alpha)^H W_{Q,j}\|^2\right). \tag{7.21}$$

The gradient and Hessian of (7.21) are basically obtained by adding gradient and Hessian of the two objective functions (5.4) w.r.t. $W_{P,j}$ and $W_{Q,j}$. The other techniques discussed in Section 5.3.2 are applied here as well in a straightforward manner.

It should be mentioned that it is indeed possible that the so obtained residual norm-minimizing shifts lead to a slower convergence compared to using (5.4) separately for the P- and Q-iteration, i.e., working with different shifts for both GCALEs. However, in our numerical tests only minor differences were observed and using the single shift approach with (7.21) paid off due to a cheaper generation effort.

7.2.4. Balanced Truncation for Second Order Systems

By rewriting a second order system (2.32) into an equivalent generalized first order system (2.5) defined by, e.g., (2.33a) or (2.33a), LR-SRBT can be readily applied. The low-rank factors Z_P, Z_Q of the Gramians can be computed by the SO-LR-ADI iteration (Algorithm 3.3) which is especially tailored for this purpose. The reduced system (7.2) will, however, no longer have the second order structure of the original system (2.32). Hence, the reduced states no longer have physical interpretations which is, e.g., especially important in elastic multibody systems [157, 92, 91, 178] or certain electrical systems [239]. These applications demand the preservation of the second order form. Several structure preserving variants of BT have been developed for this purpose [65, 239, 189, 51, 53, 178, 31, 37] and are often referred to under the name *second order BT*.

The main idea is to partition the Gramians P and Q accordingly to the structure present in the equivalent generalized first order system:

$$P = \begin{bmatrix} P_{\text{p}} & P_{1,2} \\ P_{1,2}^T & P_{\text{v}} \end{bmatrix},\ Q = \begin{bmatrix} Q_{\text{p}} & Q_{1,2} \\ Q_{1,2}^T & Q_{\text{v}} \end{bmatrix},$$

where $P_{\text{p}},\ Q_{\text{p}} \in \mathbb{R}^{n \times n}$ and $P_{\text{v}},\ Q_{\text{v}} \in \mathbb{R}^{n \times n}$ are called *position*, *velocity* reachability and observability Gramians, respectively.

We will restrict in the following to symmetric second order systems, i.e., $M, D, K \succ 0$ symmetric and $B_1 = \pm C_{\text{p}}^T,\ C_{\text{v}} = 0$ or $B_1 = \pm C_{\text{v}}^T,\ C_{\text{p}} = 0$. This class is especially

present in the applications [157, 92, 91, 178, 239] mentioned above. Symmetric second order systems can be transformed into an equivalent symmetric first order system with $E = E^T$, $A = A^T$ and $B = \pm C^T$ such that $P \equiv Q$, see, e.g., [37]. Consequently, it is enough to consider only P_{p} and P_{v} [178, 37], or respectively low-rank factorizations $P_{\mathrm{p}} \approx Z_{\mathrm{p}} Z_{\mathrm{p}}^T$ and $P_{\mathrm{v}} \approx Z_{\mathrm{v}} Z_{\mathrm{v}}^T$. Following the approach in [189], one can choose between four possible SVDs

$$\begin{aligned} Z_{\chi_1}^T M Z_{\chi_2} &= X_{\chi_1\chi_2} \Sigma_{\chi_1\chi_2} Y_{\chi_1\chi_2}^T \\ &= [X_{\chi_1\chi_2,1}, X_{\chi_1\chi_2,2}] \begin{bmatrix} \Sigma_{\chi_1\chi_2,1} & 0 \\ 0 & \Sigma_{\chi_1\chi_2,2} \end{bmatrix} [Y_{\chi_1\chi_2,1}, Y_{\chi_1\chi_2,2}]^T, \end{aligned} \tag{7.22}$$

where the subscripts $\chi_1,\ \chi_2 \in \{\mathrm{p},\ \mathrm{v}\}$ denote whether the position or velocity blocks of P are used. Similar to the first order case, the $\Sigma_{\chi_1\chi_2,1} \in \mathbb{R}^{r\times r}$ block contains the largest singular values. Depending on the choice of χ_1, χ_2, they are referred to as position-position (PP) if $\chi_1 = \chi_2 = \mathrm{p}$, velocity-velocity (VV) if $\chi_1 = \chi_2 = \mathrm{v}$, velocity-position (VP) if $\chi_1 = \mathrm{v}, \chi_2 = \mathrm{p}$, and position-velocity (PV) singular-values if $\chi_1 = \mathrm{p},\ \chi_2 = \mathrm{v}$. This yields four different pairs of matrices which perform the reduction:

$$T_{R,\chi_1\chi_2} := Z_{\chi_1} Y_{\chi_1\chi_2,1} \Sigma_{\chi_1\chi_2,1}^{-\frac{1}{2}}, \quad T_{L,\chi_1\chi_2} := Z_{\chi_2} X_{\chi_1\chi_2,1} \Sigma_{\chi_1\chi_2,1}^{-\frac{1}{2}}. \tag{7.23}$$

Hence, second order BT can be carried out in four variants which yields four different possible reduced order models in second order form:

$$\begin{aligned} \tilde{M}_{\chi_1\chi_2} \ddot{\tilde{x}}(t) + \tilde{D}_{\chi_1\chi_2} \dot{\tilde{x}}(t) + \tilde{K}_{\chi_1\chi_2} \tilde{x}(t) &= \tilde{B}_{\chi_1\chi_2} u(t), \\ \tilde{y}(t) &= \tilde{C}^{\mathrm{p}}_{\chi_1\chi_2} \tilde{x}(t) + \tilde{C}^{\mathrm{v}}_{\chi_1\chi_2} \dot{\tilde{x}}(t), \end{aligned}$$

with

$$\begin{aligned} &\tilde{M}_{\chi_1\chi_2} := T_{L,\chi_1\chi_2}^T M T_{R,\chi_1\chi_2}, \quad \tilde{D}_{\chi_1\chi_2} := T_{L,\chi_1\chi_2}^T D T_{R,\chi_1\chi_2}, \\ &\tilde{K}_{\chi_1\chi_2} := T_{L,\chi_1\chi_2}^T K T_{R,\chi_1\chi_2} \in \mathbb{R}^{r\times r}, \quad \tilde{B}_{\chi_1\chi_2} := T_{L,\chi_1\chi_2}^T B \in \mathbb{R}^{r\times m}, \\ &\tilde{C}^{\mathrm{p}}_{\chi_1\chi_2} := B^T T_{R,\chi_1\chi_2}, \quad \tilde{C}^{\mathrm{v}}_{\chi_1\chi_2} := B^T T_{R,\chi_1\chi_2} \in \mathbb{R}^{m\times r}. \end{aligned} \tag{7.24}$$

It can easily be shown that the reduced mass matrices $\tilde{M}_{\chi_1\chi_2}$ are always equal to the identity. The reduced position and velocity output matrices $\tilde{C}^{\mathrm{p}}_{\chi_1\chi_2}$, $\tilde{C}^{\mathrm{v}}_{\chi_1\chi_2}$ are only built if they exist in the original model. The preservation of the second order structure comes at the price of the absence of the guaranteed stability preservation of BT (if exact Gramians are used). Hence, second order BT does not provide an error bound like (7.8). As an alternative one can monitor the ratio of the entries in $\Sigma_{\chi_1\chi_2}$ and determine the reduced dimension once

$$\frac{\sigma_{j,\chi_1\chi_2}}{\sigma_{1,\chi_1\chi_2}} \leq \tau_{\mathrm{BT}} \tag{7.25}$$

is fulfilled [201]. There is no theoretical result describing which variant of second order balanced truncation will lead to the most accurate reduced order models. One advantage

of the position-position and velocity-velocity approaches is that they preserve stability [178, 37] and also the symmetry of the system, i.e., all reduced coefficient matrices remain symmetric, positive definite, and the reduced input matrix is still the transpose of the reduced (position or velocity) output matrix. The velocity-position and position-velocity reduced order models are adjoint to each other. Hence, they show the same frequency response plot in the spectral or Frobenius norm [178].

The goal-oriented stopping criteria mentioned before can by directly carried over to second order BT and the SO-LR-ADI iteration by simply using the relevant singular values in $\Sigma_{\chi_1\chi_2}$, see [37].

7.2.5. Numerical Examples

Here, we evaluate the execution of LR-SRBT (Algorithm 7.1) by means of low-rank solution factors computed by the (dual) G-LR-ADI iteration (Algorithms 4.1, 7.2) using a few numerical studies. We consider the test examples *ifiss66k*, *bips* and *chain*. The matrices B, C for the *ifiss66k* example consist of $m, p = 5$ columns and rows with random entries. For *bips* the SLRCF-ADI iteration [98] (cf. Section 3.2.5) and the corresponding adapted version of BT is used. The *chain* example is a symmetric second order system such that the SO-LR-ADI iteration (Algorithm 3.3) is used and the reduction is carried out to first as well as second order form.

At first, we briefly compare to solution of the GCALEs for both Gramians (7.7) by two separate runs of the G-LR-ADI iteration and by the dual G-LR-ADI iteration (Algorithm 7.2) for the examples *ifiss66k* and *bips*. The residual norm-minimizing shift approach from Section 5.3.2 is employed as well as the modification using (7.21) for the dual iteration. We do not reuse the LU factorizations because this did not led any performance gains. The history of the normalized residual norm of the P- and Q-iteration by both approaches is plotted in Figure 7.1. For the *ifiss66k* example the dual iteration requires slightly fewer iteration steps and a smaller computation time of 164.1 seconds compared to 170 seconds for the separate iterations. The result for the *bips* example are similar: the dual G-LR-ADI iteration required with 21.8 seconds less time than the separate approach which took 27.4 seconds. Using the joint objective function (7.21) to obtain shift parameters appears to work satisfactorily for the dual iteration. Figure 7.1 also shows the progress of the goal oriented stopping quantity $\Delta_1\boldsymbol{\sigma}$ from (7.20) during the dual G-LR-ADI iteration. Apparently, $\Delta_1\boldsymbol{\sigma}$ leads to an earlier termination of the iteration but its highly oscillatory behavior makes it somewhat less reliable as anticipated. The criterion (7.19) showed similar oscillations which is in line with the observations reported in [33, 37].

Now, we carry out LR-SRBT (Algorithm 7.1) on the basis of the low-rank factors obtained after the residual norm related termination using (7.18) and via the goal-oriented stopping with (7.20). The dimension of the reduced order model was determined adaptively using (7.9) with $\tau_{\mathrm{BT}} = 10^{-5}$. As error measurement we compute the transfer function matrices $H(s)$ and $\tilde{H}(s)$ of original and reduced systems and consider the

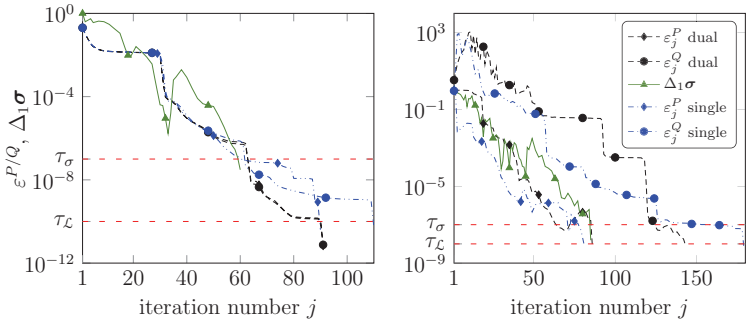


Figure 7.1.: History of the residual norms within single and dual G-LR-ADI iteration as well as $\Delta_1\boldsymbol{\sigma}$ for the *ifiss66k* (left) and *bips* (right) test systems.

absolute and relative errors

$$\mathcal{E}^{\text{abs}}(\omega) := \|H(\jmath\omega) - \tilde{H}(\jmath\omega)\|_2, \quad \mathcal{E}^{\text{rel}}(\omega) := \mathcal{E}^{\text{abs}}(\omega)/\|H(\jmath\omega)\|_2 \tag{7.26}$$

for values $\omega \in \Omega = [\omega_1,\ \omega_2] \subset \mathbb{R}_+$ from a prescribed interval. Table 7.1 gives the used interval Ω and stopping tolerances $\tau_\mathcal{L}, \tau_\sigma$, the number of dual G-LR-ADI iteration steps j_{it}^P, j_{it}^Q, the final obtained residual norms $\varepsilon_{j_{\text{it}}}^P$, $\varepsilon_{j_{\text{it}}}^Q$ w.r.t. both Gramians, the total consumed execution time t_{ADI} of the dual G-LR-ADI iteration, the obtained reduced order r, the largest values of the relative errors (7.26) in Ω, as well as the value of theoretical error bound (7.8). In the last column we also indicate if the generated reduced order model is stable. The error bound, absolute and relative errors $\mathcal{E}^{\text{abs}}$ and $\mathcal{E}^{\text{rel}}$ of the *ifiss66k* and *bips* examples are illustrated in Figure 7.2.

Evidently, using the goal-oriented stopping approach (7.20) leads to fewer total iteration steps and smaller G-LR-ADI execution times t_{ADI}, but also to larger residual norms $\varepsilon_{j_{\text{it}}}^P$, $\varepsilon_{j_{\text{it}}}^Q$ and the generated reduced order models are also less accurate judging by the values of $\mathcal{E}_{\max}^{\text{rel}}$. For the examples *ifiss66k* and *chain*, this earlier termination of the dual G-LR-ADI iteration has drastic consequences because of the generation of unstable reduced order models. This could also be observed for the residual based stopping with larger values of $\tau_\mathcal{L}$ which suggests that a minimum accuracy of the Gramian approximations might be required in order to preserve stability. Recall that in case of an unstable reduced order model, the error bound (7.8) looses its meaning. Hence, the criteria (7.19),(7.20) based on the approximate Hankel singular values should be employed with care.

It is also important to note that for the examples *ifiss*, *chain* a violation of the theoretical error bound (7.8) can be observed in some case. As remarked earlier, we suspect that this is caused by the usage of approximate GCALE solutions since (7.8) only holds when exact solutions are used.

For the symmetric second order system *chain*, we additionally apply the second order

Table 7.1.: Results for the examples regarding Gramian approximation and model reduction.

Example	stop	$j_{\mathrm{it}}^P,\ j_{\mathrm{it}}^Q$	$\varepsilon_{j_{\mathrm{it}}}^P,\quad \varepsilon_{j_{\mathrm{it}}}^Q$	t_{ADI}	r	$\mathcal{E}_{\max}^{\mathrm{rel}}$	bound	$\mathbb{C}_-$?
ifiss66k	$\tau_{\mathcal{L}} = 10^{-10}$	91, 91	$6.8{\cdot}10^{-12},\ 7.6{\cdot}10^{-12}$	164.1	339	$1.1{\cdot}10^{-10}$	$7.5{\cdot}10^{-6}$	1
$\Omega{=}[10^{-1}, 10^4]$	$\tau_{\boldsymbol{\sigma}} = 10^{-7}$	60, 60	$6.3{\cdot}10^{-7},\ 5.1{\cdot}10^{-7}$	110.7	282	$1.4{\cdot}10^{-6}$	$3.3{\cdot}10^{-6}$	0
bips	$\tau_{\mathcal{L}} = 10^{-8}$	86, 143	$7.6{\cdot}10^{-9},\ 9.7{\cdot}10^{-9}$	21.9	159	$2.7{\cdot}10^{-5}$	$8.4{\cdot}10^{-6}$	1
$\Omega{=}[10^{-1}, 10^3]$	$\tau_{\boldsymbol{\sigma}} = 10^{-7}$	85, 85	$1.6{\cdot}10^{-7},\ 3.5{\cdot}10^{-2}$	16.7	157	$2.6{\cdot}10^{-5}$	$8.2{\cdot}10^{-6}$	1
chain	$\tau_{\mathcal{L}} = 10^{-10}$	192	$6.7{\cdot}10^{-11}$	22.7	644	$2.1{\cdot}10^{-7}$	$4.6{\cdot}10^{-6}$	1
$\Omega{=}[10^{-3}, 10^3]$	$\tau_{\boldsymbol{\sigma}} = 10^{-8}$	105	$8.6{\cdot}10^{-6}$	16.4	433	$4.8{\cdot}10^{-5}$	$3.0{\cdot}10^{-6}$	0

chain $\Omega{=}[10^{-3}, 10^3]$ reduction to second order form with $r = 300$	stop	type	j_{it}^P	$\varepsilon_{j_{\mathrm{it}}}^P$	t_{ADI}	$\mathcal{E}_{\max}^{\mathrm{rel}}$	$\mathbb{C}_-$?
	$\tau_{\mathcal{L}}{=}10^{-10}$	pp	192	$9.6{\cdot}10^{-10}$	22.9	$1.3{\cdot}10^{-1}$	1
		vv				$9.5{\cdot}10^{-4}$	1
		vp				$3.2{\cdot}10^{-3}$	1
	$\tau_{\boldsymbol{\sigma}}{=}10^{-8}$	pp	87	$3.2{\cdot}10^{-4}$	12.5	$2.0{\cdot}10^{-2}$	1
		vv	123	$2.9{\cdot}10^{-7}$	21.6	$9.5{\cdot}10^{-4}$	1
		vp	123	$2.9{\cdot}10^{-7}$	21.7	$2.0{\cdot}10^{-2}$	1

BT variants but fix the reduced dimension to $r = 300$ which roughly corresponds to the half of the dimension of the reduced system obtained by BT to first order form. The obtained reduced order models in second order form are, however, less accurate than the reduced systems in first order form. The results from using the goal-oriented stopping criteria lead to reduced systems of comparable accuracy at an reduced amount of G-LR-ADI iteration steps. Because of the required SVD computations, no significant time savings are gained despite the smaller numbers of iteration steps compared to the residual based stopping. Other goal-oriented stopping ideas, e.g., based on (7.25), led to similar reduction results.

To conclude this section, the G-LR-ADI iteration can be used right away to construct low-rank solution factors of the Gramians (7.6). This can be done simultaneously by the dual G-LR-ADI iteration. The proposed goal-oriented stopping criteria lead to an earlier termination of the iteration at the price of less accurate reduced order models which are occasionally also unstable. Hence, further research effort should be put to deal with these disadvantages. For instance, taking also the singular vectors of the matrix G_j into account might give more reliable criteria.

7.3. Balanced Truncation in Limited Frequency Intervals

By looking at the error bound (7.8), the above balanced truncation framework aims at generating reduced order models that are accurate for all values $\omega \in \mathbb{R}$, which, from an application oriented view, are typically considered as frequencies. In several applications, however, the underlying physical or technical system operates only in a small frequency interval $[\omega_1,\ \omega_2]$ of interest. Restricting the BT procedure to this frequency interval

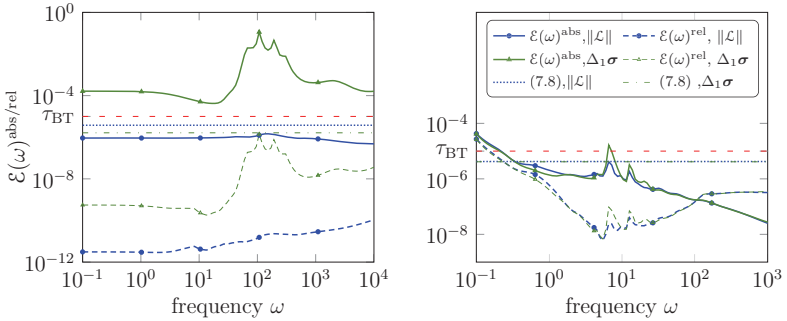


Figure 7.2.: Absolute and relative errors w.r.t. the transfer function matrix for the *ifiss66k* (left) and *bips* (right) test systems.

establishes frequency-limited balanced truncation (FLBT) which was proposed in [107]. One motivation for FLBT is that, compared to ordinary, unrestricted BT, by restricting to a small interval $[\omega_1,\ \omega_2]$, we hope to obtain higher accuracies with reduced order models of the same dimension, or to achieve a comparable accuracy with smaller reduced order models inside the interval, while allowing for larger errors outside.

The main purpose of this section is to provide a numerical efficient framework for carrying out FLBT for high-dimensional systems. For the ease of representation we restrict to standard state-space systems $(E = I_n, A, B, C) = (A, B, C)$ in the main part of this section. The general situation $E \neq I_n$ is elaborated on later. We start by reviewing the concept of frequency-limited Gramians and show how to formulate a procedure similar to the square root approach in Algorithm 7.1 for balanced truncation without frequency limitations. In Section 7.3.2, we investigate the eigenvalue decay of the frequency-limited Gramians. For the occurring CALEs, we will, as in the unlimited case, employ low-rank approximations of the solutions. It turns out that frequency-limited balanced truncation involves, in addition to solving these CALEs, evaluating a nonlinear matrix valued function whose efficient numerical treatment is topic of Section 7.3.3. We will show that it is possible to deal efficiently with both the matrix valued function and the CALEs in a single algorithm using extended or rational Krylov subspaces with appropriate shifts. Generalized LTI systems (2.5) and some comments on further modifications and variations of balanced truncation with restrictions are given in Section 7.3.4. Numerical experiments in Section 7.3.5 illustrate the performance of our approaches with respect to the accuracy of the constructed reduced systems and the computational efficiency regarding the GCALE solution.

Remark 7.2:

The techniques discussed prior in Section 7.2, i.e., the simultaneous solution of both occurring matrix equations, the goal-oriented stopping criteria, and structure exploiting variants for second order systems, can be modified easily for the use in FLBT and

are, hence, omitted for the sake of brevity. ◊

7.3.1. Frequency-Limited Gramians

By employing the Fourier transformation, the Gramians (7.6) of (A, B, C) can be represented in the frequency domain as

$$P = \frac{1}{2\pi} \int_{-\infty}^{\infty} \Psi(\jmath\nu) B B^T \Psi(\jmath\nu)^H \mathrm{d}\nu, \quad Q = \frac{1}{2\pi} \int_{-\infty}^{\infty} \Psi(\jmath\nu)^H C^T C \Psi(\jmath\nu) \mathrm{d}\nu \tag{7.27}$$

with the resolvent $\Psi(\jmath\nu) := (\jmath\nu I - A)^{-1}$. Restricting the integration limits in the integrals (7.27) to certain (unions of) intervals $\Omega \subseteq \mathbb{R}$, gives the *frequency-limited Gramians* P_Ω, Q_Ω.

Definition 7.3 (Frequency-limited Gramians [107]):

For the system (A, B, C), the frequency-limited reachability and observability Gramians w.r.t. $\Omega \subset \mathbb{R}$ are defined by

$$P_\Omega = \frac{1}{2\pi} \int_{\Omega} \Psi(\jmath\nu) B B^T \Psi(\jmath\nu)^H \mathrm{d}\nu, \quad Q_\Omega = \frac{1}{2\pi} \int_{\Omega} \Psi(\jmath\nu)^H C^T C \Psi(\jmath\nu) \mathrm{d}\nu. \tag{7.28}$$

Since the system (A, B, C) is defined by real matrices, the considered frequency region should be symmetric w.r.t. zero: $\Omega = -\Omega$, for instance, in the form

$$\Omega := [-\omega_2, -\omega_1] \cup [\omega_1, \omega_2], \quad 0 \leq \omega_1 < \omega_2 < \infty. \tag{7.29}$$

The following Theorems 7.4–7.5 give important representations of P_Ω, Q_Ω and their proofs can be found in, e.g., [107, Section 4] and [186, Section 3.1].

Theorem 7.4 (CALEs for the frequency-limited Gramians [107, 186]):

Consider the system (A, B, C) and a frequency region $\Omega \subset \mathbb{R}$, $\Omega = -\Omega$. Then the frequency limited Gramians P_Ω and Q_Ω are given in the following equivalent ways:

1. Using the ordinary reachability and observability Gramians P and Q from (7.6), it holds that

$$P_\Omega = F_\Omega P + P F_\Omega^T, \quad Q_\Omega = F_\Omega^T Q + Q F_\Omega \tag{7.30}$$

with

$$F_\Omega := \frac{1}{2\pi} \int_{\Omega} \Psi(\jmath\nu) \mathrm{d}\nu. \tag{7.31}$$

2. One can express P_Ω, Q_Ω as the solutions of the *frequency-limited reachability and observability CALEs*

$$AP_\Omega + P_\Omega A^T + F_\Omega BB^T + BB^T F_\Omega^T = 0, \tag{7.32a}$$
$$A^T Q_\Omega + Q_\Omega A + F_\Omega^T C^T C + C^T C F_\Omega = 0. \tag{7.32b}$$

The eigenvalues of the product $P_\Omega Q_\Omega$ are, similar to the case $\Omega = \mathbb{R}$, called *frequency-limited Hankel singular values*. The next theorem establishes that the matrix F_Ω is real and can also be represented via the matrix-valued natural logarithm.

Theorem 7.5 (Expression of F_Ω [107, 186]):

The matrix-valued integral (7.31) can, for Ω as in (7.29), be expressed via

$$F_\Omega = \frac{1}{\pi} \operatorname{Re} \left(\int_{\omega_1}^{\omega_2} \Psi(\jmath\nu) \mathrm{d}\nu \right) \tag{7.33a}$$
$$= \operatorname{Re} \left(\frac{\jmath}{\pi} \ln \left((A + \jmath\omega_1 I_n)^{-1} (A + \jmath\omega_2 I_n) \right) \right) = \operatorname{Re} \left(\frac{\jmath}{\pi} \ln \left(\mathcal{C}(A, \jmath\omega_1, \jmath\omega_2) \right) \right), \tag{7.33b}$$

where $\mathcal{C}(A,\ \mu,\ \nu) = (A + \mu I_n)^{-1}(A + \nu I_n)$ denotes a Cayley transformation of A (cf. Definition 2.15) and $\ln(M)$ is the principal branch of the matrix valued natural logarithm of M with $\Lambda(M) \cap \mathbb{R}_- = \emptyset$. ◊

For frequency regions of the form $\Omega = [-\omega,\ \omega]$, it can be shown that (7.33b) simplifies to

$$F_\Omega = \operatorname{Re} \left(\frac{\jmath}{\pi} \ln \left(-A - \jmath\omega I_n \right) \right).$$

Moreover, the above Theorems 7.4 and 7.5 can be generalized to multiple, concatenated segments in the frequency domain

$$\Omega = \bigcup_{i=1}^{k} [-\omega_{2i},\ -\omega_{2i-1}] \cup [\omega_{2i-1},\ \omega_{2i}] \quad \text{with} \quad 0 \leq \omega_1 < \omega_2 < \ldots < \omega_{2k} < \infty,$$

where the matrix-valued logarithm of a product of k Cayley transformations occurs, see [186, Corollary 3.1]. For simplification and brevity, we mainly focus on frequency restrictions of the form (7.29) in the remainder.

Computing F_Ω involves the evaluation of a function $f(.)$ in A with $f(z) = \ln \frac{z+\jmath\omega_2}{z+\jmath\omega_1}$, i.e., the logarithm of a Cayley transformation. For matrices of large dimensions, this appears to be a very formidable and expensive task. Some strategies that make an efficient treatment of F_Ω possible are proposed in Section 7.3.3. There, the numerical approximation of the frequency-limited Gramians by means of low-rank solutions $P_\Omega \approx Z_{P_\Omega} Z_{P_\Omega}^T$, $Q_\Omega \approx Z_{Q_\Omega} Z_{Q_\Omega}^T$ is also investigated. The eigenvalue decay of P_Ω, Q_Ω and, consequently,

how well they can be approximated by such low-rank solutions is considered in the next section. With the low-rank solution factors Z_{P_Ω}, Z_{Q_Ω}, FLBT can be carried out in a similar way as Algorithm 7.1 by substituting the CALEs (7.6) for the infinite Gramians by the frequency-limited CALEs (7.32) in Line 1 and using Z_{P_Ω}, Z_{Q_Ω} in the remaining steps. In contrast to BT without frequency restrictions, FLBT is not guaranteed to preserve the stability of the original system and also no error bound like (7.8) can be given. We briefly come back to this issue in Section 7.3.4.

7.3.2. On the Eigenvalue Decay of the Frequency-Limited Gramians

We expect that the frequency-limited Gramians P_Ω, Q_Ω in (7.32) can be well approximated by low-rank solutions because their inhomogeneities are of low rank $2m$, $2p \ll n$. Comparing the infinite CALEs (7.6) with the frequency-limited ones (7.32), these inhomogeneities are the only differences in (7.46). Recalling the theory on the existence of low-rank solutions of matrix equations [184, 4, 113, 222], the rank of the inhomogeneity of a matrix equation is an influential factor on the numerical rank of the solution. The rank of the inhomogeneities in (7.32) is twice as large as the rank of the inhomogeneities BB^T and C^TC of the CALEs (7.6) for the infinite Gramians. Thus, one is, e.g., by reviewing Corollary 2.41, tempted to expect that the numerical ranks of P_Ω, Q_Ω are larger than the numerical ranks of P, Q. Observations in practice, however, often show the exact opposite phenomenon, i.e., P_Ω, Q_Ω have smaller numerical ranks than P, Q.

On the one hand, this seems to be counterintuitive as the coefficient matrices in both (7.6) and (7.32) are identical. On the other hand, comparing (7.27) and (7.28), it appears intuitively clear that P_Ω, Q_Ω have smaller numerical ranks since the integration range is smaller such that less information enters the integrals. A general approach for investigating the numerical rank of solutions of matrix equations is to look at the eigenvalue decay of the solutions. As for the unlimited Gramians, however, obtaining a general analytic prediction on the exact eigenvalue decay is difficult. Motivated by the approaches used in [4, 222] (cf. Section 2.3.3), we try to bound the eigenvalues of P and P_Ω in the following. To this end, we restrict to the infinite and frequency-limited reachability Gramians P, P_Ω in the SISO case, i.e., $B = b \in \mathbb{R}^n$. The observability Gramians can be dealt with similarly and generalizations to the MIMO case can be drawn from, e.g, [4]. We introduce the notation $\mu_j^{\downarrow}$ which refers to the j-th largest element in magnitude of a complex set $\{\mu_i\} \subset \mathbb{C}$, $i \geq 1$, i.e., the μ_i's are assumed to be ordered like $|\mu_1| \geq \ldots \geq |\mu_n|$. The next lemma provides useful factorizations of P and P_Ω.

Lemma 7.6 (Factorization of P and P_Ω):

Let A in (7.6) and (7.32a) be diagonalizable, i.e., there exists a nonsingular matrix $X \in \mathbb{C}^{n\times n}$ such that $A = X\Lambda X^{-1}$ with $\Lambda = \operatorname{diag}(\lambda_1, \ldots, \lambda_n)$, $\lambda_i \in \Lambda(A)$ (cf. Definition 2.2). Furthermore, assume that $(A,\ b)$ is controllable (cf. Definition 2.21).

a) The reachability Gramian P can be expressed as

$$P = X_b K X_b^H \quad \text{with} \quad X_b := X \operatorname{diag}\left(X^{-1}b\right). \tag{7.34}$$

The matrix $K = \left(\frac{-1}{\lambda_i + \overline{\lambda_j}}\right)_{i,j=1}^{n}$ is a Hermitian positive definite Cauchy matrix.

b) The frequency-limited reachability Gramian P_Ω can be factorized as

$$P_\Omega = X_b K_\Omega X_b^H, \quad \text{where} \quad K_\Omega := \Gamma K + K\Gamma^H \tag{7.35}$$

with $\Gamma = X^{-1} F_\Omega X = \operatorname{diag}(\phi_1, \ldots, \phi_n)$. The matrix K_Ω is a Hermitian positive definite Loewner matrix. ◊

Proof. The result a) is established in [4, Lemma 3.2], see also (2.28).

For b), the expression (7.30) reveals

$$\begin{aligned} P_\Omega = F_\Omega P + P F_\Omega^H &= X\Gamma X^{-1} X_b K X_b^H + X_b K X_b^H X^{-H} \Gamma^H X^H \\ &= X\Gamma \operatorname{diag}\left(X^{-1} b\right) K X_b^H + X_b K \operatorname{diag}\left(X^{-1} b\right)^H \Gamma^H X^H, \end{aligned}$$

from which the factorization follows since diagonal matrices commute. Upon closer inspection, $K_\Omega = \Gamma K + K\Gamma^H = \left(\frac{\phi_i + \overline{\phi_j}}{\lambda_i + \overline{\lambda_j}}\right)_{i,j=1}^{n}$ is obviously a Hermitian Loewner matrix [95], which inherits the positive definiteness from $P_\Omega \succ 0$. □

For every product FTL of three matrices F, T, L of appropriate dimensions, there is the well known result [132, Theorem 3.3.2] that $\sigma_j(FTL) \leq \|F\| \|L\| \sigma_j(T)$. Applying this to (7.34) and (7.35) yields

$$\lambda_j^\downarrow(P) \leq \zeta \lambda_j^\downarrow(K), \quad \lambda_j^\downarrow(P_\Omega) \leq \zeta \lambda_j^\downarrow(K_\Omega) \tag{7.36}$$

with $\zeta := \|X_b\|^2$. Here, we used that for any Hermitian positive definite matrix, its eigenvalues in a decreasing order coincide with its singular values. Hence, the eigenvalues of P and P_Ω are bounded by the eigenvalues of K and K_Ω, respectively. However, as $\|X_b\|$ can be arbitrarily large, e.g., when A is non-normal ($\kappa(X) > 1$), there might be a large deviation between $\lambda_j^\downarrow(P)$, $\lambda_j^\downarrow(P_\Omega)$ and $\lambda_j^\downarrow(K)$, $\lambda_j^\downarrow(K_\Omega)$. The effect of non-normality to CALE solutions is investigated from a different perspective in [7]. At this point we stress out that the focus of this section is not to give a precise estimation of the eigenvalue decay of the Gramians, but to relate $\lambda_j^\downarrow(P)$ to $\lambda_j^\downarrow(P_\Omega)$ in the sense that $\lambda_j^\downarrow(K_\Omega)$ can be bounded by $\lambda_j^\downarrow(K)$. For this, the following bound can be readily established.

Lemma 7.7:

For the eigenvalues of the matrices K and K_Ω in (7.34) and (7.35), respectively, it holds for $j = 1, \ldots, n$ that

$$\lambda_j^\downarrow(K_\Omega) \leq 2|\lambda_j^\downarrow(\Gamma)| \lambda_j^\downarrow(K) \leq 2\rho(\Gamma) \lambda_j^\downarrow(K),$$

and, thus, $\lambda_j^\downarrow(P_\Omega) \leq 2\zeta |\phi_j^\downarrow| \lambda_j^\downarrow(K)$. ◊

Proof. Recall that, for any matrix $M \in \mathbb{C}^{n\times n}$ its Hermitian part is $H_M := \frac{1}{2}(M + M^H)$. The eigenvalues of H_M can be bounded by the singular values of M by [132, Corollary 3.1.5]. Obviously, K_Ω is twice the Hermitian part of $N := \Gamma K$, i.e, $K_\Omega = 2H_N = N + N^H$ and a straightforward application of the said corollary and [132, Theorem 3.3.2] yields $\lambda_j^{\downarrow}(K_\Omega) \leq 2\sigma_j(N) \leq 2\sigma_j(\Gamma)\sigma_j(K)$, which leads to the result since both Γ is diagonal and K Hermitian positive definite. □

This result should by no means be understood as very accurate because [132, Corollary 3.1.5] introduces a large over estimation regarding the magnitudes of $\lambda_j^{\downarrow}(K_\Omega)$. To the authors knowledge, there are no tighter bounds for the eigenvalues of the Hermitian part of a matrix available. Nonetheless, Lemma 7.7 reveals that the spectral radius $\rho(\Gamma) = \rho(F_\Omega)$ has a huge influence on the eigenvalues of K_Ω. In the next lemma we derive insightful bounds for the eigenvalues and the spectral radius of F_Ω.

Lemma 7.8 (Eigenvalue bound for F_Ω):

Let A satisfy the same assumptions as above and define

$$\hat{\rho} := |\lambda_q|, \quad \hat{\eta} := |\operatorname{Im}(\lambda_q)|, \quad \lambda_q := \operatorname*{argmax}_{\lambda\in\lambda(A)} \left|\frac{\operatorname{Im}(\lambda)}{\operatorname{Re}(\lambda)}\right|,$$

i.e., λ_q is the eigenvalue where the opening angle $\psi(A)$ of $\Lambda(A)$ is attained (cf. (2.27) in Section 2.3.3). Then F_Ω is nonsingular and for $\phi_j \in \Lambda(F_\Omega)$, $j = 1, \ldots, n$, it holds that

$$0 < \operatorname{Re}(\phi_j) < \frac{1}{2}, \tag{7.37a}$$

$$|\operatorname{Im}(\phi_j)| < \tfrac{1}{4\pi} \ln \tfrac{\hat{\rho}+\hat{\eta}}{\hat{\rho}-\hat{\eta}} =: \iota \tag{7.37b}$$

and, consequently, $\rho(F_\Omega) = \rho(\Gamma) < \frac{1}{2}\sqrt{1+4\iota^2}$. ◊

Proof. Since A is assumed to be diagonalizable, we have

$$\begin{aligned} F_\Omega &= \operatorname{Re}\left(\frac{\jmath}{\pi}\ln\left(\mathcal{C}(A, \jmath\omega_1, \jmath\omega_2)\right)\right) = \operatorname{Re}\left(\frac{\jmath}{\pi} X \ln\left(\operatorname{diag}(\theta_1, \ldots, \theta_n)\right) X^{-1}\right) \\ &= \operatorname{Re}\left(X \operatorname{diag}\left(\hat{\phi}_1, \ldots, \hat{\phi}_n\right) X^{-1}\right), \end{aligned}$$

where

$$\hat{\phi}_j := \frac{\jmath}{\pi}\left(\ln|\theta_j| + \jmath \arg\theta_j\right), \quad \theta_j = \frac{\lambda_j + \jmath\omega_2}{\lambda_j + \jmath\omega_1} \in \Lambda(\mathcal{C}(A, \jmath\omega_1, \jmath\omega_2)), \ \lambda_j \in \Lambda(A).$$

Since $\theta_j \neq 1$ it holds $\hat{\phi}_j \neq 0\ \forall j$, which proves the non-singularity of F_Ω. Furthermore, the eigenpairs of A occur either in the form $(\lambda_j,\ x_j) \in \mathbb{R}_- \times \mathbb{R}^n$ or as two complex conjugate pairs (λ_j, x_j), $(\lambda_{j+1} = \overline{\lambda}_j,\ x_{j+1} = \overline{x}_j) \in \mathbb{C}_- \times \mathbb{C}^n$. Hence, there exists a block-diagonal, nonsingular matrix $T = \operatorname{diag}(T_1, \ldots, T_n)$ with $T_j = 1$ if $\lambda \in \mathbb{R}_-$, and $T_j = \left[\begin{smallmatrix} 1 & -\jmath \\ 1 & \jmath \end{smallmatrix}\right]$ if $\lambda_j \in \mathbb{C}_-$, such that $X_\mathbb{R} := XT \in \mathbb{R}^{n\times n}$ and, hence,

$$F_\Omega = X_\mathbb{R} \operatorname{Re}\left(T^{-1} \operatorname{diag}\left(\hat{\phi}_1, \ldots, \hat{\phi}_n\right) T\right) X_\mathbb{R}^{-1}. \tag{7.38}$$

Let us at first investigate the case of real eigenvalues λ_j for which $T_j = 1$ and the diagonal entries above are

$$\hat{\phi}_j = \phi_j := \operatorname{Re}\left(\hat{\phi}_j\right) = \frac{-\arg\theta_j}{\pi}.$$

Since

$$\theta_j = \frac{\lambda_j^2 + \omega_1\omega_2}{\lambda_j^2 + \omega_1^2} + \jmath\frac{(\omega_2 - \omega_1)\lambda_j}{\lambda_j^2 + \omega_1^2}, \tag{7.39}$$

we have $\operatorname{Re}\left(\theta_j\right) > 0$ and $\operatorname{Im}\left(\theta_j\right) < 0$ such that $-\frac{\pi}{2} < \arg\theta_j < 0$, which yields the desired result (7.37a), as well as trivially (7.37b).

For each complex conjugate pair of eigenvalues $\lambda_j,\ \overline{\lambda_j}$ the associated 2×2 block in (7.38) is

$$\begin{aligned}
\operatorname{Re}\left(T_j^{-1}\operatorname{diag}\left(\hat{\phi}_j, \hat{\phi}_{j+1}\right)T_j\right) &= \frac{1}{2}\operatorname{Re}\left(\begin{bmatrix}1 & 1\\ \jmath & -\jmath\end{bmatrix}\begin{bmatrix}\hat{\phi}_j & 0\\ 0 & \hat{\phi}_{j+1}\end{bmatrix}\begin{bmatrix}1 & -\jmath\\ 1 & \jmath\end{bmatrix}\right)\\
&= \frac{1}{2}\operatorname{Re}\left(\begin{bmatrix}\hat{\phi}_j+\hat{\phi}_{j+1} & -\jmath(\hat{\phi}_j-\hat{\phi}_{j+1})\\ \jmath(\hat{\phi}_j-\hat{\phi}_{j+1}) & \hat{\phi}_j+\hat{\phi}_{j+1}\end{bmatrix}\right)\\
&= \frac{1}{2\pi}\begin{bmatrix}-\arg\theta_j-\arg\theta_{j+1} & \ln|\theta_j|-\ln|\theta_{j+1}|\\ -\ln|\theta_j|+\ln|\theta_{j+1}| & -\arg\theta_j-\arg\theta_{j+1}\end{bmatrix}\\
&= \frac{1}{2\pi}\begin{bmatrix}-\arg\theta_j\cdot\theta_{j+1} & \ln|\theta_j/\theta_{j+1}|\\ -\ln|\theta_j/\theta_{j+1}| & -\arg\theta_j\cdot\theta_{j+1}\end{bmatrix}.
\end{aligned}$$

Hence, F_Ω has the eigenvalues

$$\{\phi_j,\ \phi_{j+1} = \overline{\phi_j}\} = \left\{\frac{-1}{2\pi}\left(\arg\left(\theta_j\cdot\theta_{j+1}\right) \mp \jmath\ln|\theta_j/\theta_{j+1}|\right)\right\}$$

corresponding to each complex pair of eigenvalues $\{\lambda_j,\ \overline{\lambda_j}\} \subset \Lambda(A)$. It holds

$$\begin{aligned}
\theta_j\theta_{j+1} = \frac{1}{z}\Big(&(|\lambda_j|^2-\omega_2^2)(|\lambda_j|^2-\omega_1^2) + 4\omega_1\omega_2\operatorname{Re}\left(\lambda_j\right)^2\\
&+\jmath 2\operatorname{Re}\left(\lambda_j\right)(|\lambda_j|^2+\omega_1\omega_2)(\omega_2-\omega_1)\Big)
\end{aligned} \tag{7.40}$$

with $z = (|\lambda_j|^2-\omega_1^2)^2 + 4\omega_1^2\operatorname{Re}\left(\lambda_j\right)^2$ and we find $\operatorname{Im}\left(\theta_j\theta_{j+1}\right) < 0$ such that $-\pi < \arg\theta_j\theta_{j+1} < 0$ from which $\operatorname{Re}\left(\phi_j\right) = \operatorname{Re}\left(\overline{\phi_j}\right) = -\frac{1}{2\pi}\arg\left(\theta_j\theta_{j+1}\right) < \frac{1}{2}$ follows. For the imaginary parts of ϕ_j assume w.l.o.g. $\operatorname{Im}\left(\lambda_j\right) > 0$ and consider

$$0 < \Theta_j := \left|\frac{\theta_j}{\theta_{j+1}}\right|^2 = \left(\frac{|\lambda_j|^2+\omega_1^2-2\operatorname{Im}\left(\lambda_j\right)\omega_1}{|\lambda_j|^2+\omega_1^2+2\operatorname{Im}\left(\lambda_j\right)\omega_1}\right)\left(\frac{|\lambda_j|^2+\omega_2^2+2\operatorname{Im}\left(\lambda_j\right)\omega_2}{|\lambda_j|^2+\omega_2^2-2\operatorname{Im}\left(\lambda_j\right)\omega_2}\right) = \zeta_1\zeta_2$$

with $0 < \zeta_1 \le 1 < \zeta_2$. Hence,

$$|\ln\Theta_j| < \max\left(\ln\max\Theta_j, |\ln\min\Theta_j|\right).$$

Now , ζ_2 is, for a fixed λ_j, maximal if $\omega_2 = |\lambda_j|$. In that case

$$\zeta_2 = \frac{|\lambda_j| + \operatorname{Im}(\lambda_j)}{|\lambda_j| - \operatorname{Im}(\lambda_j)} = \frac{\sqrt{1+q_j^2}+q_j}{\sqrt{1+q_j^2}-q_j}, \quad q_j := \frac{\operatorname{Im}(\lambda_j)}{|\operatorname{Re}(\lambda_j)|},$$

which increases as q_j increases and, hence, the maximum value of ζ_2 is attained at $q_{\max} = \frac{\operatorname{Im}(\lambda_q)}{|\operatorname{Re}(\lambda_q)|}$, i.e., for λ_q. Using also $\max \zeta_1 = 1$ (attained at $\omega_1 = 0$), yields

$$\max \Theta_j < \left(\frac{\hat{\rho}+\hat{\eta}}{\hat{\rho}-\hat{\eta}}\right).$$

Furthermore, $\min \zeta_2 = 1$ and, for a fixed λ_j, ζ_1 is by a similar reasoning minimal if $\omega_1 = |\lambda_q|$. This leads finally to

$$|\ln \Theta_j| < \max\left(\ln \frac{\hat{\rho}+\hat{\eta}}{\hat{\rho}-\hat{\eta}}, \left|\ln \frac{\hat{\rho}-\hat{\eta}}{\hat{\rho}+\hat{\eta}}\right|\right)$$

from which (7.37b) follows. □

Because the proof deals with the real and imaginary parts of the ϕ_i independently, the results represent upper bounds for the largest attainable real, imaginary part and spectral radius. Also, if $\omega_1 = 0$, the bound for $\rho(\Gamma)$ can be slightly altered to $\rho(\Gamma) < \frac{1}{4}\sqrt{1+16\iota^2}$. Together with Lemma 7.7 these bounds reveal how the spectral radius of Γ influences the eigenvalues of K_Ω. In particular, we can deduce possibilities when the $\lambda_j^{\downarrow}(K_\Omega)$ are significantly smaller than the $\lambda_j^{\downarrow}(K)$. For matrices A with real spectra, F_Ω also has only real eigenvalues with $0 < \phi_j < \frac{1}{2}$ by Lemma 7.8. Thus, $\Gamma \succ 0$ such that the bound in Lemma 7.7 becomes $\lambda_j^{\downarrow}(K_\Omega) \leq 2\phi_j^{\downarrow}\lambda_j^{\downarrow}(K)$, for $j = 1, \ldots, n$. Obviously, if the considered frequency interval is small, i.e., $\omega_2 - \omega_1$ is small compared to $\frac{\lambda_j}{\lambda_j^2+\omega_1\omega_2}$, then $\operatorname{Im}(\theta_j)$ will be close to zero and so will $\arg \theta_j$.

If $\Lambda(A)$ has complex eigenvalues the situation is considerable more subtle, but the proof of Lemma 7.8 already indicates that setting the interval limits ω_1, ω_2 equal or close to absolute values of eigenvalues of A can increase the spectral radius $\rho(F_\Omega)$. In particular, setting the interval limits close to $|\lambda_q|$ will lead to the largest values of ι in (7.37b) and, thus, to large spectral radii. This is observed in numerical experiments but, however, even if $\rho(F_\omega) > \frac{1}{2}$, the eigenvalues of K_Ω seem to be never much greater then the ones of P.

To conclude, we expect that the values of $\lambda_j^{\downarrow}(K_\Omega)$ will be noticeably smaller than $\lambda_j^{\downarrow}(K)$ if the chosen interval $[\omega_1, \omega_2]$ is small compared to the spectral radius of A and if the interval boundaries ω_1, ω_2 are not close to the magnitude of the eigenvalues of A whose imaginary parts dominate their real parts. This argumentation will also carry over to $\lambda_j^{\downarrow}(P_\Omega)$ and $\lambda_j^{\downarrow}(P)$ by (7.36).

The main conclusion of this section is that although

$$\operatorname{rank}\left(F_\Omega BB^T + BB^T F_\Omega^T\right) = 2\operatorname{rank}\left(BB^T\right),$$

under reasonable and quantifiable assumptions on the interval $[\omega_1,\ \omega_2]$, we can expect that the eigenvalues of P_Ω decay at a faster rate than those of P. Hence, P_Ω has a smaller numerical rank than P. Consequently, we can expect that there exist low-rank solution factors Z_P and Z_{P_Ω} leading to low-rank solutions of comparable accuracy in the sense

$$\|P - Z_P Z_P^H\| \approx \|P_\Omega - Z_{P_\Omega} Z_{P_\Omega}^H\|$$

but with $\operatorname{rank}(Z_P) \geq \operatorname{rank}\left(Z_{P_\Omega}\right)$. This is also confirmed by our numerical experiments. The same holds trivially also for low-rank approximations of Q and Q_Ω. Moreover, it will turn out later that in some cases computing Z_{P_Ω}, Z_{Q_Ω} is less costly than computing Z_P, Z_Q, which can even make FLBT numerically cheaper than the standard (unlimited) BT. Algorithms for computing the low-rank solution factors are topic of the next section.

Remark 7.9:

a) For the standard CALEs (7.6), the above considerations are continued in [4], where a square-root free Cholesky factorization of the form $K = L\Delta L^H$ with a unit lower triangular L and $\Delta = \operatorname{diag}\left(\delta^{(1)}, \ldots, \delta^{(n)}\right) \succ 0$ is used. There are explicit formulas [108] for the diagonal entries $\delta^{(i)}$, which appear to decay to zero at a similar rate as the eigenvalues of P, especially if A is not too far from normal. A square-root free Cholesky factorization also exists for the frequency-limited Gramian: $K_\Omega = L_\Omega \Delta_\Omega L_\Omega^H$ with $\Delta_\Omega = \operatorname{diag}\left(\delta_\Omega^{(1)}, \ldots, \delta_\Omega^{(n)}\right) \succ 0$. A basic Cholesky algorithm [111] can be employed to find how the entries of L_Ω and Δ_Ω are built from the entries of L, Δ, and Γ. It is easy to show that $\delta_\Omega^{(1)} = 2\operatorname{Re}(\phi_1)\,\delta^{(1)} < \delta^{(1)}$, but the calculations for the remaining entries become very tedious and lengthy such that we do not report them here as this would clutter the presentation.

b) In Corollary 2.41 we mentioned another well known theoretical result from [113] regarding the existence of low-rank solutions of linear matrix equations. Following [113], low-rank solutions of P and P_Ω are given by $P_k := \sum_{i=-k}^{k} \tilde{\omega}_i B_i B_i^T$, $B_i := \exp(\tilde{t}_i A)B$ and $P_{\Omega,k} := \sum_{i=-k}^{k} \tilde{\omega}_i \tilde{B}_i F_m \tilde{B}_i^T$, $\tilde{B}_i := \exp(\tilde{t}_i A)[B,\ B_\Omega]$, respectively, with the *flipping matrix*

$$F_h := \left[\begin{smallmatrix} 0 & 1 \\ 1 & 0 \end{smallmatrix}\right] \otimes I_h. \tag{7.41}$$

Using (2.25) the approximation errors can be bounded by

$$\|P - P_k\| \leq \zeta_{\text{Lyap}} \pi^{-1} \|B\|^2, \quad \|P_\Omega - P_{\Omega,k}\| \leq \zeta_{\text{Lyap}} \pi^{-1} \|[B,\ B_\Omega] F_m [B,\ B_\Omega]^T\|,$$

where the constant ζ_{Lyap} depends entirely on A (cf. (2.26)). Obviously, the bound $\operatorname{rank}\left(P_{\Omega,k}\right) \leq 2(k+1)m$ is larger than $\operatorname{rank}(P_k) \leq (k+1)m$ and the above error bounds differ only by the norms of the inhomogeneities of the CALEs (7.6), (7.32a). Notice that the difference in the decay rates of the eigenvalues of P and P_Ω can be also observed if the inhomogeneities would be scaled to unit norm. Hence, this approach offers no useful explanation why P_Ω can in practice often be approximated by low-rank solutions of smaller rank compared to P. ◊

7.3.3. Numerical Methods for Computing the Low-Rank Approximations

Motivated by the expected low numerical rank of $P_\Omega,\ Q_\Omega$, we aim at computing, as in standard BT, low-rank approximations $P_\Omega \approx Z_{P_\Omega} Z_{P_\Omega}^T$, $Q_\Omega \approx Z_{Q_\Omega} Z_{Q_\Omega}^T$ with $Z_{P_\Omega} \in \mathbb{R}^{n\times r_1}$, $Z_{Q_\Omega} \in \mathbb{R}^{n\times r_2}$, $r_1,\ r_2 \ll n$. Before the frequency-limited CALEs can be approached by numerical methods which compute low-rank solution factors, the matrix F_Ω has to be treated. This is the subject of the next subsection. After that, some strategies for computing the low-rank solution factors Z_{P_Ω}, Z_{Q_Ω} will be discussed.

Dealing with the Matrix-Valued Logarithm

In FLBT, the matrix F_Ω requires the evaluation of a matrix-valued function f in A. Most state-of-the-art algorithms for that purpose work, e.g., with the Schur form of A and additional matrix multiplications [128]. For the matrix-logarithm, a very robust and often applied method is the *inverse scaling and squaring algorithm* [128, Chapter 11] and its variants. The method is called by the MATLAB routine `logm`. However, as computing the Schur form has a cubic complexity and quadratic memory demands, these approaches are not feasible for the large-scale case we are interested in.

If we plan to obtain the frequency-limited Gramians P_Ω, Q_Ω via approximate solutions of the frequency-limited CALEs (7.32), we observe that only

$$B_\Omega := F_\Omega B = f(A)B \quad \text{and} \quad C_\Omega := CF_\Omega = Cf(A) \tag{7.42}$$

are required for setting up the inhomogeneities in (7.32). Hence, only the products $f(A)B$ and $f(A)^H C^T$, i.e., m and p matrix-vector products with $f(A)$ and $f(A)^H$, respectively, are needed. Although $f(A)$ is still involved, computing the matrix-vector products of the form

$$w = f(A)v, \quad w, v \in \mathbb{C}^n \tag{7.43}$$

constitutes a much more attractive problem to overcome, even for large matrices A, see, for instance, [128, Chapter 13], [146, 73, 104, 147, 123] and the references therein, which provide several efficient numerical methods for this task.

Remark 7.10:

The computation of the frequency-limited reachability Gramians P_Ω by (7.30) and using a low-rank solution factor Z_P of the ordinary Gramian P can also be reduced to the problem of computing the matrix-vector products of $f(A)$ with $Z_P \in \mathbb{R}^{n\times k_P}$ and similarly the problem of computing Q_Ω by (7.30) using $f(A)^H Z_Q \in \mathbb{R}^{n\times k_Q}$. This approach is in general more expensive than using (7.42) and subsequently solving (7.32) because, in general, $k_P > m$ and $k_Q > p$, such that significantly more matrix-vector products with $f(A)$ would be required. ◊

In the following we discuss some approaches for (7.43) and their usage in frequency-limited balanced truncation.

Quadrature Based Approaches Recall that F_Ω was at first defined as the integral (7.31) which simplifies to (7.33a). These integrals are also obtained by using the integral expression [190]

$$\ln M = \int_0^1 (M - I_n)\left(t(M - I_n) + I_n\right)^{-1} \mathrm{d}t, \quad \forall M \in \mathbb{C}^{n\times n}, \quad \Lambda(M) \cap \mathbb{R}_- = \emptyset$$

and setting $M = \mathcal{C}(A, \jmath\omega_1, -\jmath\omega_2)$. Exemplary, for $B_\Omega = F_\Omega B$, one can approximate (7.53) by means of a quadrature rule, i.e.,

$$B_\Omega = \frac{1}{\pi}\,\mathrm{Re}\left(\int_{\omega_1}^{\omega_2} (\jmath\nu I_n - A)^{-1} B \mathrm{d}\nu\right) \approx \frac{1}{\pi}\,\mathrm{Re}\left(\sum_{k=1}^{h} \zeta_k (\jmath\nu_k I_n - A)^{-1} B\right) \tag{7.44}$$

using quadrature nodes $\nu_k \in [\omega_1,\ \omega_2]$, and weights ζ_k, $k = 1, \ldots, h$ whose choice depends on the selected quadrature approach. In principle, any quadrature rule can be applied, where for reasons of accuracy, as well as efficiency, a method using an adaptive selection of nodes and weights is typically chosen. The `integral` command in MATLAB, e.g., employs adaptive Gauss-Kronrod quadrature [141] and will be used in our numerical examples. Using (7.44) requires the solution of h shifted linear systems with m right hand sides, which might easily become expensive, depending on the number of quadrature nodes. In [136], numerical quadrature is applied directly to the integrals (7.53) to obtain low-rank solution factors of $P_\Omega,\ Q_\Omega$. The POD approach in [93] is analogous.

Projection Type Methods A further, often successfully applied, and investigated approach for the matrix function times vector problem (7.43) is to use projections onto low-dimensional subspaces. Let $\mathcal{Q} \subset \mathbb{C}^n$ be a subspace with $\dim \mathcal{Q} = k \ll n$ and let $Q_k = [q_1, \ldots, q_k] \in \mathbb{C}^{n\times k}$ with $\{q_1, \ldots, q_k\}$ being an orthonormal basis of $\mathcal{Q}$. Typically, $v \in \mathcal{Q}$ such that $v = Q_k \tilde{v}_k$ with $\tilde{v}_k := Q_k^H v \in \mathbb{C}^k$. Approximating w by its orthogonal projection onto $\mathcal{Q}$ gives

$$w \approx Q_k Q_k^H w = Q_k \tilde{w} \in \mathcal{Q}, \quad \tilde{w}_k = Q_k^H w \in \mathbb{C}^k.$$

Imposing a Ritz-Galerkin condition on this approximation yields

$$Q_k \tilde{w}_k - f(A) Q_k \tilde{v}_k \quad \perp \quad \mathcal{Q} \qquad \Leftrightarrow \quad \tilde{w}_k = Q_k^H f(A) Q_k \tilde{v}_k \approx f(\tilde{A}_k)\tilde{v}_k$$

with $\tilde{A}_k := Q_k^H A Q_k$. Hence, the approximate result can be computed by

$$w_k := Q_k f(\tilde{A}_k) Q_k^H v.$$

Due to the small size of $\tilde{A}_k \in \mathbb{C}^{k\times k}$, the computation of $f(\tilde{A}_k)$ can then be carried out using methods for small, dense problems, e.g., the inverse scaling and squaring method discussed earlier in this section for our particular application. The quality of

the approximation w_k depends on how good $Q_k f(\tilde{A}_k) Q_k^H v$ approximates $f(A)v$ which, for general matrices A and functions f, is difficult to predict. The above projection framework is usually carried out in an iterative manner using a sequence of nested subspaces $\mathcal{Q}_1 \subseteq \mathcal{Q}_2 \subseteq \ldots \subseteq \mathcal{Q}_k \subseteq \mathcal{Q}_{k+1}$ with increasing dimensions. To stop this iteration, we will employ the simple stopping criterion

$$\|w_k - w_{k-1}\| \leq \tau_f \|w_k\|. \tag{7.45}$$

For certain choices of $\mathcal{Q}$, special matrices, and functions, more advanced stopping tests as well as error bounds can be found, e.g., in [128, Chapter 13.2], [104, 147, 123, 139]. The the sequence of subspaces can be constructed in different ways and we will restrict ourselves to the most common ones, e.g., similar approaches as we used before in Section 5.3.3.

Setting up the subspace $\mathcal{Q}$ as standard Krylov subspace

$$\mathcal{K}_k(A,\ v) = \operatorname{span}\left\{v, Av, A^2v, \ldots, A^{k-1}v\right\}$$

is used for the approximation of (7.43) in, e.g., [146, 197]. The construction of the matrices Q_k, $\tilde{A}_k$ is usually done by the Lanczos or Arnoldi algorithm [111] for Hermitian and, respectively, non-Hermitian A. It can be shown that with $\mathcal{K}_k$, the function f is approximated by a polynomial p_{k-1} of degree at most $k-1$ whose roots are the eigenvalues of $\tilde{A}_k$. To obtain a good approximation of (7.43), often large dimensions k are needed, which makes this approach less practical.

Rational Krylov subspaces [83, 123] often provide much better approximations with smaller subspace dimension k. They can by defined via

$$\mathcal{K}_k^{\text{rat}}(A,\ v) = d_{k-1}(A)^{-1} \mathcal{K}_k(A,\ v)$$

with the denominator polynomial $d_{k-1}(z) = \prod_{j=1}^{k-1} (1 - \frac{z}{\xi_j})$ of degree $k-1$ having the poles $\xi_1, \ldots, \xi_{k-1} \in \mathbb{C} \cup \{\infty\}$. Hence, with $\mathcal{K}_k^{\text{rat}}$ the function f is approximated by a rational function $r_k = p_{k-1}/d_{k-1}$. The poles ξ of r_k are typically referred to as shifts for the rational Krylov subspace. The orthonormal basis for $\mathcal{K}^{\text{rat}}$ can be constructed by the rational Arnoldi algorithm [193] whose main numerical costs occur at solving linear systems of the form $(I - A/\xi)s = u$ for s.

The shifts are crucial for a fast convergence and good approximation results. A comprehensive overview addressing various choices of a priori selected shifts for several functions f can be found in [123]. In [83], an adaptive strategy is proposed, where the shift ξ_{k+1} is computed from the data available after the rational Arnoldi iteration step k is completed. The main idea is to consider the greedy method,

$$\xi_{k+1} = \operatorname*{argmax}_{s \in \mathcal{D}_k} |r_k(s)|^{-1},$$

where $\mathcal{D}_k$ is a set of discrete points from the boundary of the convex hull of $-\Lambda(\tilde{A}_k)$. As in the experiments in Section 5.3.3, we will use this adaptive shift generation in the upcoming numerical examples.

In our situation f can be represented as integral (see (7.27)) of $\Psi(\jmath\nu)$ within the integration domain Ω and, thus, it appears reasonable to restrict the shifts to the imaginary region $\jmath\Omega$. Therefore, we propose as modifications of the above adaptive shift strategy to choose the set $\mathcal{D}_k$ as set of discrete points from the purely imaginary interval $\jmath[\omega_1,\ \omega_2]$. In most examples this leads to better results compared to the adaptive shifts based on the convex hull of $-\Lambda(\tilde{A}_k)$. A simplification of this approach is to simply use the shifts $\xi_{2k} = \jmath\omega_1$ and $\xi_{2k-1} = \jmath\omega_2$ alternatingly.

Choosing the shifts $\xi_{2k} = \infty$ and $\xi_{2k-1} = 0$ in an alternating fashion yields the extended Krylov subspace $\mathcal{K}_k^{\text{ext}}(A, v)$ proposed in [81] and further investigated in [147]. In [209], an efficient algorithmic framework for constructing the basis and representation matrix Q_k, $\tilde{A}_k$ is established. One advantage of that choice is that the coefficient matrix of the linear system to solve does not change such that one can store and reuse a factorization of A if direct solvers are applied. This can also be done if the shifts ξ for $\mathcal{K}^{\text{rat}}$ are constant which yields the shift-and-invert Krylov subspace $\mathcal{K}_k^{\text{SI}}(A,\ v)$ [176, 224]. For the alternate use of $\jmath\omega_1$, $\jmath\omega_2$, two sparse matrix factorizations might be saved. We end this short introduction to Krylov subspace methods for $v = f(A)w$ with a number of comments regarding actual implementations. For the numerical tests in this section, we will use a version of Algorithm A.1 in the appendix, modified to tackle also the problem (7.43).

Remark 7.11:

1. As before, complex shifts ξ are dealt with by the modification of the rational Arnoldi process proposed in [195]. This will also yield a real restriction $\tilde{A}_k$. Using this modification is, of course, reasonable because the number of complex arithmetic operations is reduced. In our application, since establishing the relation (7.33b) from (7.31) relied heavily on the fact that the matrix A in $\Psi(\jmath\nu) = (\jmath\nu I - A)^{-1}$ is real, having a real restriction $\tilde{A}_k$ appears to be also important if we want to safely employ (7.33b) to $\tilde{A}_k$. Issues with complex shifts are not present in methods using standard or extended Krylov subspaces.

2. For the case $m > 1$ we utilize block versions of $\mathcal{K}^{\text{rat}}$ and $\mathcal{K}^{\text{ext}}$. Alternatives might be global Arnoldi methods [218] or, in the case of rational Krylov subspaces, tangential approaches [84].

3. In our application we also need matrix vector products $z = f(A)^T u$ of the transposed matrix function for computing $C_\Omega = CF_\Omega$. Thus, the above approaches have to be also applied using A^T and u, leading to a subspace $\mathcal{Z}$ generated by A^T and u, e.g.,

$$\mathcal{Z} = \mathcal{K}_k(A^T,\ u) = \text{span}\left\{u, A^T u, \ldots, (A^T)^{k-1}\right\}$$

and the dual versions of $\mathcal{K}^{\text{rat}}$, $\mathcal{K}^{\text{ext}}$ are easily deduced. It is possible to utilize a two-sided Petrov-Galerkin condition [130] and construct $\mathcal{Z}$ bi-orthogonal to $\mathcal{Q}$. Both spaces can be generated simultaneously with the two-sided Lanczos process or modifications thereof. Further details on generating dual subspaces $\mathcal{Q}$, $\mathcal{Z}$ can, e.g., be found in [179, 114, 117]. $\Diamond$

Computing Low-Rank Solution Factors of the Frequency-Limited Gramians

Having computed approximations of B_Ω and C_Ω, the frequency-limited CALEs (see also (7.32))

$$AP_\Omega + P_\Omega A^T + B_\Omega B^T + BB_\Omega^T = 0, \tag{7.46a}$$
$$A^T Q_\Omega + Q_\Omega A + C_\Omega^T C + C^T C_\Omega = 0 \tag{7.46b}$$

have to be solved for low-rank approximations. Any low-rank solution algorithm for large-scale CALEs can be employed here. In the following, we will discuss extended and rational Krylov subspace methods as well as the LR-ADI iteration from the previous chapters for this task. We will mainly focus the treatment of the frequency-limited reachability CALE since the frequency-limited observability CALE can be dealt with analogously.

Using Krylov Subspace and Related Methods

In a similar way as we discussed in Section 7.3.3, we can use a projection approach for solving the frequency-limited CALEs. Having a low-dimensional subspace $\mathcal{Q}_k = \operatorname{span}\{Q_k\}$ constructed, the low-rank solution is obtained as $P_\Omega \approx P_{\Omega,k} := Q_k \tilde{P}_{\Omega,k} Q_k^T$, where $\tilde{P}_{\Omega,k}$ is the solution of the projected frequency-limited CALE

$$\tilde{A}_k \tilde{P}_{\Omega,k} + \tilde{P}_{\Omega,k} \tilde{A}_k^T + \tilde{B}_{\Omega,k} \tilde{B}_k^T + \tilde{B}_k \tilde{B}_{\Omega,k}^T = 0, \qquad \tilde{B}_k := Q_k^T B. \tag{7.47}$$

Due to its small dimension, it can be solved by direct methods. Choices for $\mathcal{Q}_k$ include the same possibilities as for the matrix-function evaluations: standard [137], extended [209], as well as rational Krylov subspaces [83, 84], generated using A, B, and possibly a collection of (adaptively computed) shift parameters. If $P_{\Omega,k}$ is not accurate enough, the particular Krylov process used is continued.

Now assume B_Ω is approximated by such an projection approach, i.e., $B_\Omega \approx B_{\Omega,k} := Q_k \tilde{B}_{\Omega,k}$ with $\tilde{B}_{\Omega,k} := f(\tilde{A}_k) Q_k^T B$, $\tilde{A}_k := Q_k^T A Q_k$, where Q_k is a real orthogonal matrix, which spans one of the aforementioned (rational or extended) Krylov subspaces. In that case an obvious strategy is to reuse the information contained in the basis matrix Q_k and continue the Krylov method for solving the frequency-limited CALE. In the majority of our numerical tests only very few additional iteration steps of the employed Krylov subspace method were necessary to obtain the desired accuracy for P_Ω once an accurate $B_{\Omega,k}$ was found. Often, accurate approximations $B_{\Omega,k}$ and $P_{\Omega,k}$ were obtained in the same iteration step k, which makes this approach exceptionally efficient. Provided that $\tilde{P}_{\Omega,k} \geq 0$ in the projected equation (7.47), low-rank solution factors of P_Ω are given by $Z_{P_{\Omega,k}} = Q_k L_k$, where L_k is a lower triangular Cholesky factor of $\tilde{P}_{\Omega,k}$. Alternatively, an eigendecomposition of $\tilde{P}_{\Omega,k}$ can be used which also enables a rank truncation, see, e.g., [209]. In Algorithm 7.3, we illustrate this strategy for approximating B_Ω and P_Ω in a single Krylov subspace algorithm. In Lines 1 and 14, `orth` should be understood as any stable (block) orthogonalization routine. By adjusting the basis generation in Line 14 appropriately, any version of extended, rational or adaptive rational Krylov

Algorithm 7.3: Krylov subspace method for frequency-limited CALEs (7.46a)

Input : $A,\ B,\ [\omega_1, \omega_2]$ as in (7.46a), tolerances $0 < \tau_f,\ \tau_P \ll 1$.
Output: $\hat{Z}_{P_\Omega,j} \in \mathbb{R}^{n \times k_{P_\Omega}}$ such that $\hat{Z}_{P_\Omega,j}(\hat{Z}_{P_\Omega,j})^T \approx P_\Omega$ with $k_{P_\Omega} \leq mj \ll n$.

1 $Q_1 = \operatorname{orth}(B)$.
2 **for** $j = 1, 2, \ldots$ **do**
3 $\quad \tilde{A}_j = Q_j^T A Q_j,\ \tilde{B}_j = Q_j^T B$.
4 $\quad \tilde{B}_{\Omega,j} = \operatorname{Re}\left(\frac{1}{\pi} \ln\left(\mathcal{C}(\tilde{A}_j,\ \jmath\omega_1,\ \jmath\omega_2)\right)\tilde{B}_j\right),\ B_{\Omega,j} = Q_j \tilde{B}_{\Omega,j}$.
5 $\quad$ **if** $\|B_{\Omega,j} - B_{\Omega,j-1}\|/\|B_{\Omega,j}\| < \tau_f$ **then**
6 $\quad\quad$ Solve
7 $\quad\quad$ $\tilde{A}_j \tilde{P}_{\Omega,j} + \tilde{P}_{\Omega,j} \tilde{A}_j^T + \tilde{B}_{\Omega,j} \tilde{B}_j^T + \tilde{B}_j \tilde{B}_{\Omega,j}^T = 0$ for $\tilde{P}_{\Omega,j}$.
8 $\quad\quad$ Set
9 $\quad\quad$ $\xi_j := \left\| B_{\Omega,j} B^T + B B_{\Omega,j}^T \right\| = \left\| \tilde{B}_{\Omega,j} \tilde{B}_j^T + \tilde{B}_j \tilde{B}_{\Omega,j}^T \right\|$ and
$\quad\quad$ $\mathcal{L}_j := \left\| A\left(Q_j \tilde{P}_{\Omega,j} Q_j^T\right) + \left(Q_j \tilde{P}_{\Omega,j} Q_j^T\right) A^T + B_{\Omega,j} B^T + B B_{\Omega,j}^T \right\| / \xi_j$.
10 $\quad\quad$ **if** $\mathcal{L}_j < \tau_P$ **then**
11 $\quad\quad\quad$ Compute (and truncate) Eigendecomposition $\tilde{P}_{\Omega,j} = \tilde{X}_j \tilde{\Lambda}_{\Omega,j} \tilde{X}_j^T$,
$\quad\quad\quad$ $\tilde{X}_j^T \tilde{X}_j = I_{mj},\ \tilde{\Lambda}_{\Omega,j} = \operatorname{diag}\left(\tilde{\lambda}_1, \ldots, \tilde{\lambda}_{mj}\right)$.
12 $\quad\quad\quad$ Construct low-rank solution factors $\hat{Z}_{P_\Omega,j} = Q_j \tilde{X}_j \tilde{\Lambda}_{\Omega,j}^{\frac{1}{2}}$.
13 $\quad\quad\quad$ Stop Krylov process.
14 $\quad$ Orthogonally extend basis matrix Q_j by new basis vectors S:
$\quad$ $Q_{j+1} = \operatorname{orth}([Q_j,\ S])$.

subspace methods (cf. Algorithm A.1) can be incorporated easily. The restriction $\tilde{A}_j$ in Line 3 can be computed efficiently without additional matrix vector products with A by using relations developed in [209] and [193] for extended and rational Krylov subspace methods, respectively. Typically one has $\operatorname{span}\{B\} \subset \operatorname{span}\{Q_j\}$ and, thus, $\tilde{B}_j = Q_j^T B = [\beta^T, 0]^T \in \mathbb{R}^{mj \times m}$ with $\beta \in \mathbb{R}^{m \times m}$ such that the inhomogeneity of the projected CALE in Line 7 is given by

$$\tilde{B}_{\Omega,j} \tilde{B}_j^T + \tilde{B}_j \tilde{B}_{\Omega,j}^T = \begin{bmatrix} \tilde{\beta}\beta^T + \beta\tilde{\beta}^T & g^T \\ g & 0 \end{bmatrix}, \quad \text{where} \quad g := \delta\beta^T, \quad \tilde{B}_{\Omega,j} = \begin{bmatrix} \tilde{\beta} \\ \delta \end{bmatrix},\ \tilde{\beta} \in \mathbb{R}^{m \times m},\ \delta \in \mathbb{R}^{(j-1)m \times m}. \tag{7.48}$$

After $\|B_{\Omega,j} - B_{\Omega,j-1}\|/\|B_{\Omega,j}\| < \tau_f$ is achieved, one can also skip the computation of newer approximations $B_{\Omega,k}$, $k > j$, in the following iterations to save some computations in Step 4. In Line 10, we employed a stopping criterion based on the scaled norm Lyapunov residual matrix. This norm can be computed efficiently without working with matrices

A, B, $B_{\Omega,j}$, Q_j of leading dimension n [209]. If the rational Krylov subspace method with the adaptive, imaginary shifts has been used to approximate $B_{\Omega,j}$, one should switch to the convex hull based adaptive shifts in the iteration steps associated to the CALE. For the approximation of the CALE solution, the LR-ADI iteration can be an alternative.

Using the LR-ADI Iteration

We can also employ the LR-ADI iteration which was extensively studied in the previous chapters. The LR-ADI iteration considered so far expects that the inhomogeneity of the CALE to be solved is given in a symmetric definite form, BB^T. However, the inhomogeneities of the frequency-limited CALEs (7.46) are given by

$$\begin{aligned} B_\Omega B^T + B B_\Omega^T &= \hat{B} F_m \hat{B}^T, \quad \hat{B} = [B,\ B_\Omega], \\ C_\Omega^T C + C^T C_\Omega &= \hat{C}^T F_p \hat{C}, \quad \hat{C} = \left[\begin{smallmatrix} C \\ C_\Omega, \end{smallmatrix}\right] \end{aligned}$$

with the flipping matrix F_h from (7.41). Since $\lambda(F_h) = \{\pm 1\}$, these inhomogeneities are in general indefinite matrices. To tackle the indefiniteness of the inhomogeneities, the LDL^T-variant [152, Algorithm 1], [153] of the LR-ADI iteration can be used. This will only introduce the following slight changes to Algorithm 3.2: the computed approximate solution and the corresponding residual after j iteration steps are of the form

$$\begin{aligned} P_\Omega \approx P_{\Omega,j}^{\mathrm{ADI}} &= Z_{P_\Omega,j} \left(I_j \otimes F_m\right) \left(Z_{P_\Omega,j}\right)^T, \\ \mathcal{L}_j(P_{\Omega,j}^{\mathrm{ADI}}) &= \|W_j F_m W_j^H\| = \lambda_{\max}(W_j^H W_j F_m). \end{aligned}$$

Although $I_j \otimes F_m$ is an indefinite matrix, we assume $P_{\Omega,j}^{\mathrm{ADI}} \succeq 0$ since $P_\Omega \succ 0$. In practice this might only hold if $P_{\Omega,j}^{\mathrm{ADI}}$ is a sufficiently accurate approximation of P_Ω. A semidefinite factorization of $P_{\Omega,j}^{\mathrm{ADI}}$ can be obtained as follows: compute a thin QR-decomposition $U_1 R = Z_{P_\Omega,j}$ followed by a spectral decomposition $U_2 \hat{\Lambda} U_2^T = R\left(I_j \otimes F_m\right) R^T$ with $U_2^T U_2 = I_{2mj}$, $\hat{\Lambda} = \operatorname{diag}\left(\hat{\lambda}_1, \ldots, \hat{\lambda}_{2mj}\right)$. Then, the approximate solution $P_{\Omega,j}^{\mathrm{ADI}}$ can be represented by a semidefinite factorization

$$P_{\Omega,j}^{\mathrm{ADI}} = \hat{Z}_{P_\Omega,j} (\hat{Z}_{P_\Omega,j})^T, \quad \hat{Z}_{P_\Omega,j} := U_1 U_2 \hat{\Lambda}^{\frac{1}{2}}.$$

By neglecting very small eigenvalues $\hat{\lambda}$ and the corresponding columns of U_2, this procedure also enables a rank truncation of the approximate solution $P_{\Omega,j}^{\mathrm{ADI}}$, similar to Line 12 in Algorithm 7.3, to get rid of nearly linearly dependent columns.

Observe that $P_\Omega = N_\Omega + N_\Omega^T$, where N_Ω solves the Sylvester equation

$$A N_\Omega + N_\Omega A^T + B B_\Omega^T = 0. \tag{7.49}$$

Hence, the modification of the factored ADI (fADI) iteration [43] given by (3.50), Algorithm 4.6, [32, Algorithm 4] can be applied directly to (7.49) and yields, after j iteration steps,

$$N_\Omega \approx N_{\Omega,j} = \tilde{Z}_{P_\Omega,j} (\tilde{Y}_{P_\Omega,j})^T, \quad \tilde{Z}_{P_\Omega,j},\ \tilde{Y}_{P_\Omega,j} \in \mathbb{R}^{n \times mj}.$$

The constructed low-rank approximation is $P_\Omega \approx [\tilde{Z}_{P_\Omega,j},\ \tilde{Y}_{P_\Omega,j}]F_{jm}[\tilde{Z}_{P_\Omega,j},\ \tilde{Y}_{P_\Omega,j}]^T$ which can be transformed into a semidefinite factorization using similar steps as above. Neglecting this transformation, both approaches are equivalent. The numerical effort of both methods is also identical and we use the LDL^T version of the LR-ADI iteration in the remainder.

The additional requirement to construct semidefinite low-rank factorizations introduces additional costs. This can be seen as a disadvantage of the LR-ADI iteration. Another shortcoming is that the information from computing B_Ω is not reused which, as observed in practice, often leads to more required iteration steps of the LR-ADI iteration compared to the projection type approach mentioned above, e.g, Algorithm 7.3.

7.3.4. Miscellaneous

Generalized State-Space Systems

Until now we only considered standard state-space systems, but everything can easily be modified to handle generalized state-space systems (2.5) with a nonsingular $I \neq E \in \mathbb{R}^{n\times n}$ by using similar techniques as in the unlimited BT framework in Section 7.2.

Corollary 7.12 (Frequency-limited Gramians for generalized systems):

For a generalized state-space system (2.5) and the frequency intervals Ω in (7.29), the frequency-limited Gramians are P_Ω and $E^TQ_\Omega E$, which are obtained from either of the following two approaches:

1. Using the solutions P and Q of the ordinary reachability and observability and GCALEs (7.7), it holds

$$P_\Omega = F_\Omega EP + PE^TF_\Omega^T, \quad Q_\Omega = F_\Omega^TE^TQ + QEF_\Omega \tag{7.50a}$$

with

$$F_\Omega := \frac{1}{2\pi}\int_\Omega (\jmath\nu E - A)^{-1}\mathrm{d}\nu. \tag{7.51}$$

2. The frequency-limited Gramians are given from the solutions of the *frequency-limited reachability and observability GCALEs*

$$AP_\Omega E^T + EP_\Omega A^T + B_\Omega B^T + BB_\Omega^T = 0, \quad B_\Omega := EF_\Omega B \tag{7.52a}$$
$$A^TQ_\Omega E + E^TQ_\Omega A + C_\Omega^TC + C^TC_\Omega = 0, \quad C_\Omega := CF_\Omega E. \tag{7.52b}$$

The matrix-valued integral can be represented in terms of the matrix logarithm via

$$F_\Omega = \mathrm{Re}\left(\frac{\jmath}{\pi}\ln\left((A+\jmath\omega_1E)^{-1}(A+\jmath\omega_2E)\right)\right)E^{-1} \tag{7.53a}$$
$$= E^{-1}\,\mathrm{Re}\left(\frac{\jmath}{\pi}\ln\left((A+\jmath\omega_2E)(A+\jmath\omega_1E)^{-1}\right)\right). \tag{7.53b}$$

Proof. Using the equivalent standard state-space system defined by $\hat{A} := E^{-1}A,\ \hat{B} := E^{-1}B,\ C$ leads, by employing (7.31) and (7.51), to

$$\frac{1}{2\pi}\int_{\Omega}(\jmath\nu I_n - \hat{A})^{-1}\mathrm{d}\nu = F_\Omega E$$

which, by applying [186, Theorems 3.1-3.2] and (7.30), immediately gives (7.50). The reachability CALE w.r.t. $\hat{A},\ \hat{B}$ is

$$\begin{aligned} & 0 = \hat{A}P_\Omega + P_\Omega\hat{A}^T + F_\Omega E\hat{B}\hat{B}^T + \hat{B}\hat{B}^T F_\Omega^T E^T \\ & \quad = E^{-1}AP_\Omega + P_\Omega A^T E^{-T} + F_\Omega BB^T E^{-T} + E^{-1}BB^T F_\Omega^T \\ \Leftrightarrow \quad & 0 = AP_\Omega E^T + EP_\Omega A^T + EF_\Omega BB^T + BB^T F_\Omega^T E^T \end{aligned}$$

and (7.52a) is established. The frequency-limited observability GCALE (7.52b) is derived using similar steps. For (7.53), first note that $F_\Omega = \frac{1}{2\pi}\int_{\Omega}(\jmath\nu I_n - \hat{A})^{-1}\mathrm{d}\nu E^{-1}$ from which (7.53a) easily follows by using Theorem 7.5. Alternatively, it holds $F_\Omega = E^{-1}\frac{1}{2\pi}\int_{\Omega}(\jmath\nu I_n - AE^{-1})^{-1}\mathrm{d}\nu$, which, by applying Theorem 7.5 again, leads to (7.53b). □

The algorithms we suggested in the standard state-space case for computing approximations of B_Ω, C_Ω, P_Ω, and Q_Ω are also applicable here with minor modifications. Some care must be taken when Krylov subspace methods are used for this purposes, since they implicitly work on $E^{-1}A$ or on $L^{-1}AL^{-T}$ if $0 \prec E = LL^T$. Hence, the correct formulation of F_Ω should be chosen. As alternative, the use of the generalized LR-ADI (G-LR-ADI) iteration (Algorithm 3.2, [39]) for the GCALEs is straightforward. The concept of frequency-limited Gramians for descriptor systems and certain classes of nonlinear control systems is considered in [136] and [206], respectively. However, the numerical efficient realization, e.g. in terms of low-rank approximations, of FLBT for such systems is still not investigated.

Stability Preservation and Modified Frequency-Limited Balanced Truncation

It is not guaranteed that FLBT preserves stability in the reduced order model [107]. In [116], a modification of frequency-limited BT is presented which ensure this preservation. Consider the EVDs of the inhomogeneities of (7.46) and (7.52):

$$\begin{aligned} B_\Omega B^T + BB_\Omega^T = Q_B S_B Q_B^T, \quad & S_B = \operatorname{diag}(\theta_1, \ldots, \theta_{2m}, 0, \ldots, 0) \in \mathbb{R}^{n\times n}, \\ C_\Omega^T C + C^T C_\Omega = Q_C S_C Q_C^T, \quad & S_C = \operatorname{diag}\left(\eta_1, \ldots, \eta_{2p}, 0, \ldots, 0\right) \in \mathbb{R}^{n\times n} \end{aligned}$$

with $Q_B^T Q_B = Q_C^T Q_C = I,\ Q_B,\ Q_C \in \mathbb{R}^{n\times n}$. Assuming that $\operatorname{rank}([B,\ B_\Omega]) = r_B \leq 2m$ and $\operatorname{rank}\left([C^T,\ C_\Omega^T]\right) = r_C \leq 2p$, it holds $\theta_i, \eta_j \neq 0$ for $i = 1, \ldots, r_B$, $j = 1, \ldots, r_C$. However, there can be both negative and positive values of θ_i, η_j. Let $Q_{B,1} \in \mathbb{R}^{n\times r_B},\ Q_{C,1} \in \mathbb{R}^{n\times r_C}$ be the first r_B as well as r_C columns of Q_B and Q_C, and consider the *modified*

frequency-limited GCALEs

$$\begin{aligned} AP_\Omega^{\mathrm{mod}}E^T + EP_\Omega^{\mathrm{mod}}A^T + B_\Omega^{\mathrm{mod}}B_\Omega^{\mathrm{mod}\,T} &= 0, \\ A^TQ_\Omega^{\mathrm{mod}}E + E^TQ_\Omega^{\mathrm{mod}}A + C_\Omega^{\mathrm{mod}\,T}C_\Omega^{\mathrm{mod}} &= 0 \end{aligned} \tag{7.54}$$

$$\text{with} \quad B_\Omega^{\mathrm{mod}} := Q_{B,1}\operatorname{diag}\left(|\theta_1|,\dots,|\theta_{r_B}|\right)^{\frac{1}{2}}, \quad C_\Omega^{\mathrm{mod}} := \operatorname{diag}\left(|\eta_1|,\dots,|\eta_{r_C}|\right)^{\frac{1}{2}} Q_{C,1}^T.$$

That is, the negative values in S_B and S_C are essentially simply negated. Computing the $r_B, r_C \ll n$ nonzero eigenvalues θ_i, η_j and their corresponding eigenvectors can be done very inexpensively. Performing balancing and truncation on the basis of these modified frequency-limited Gramians P_Ω^{mod}, Q_Ω^{mod} yields modified FLBT (FLBT$^{\mathrm{mod}}$). This approach ensures that, under some mild conditions [116, Theorem 11], the reduced order model is asymptotically stable and that, similar to (7.8), the error bound

$$\|H - \tilde{H}\|_{\mathcal{H}_\infty} \le 2\|J_B\|\|J_C\| \sum_{j=r+1}^{n} \sigma_j^{\mathrm{mod}}, \tag{7.55}$$

$$J_B := \operatorname{diag}\left(()\,|\eta_1^B|,\dots,|\eta_{r_B}^B|\right)^{-\frac{1}{2}} Q_{B,1}^T B, \quad J_C := CQ_{C,1}\operatorname{diag}\left(()\,|\eta_1^C|,\dots,|\eta_{r_C}^C|\right)^{-\frac{1}{2}},$$

can be established, where the σ_j^{mod} denote the modified frequency-limited singular values, i.e., the square roots of the eigenvalues of $P_\Omega^{\mathrm{mod}}Q_\Omega^{\mathrm{mod}}$. The computation of low-rank factors of P_Ω^{mod}, Q_Ω^{mod} can be carried out similarly as without this modification using Algorithm 7.3, i.e., after an accurate approximation to, e.g., B_Ω has been computed, B_Ω^{mod} is constructed as above. For reusing the generated Krylov basis to subsequently compute low-rank solution factors of P_Ω^{mod}, some small changes are necessary. Since B_Ω^{mod} is obtained by altering B, B_Ω, the inhomogeneity of the projected, modified, frequency limited GCALE cannot be built similar to (7.48). Hence, B_Ω^{mod} has to be projected explicitly via $Q_k^T B_\Omega^{\mathrm{mod}}$. The inhomogeneities of (7.54) are symmetric positive semidefinite, such that the (G-)LR-ADI iteration can be applied directly.

However, because P_Ω^{mod}, Q_Ω^{mod} do not fulfill the relations (7.30) and (7.50), one cannot expect that they also exhibit a fast eigenvalue decay similar to P_Ω, Q_Ω. Some numerical experiments show that the eigenvalues of P_Ω^{mod}, Q_Ω^{mod} decay at a similar speed as those of the infinite Gramians P, Q. This can also lead to more iteration steps required by the applied Krylov subspace method or the G-LR-ADI iteration compared to P_Ω, Q_Ω.

Time-Limited Variants

In [107], a series of related approaches is proposed which restrict BT (for the case $E = I_n$) in certain ways. One possibility is to consider a time interval $\mathbb{T}$, e.g., $\mathbb{T} = [0, t_1]$, $t_1 < \infty$. Restricting BT to $\mathbb{T}$ leads to time-limited BT and aims at finding a reduced model whose output $\tilde{y}$ is an accurate approximation of the original output y, but only within the time frame $\mathbb{T}$. This leads to time-limited Gramians $P_\mathbb{T}$, $Q_\mathbb{T}$ which are the solutions of the time-limited CALEs

$$AP_\mathbb{T} + P_\mathbb{T}A^T + BB^T - B_\mathbb{T}B_\mathbb{T}^T = 0, \quad A^TQ_\mathbb{T} + Q_\mathbb{T}A + C^TC - C_\mathbb{T}^TC_\mathbb{T} = 0,$$

where $B_{\mathbb{T}} := F_{\mathbb{T}}B$, $C_{\mathbb{T}} := CF_{\mathbb{T}}$ with $F_{\mathbb{T}} := \mathrm{e}^{At_1}$. Hence, the numerical approaches for FLBT presented before can be easily adjusted to this time limiting setting. The main difference is that one has to deal with the matrix valued exponential. It is also possible to combine frequency and time-limited BT, where products of the form $F_{\Omega}F_{\mathbb{T}}$ occur that have to be dealt with.

Restricted Balanced Truncation for Discrete-Time Systems

Both frequency- and time-limited BT can also be carried out for discrete-time systems (2.6). Let for example $\Omega_d = [-\omega_1,\ \omega_1]$, $\omega_1 < \pi$, and $\mathbb{T}_d = [0,\ i_1]$, $i_1 \in \mathbb{N}_+$ be the considered frequency and discrete-time intervals. The infinite, frequency-, and time-limited Gramians are the solutions of the DALEs

$$\begin{aligned} P - APA^T &= BB^T, & Q - A^TQA &= C^TC,\\ P_{\Omega_d} - AP_{\Omega_d}A^T &= B_{\Omega_d}B^T + BB_{\Omega_d}^T, & Q_{\Omega_d} - A^TQ_{\Omega_d}A &= C_{\Omega_d}^TC + C^TC_{\Omega_d},\\ P_{\mathbb{T}_d} - AP_{\mathbb{T}_d}A^T &= BB^T + B_{\mathbb{T}_d}B_{\mathbb{T}_d}^T, & Q_{\mathbb{T}_d} - A^TQ_{\mathbb{T}_d}A &= C^TC + C_{\mathbb{T}_d}^TC_{\mathbb{T}_d}, \end{aligned}$$

where

$$\begin{aligned} B_{\Omega_d} &:= F_{\Omega_d}B, \quad C_{\Omega_d} := CF_{\Omega_d}, \quad F_{\Omega_d} := \tfrac{1}{2\pi}\,\mathrm{Re}\,\left(\omega_1 I - 2\jmath\ln\left(I - A\mathrm{e}^{-\jmath\omega_1}\right)\right),\\ B_{\mathbb{T}_d} &:= F_{\mathbb{T}_d}B, \quad C_{\mathbb{T}_d} := CF_{\mathbb{T}_d}, \quad F_{\mathbb{T}_d} := A^{i_1}, \end{aligned}$$

see, e.g., [107, 186]. Approximating the products with the different matrix valued functions F_{Ω_d}, $F_{\mathbb{T}_d}$ can be done exactly as we described above. Subsequently, computing low-rank solution factors of the DALEs can also be done by similar methods, e.g. Algorithm 4.7, as we used before, which again enables an efficient realization of these BT variants also for large-scale systems. We plan to investigate the eigenvalue decay of the Gramians mentioned in this section, as well as specially tailored numerical algorithms for their approximation, in future work.

BT at a Single Frequency

Another conceptually very different modification is presented in [86], where BT is restricted to a single frequency $\omega_1 \in \mathbb{R}_+$. There, the CALEs to be solved are

$$A_{\omega_1}P_{\omega_1} + P_{\omega_1}A_{\omega_1}^H + B_{\omega_1}B_{\omega_1}^T = 0, \quad A_{\omega_1}^HQ_{\omega_1} + Q_{\omega_1}A_{\omega_1} + C_{\omega_1}^TC_{\omega_1} = 0,$$

where $B_{\omega_1} := F_{\omega_1}B$, $C_{\omega_1} := CF_{\omega_1}$ with $F_{\omega_1} := \epsilon(\epsilon I_n + \jmath\omega_1 I_n - A)^{-1}$ and $A_{\omega_1} := \jmath\omega_1 I_n + \epsilon(\epsilon I_n + \jmath\omega_1 I_n - A)^{-1}(\jmath\omega_1 I - A)$, $\epsilon > 0$. Here, B_{ω_1}, C_{ω_1} are easily obtained by solving linear systems of equations, but since the coefficients of the occurring CALEs are now also matrix-valued functions of A, the application of low-rank solvers is not as straightforward as for the cases considered before. Also note that the CALEs are defined by complex data such that P_{ω_1}, Q_{ω_1} will be complex. An extension of this approach to frequency intervals is under current research. First experiments in [86] raise the expectation that the eigenvalues of P_{ω_1}, Q_{ω_1} also decay faster than those of P, Q.

7.3.5. Numerical Examples

Here, we numerically evaluate the results of Section 7.3.2 regarding the eigenvalue decay of the frequency-limited Gramians, the numerical approaches presented in Section 7.3.3 to approximate the product with the matrix-valued function as well as the low-rank approximations of the GCALE solutions, and the accuracy of the reduced order models obtained by the considered BT variants employing these low-rank approximations.

As test cases we use the *FDM* example with $f_1 = 10^2\xi_1$, $f_2 = 10^3\xi_2$ with $n_0 = 350$ grid points, as well as the *rail79k* and *ifiss66k* examples (cf. Sections 2.4, 5.3.3).

Eigenvalue Decay and Numerical Rank of the Gramians

For illustrating the eigenvalue decay, we use a small version of the *FDM* example with $n_0 = 30$ grid points such that $n = n_0^2 = 900$ for A. The input matrix $B \in \mathbb{R}^n$ is chosen as vector with random entries from a normal distribution. This small system admits an exact numerical computation of the controllability Gramians P, P_Ω, and P_Ω^{mod} (cf. (7.54)) by using the `lyap` routine of MATLAB.

At first, the frequency interval boundaries are $\omega_1 = 10^3$, $\omega_2 = 10^4$ and the matrix valued function F_Ω can be computed by the `logm` routine. The eigenvalues of all three Gramians P, P_Ω, and P_Ω^{mod}, scaled by their respective largest entry, are plotted in the left plot of Figure 7.3. Apparently, the eigenvalues of P_Ω decay significantly faster than the ones of P and P_Ω^{mod}, which decay nearly identically. The numerical rank of the infinite Gramian w.r.t. the machine precision obtained by the MATLAB command `rank` is $\operatorname{rank}(P, \mathbf{u}_{\text{mach}}) = 72$.

In Table 7.2 we list the numerical ranks of the frequency-limited Gramian P_Ω, of the modification P_Ω^{mod}, and the spectral radius of the matrix Γ. For the used frequency interval, the numerical rank of P_Ω is, as predicted by Figure 7.3, noticeable smaller then the numerical rank of P. We also tested other frequency intervals, where we used the quantity $|\lambda_q|$ from Lemma 7.8 , i.e., the magnitude of the eigenvalue where $\frac{|\operatorname{Im}(\lambda)|}{|\operatorname{Re}(\lambda)|}$ is maximal. Notice that in the right plot of Figure 7.3, the Bode magnitude plot of the transfer function matrix H shows a distinct bulk near $|\lambda_q|$. According to the discussion in the end of Section 7.3.2, setting the interval boundaries equal to $|\lambda_q|$, can increase the spectral radius $\rho(\Gamma)$ and thus slow down the eigenvalue decay of P_Ω. The results in the last row confirm this for the choice $\omega_2 = |\lambda_q|$ which yields $\rho(\Gamma) > \frac{1}{2}$ and very close numerical ranks of P_Ω and P. In conclusion, the results in Table 7.2 seem to confirm the expected influence of $\rho(\Gamma)$ on the numerical rank of P_Ω. For the last frequency interval $[0,\ 10^5]$, one can also see that $\rho(\Gamma)$ approaches $\frac{1}{2}$ for increasingly large frequency intervals. The numerical rank of the modified frequency-limited Gramian P_Ω^{mod} appears to be largely unaffected by different frequency intervals and is always very close to the numerical rank of P.

Influence of the Different Inhomogeneities to the Low-Rank Solvers

Before we test the proposed Algorithm 7.3 and carry out frequency-limited balanced truncation using the computed low-rank GCALE solutions, we consider how the different

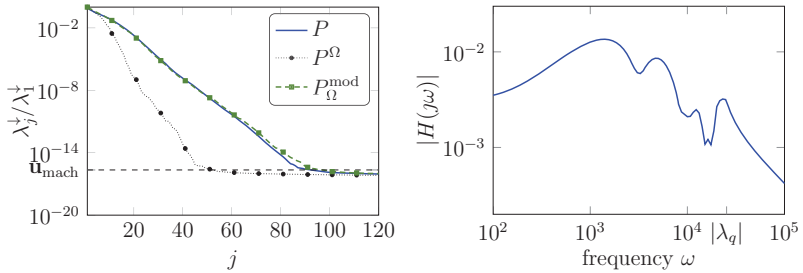


Figure 7.3.: Left: Scaled eigenvalues of the Gramians. Right: Transfer function plot of the system.

Table 7.2.: Computed numerical rank of the different Gramians w.r.t. various frequency interval boundaries. Here, $\operatorname{rank}(P, \mathbf{u}_{\text{mach}}) = 72$ and $|\lambda_q| = 2.5337 \cdot 10^4$.

$\omega_1,\ \omega_2$	$\rho(\Gamma)$	$\operatorname{rank}(P_\Omega, \mathbf{u}_{\text{mach}})$	$\operatorname{rank}(P_\Omega^{\text{mod}}, \mathbf{u}_{\text{mach}})$
10^3, 10^4	0.43	39	74
10^2, 10^3	0.21	10	69
10^3, $\lvert\lambda_q\rvert$	0.57	67	74
0, 10^5	0.49	74	74

low-rank solvers perform when applied to the standard GCALE (7.7), the frequency-limited GCALEs (7.30) and (7.50) as well as the modified frequency-limited GCALE (7.54). In other words, we investigate how the performance of the low-rank solvers differs w.r.t. to the different inhomogeneities, as these are the only differences in all three GCALEs. For this we restrict the investigation to the controllability Gramians P, P_Ω and P_Ω^{mod}. Since the direct calculation of the matrix valued logarithm via the `logm` command is too much memory and time consuming, we use approximations of B_Ω obtained by numerical quadrature employing the `integral` command. The necessary eigenvalues and -vectors to construct B_Ω^{mod} for (7.54) (cf. Section 7.3.4) are computed using the `eigs` routine, which took less then one second in all cases. We employ EKSM [209], RKSM with the convex hull based shifts (RKSM($\mathcal{D}$)) [83] as well as the G-LR-ADI iteration [183, 36, 39] (and its LDL^T-variant [153] for P_Ω) for computing the low-rank solution factors. We point out that computing low-rank solution factors of P_Ω and P_Ω^{mod} in this way only serves a comparative purpose. As shown later in Section 7.3.5, using the proposed approach in Algorithm 7.3, is clearly more practical and efficient as it also provides approximations to B_Ω at once. The shifts for the G-LR-ADI iteration are generated adaptively using the approximate residual norm-minimizing shifts proposed in Section 5.3. All methods are terminated as before when the scaled GCALE residual norm (cf. Line 10 in Algorithm 7.3) drops below $\tau_P = 10^{-8}$. Afterwards, a rank truncation (cf. Lines 11-12 in Algorithm 7.3) is invoked, where all eigenvalues of the low-rank solution

Table 7.3.: Numerical results regarding the approximation of the different Gramians by different methods.

Example	method	P			P_Ω			P_Ω^{mod}		
		d	g	t_c	d	g	t_c	d	g	t_c
ifiss66k $\omega_1 = 1,\ \omega_2 = 10^2$	EKSM	850	334	237.3	880	233	157.3	1060	344	242.1
	RKSM	300	290	95.4	490	242	118.5	550	359	141.9
	LR-ADI	335	322	63.8	680	247	95.8	720	350	103.4
rail79k $\omega_1 = 10^{-2},\ \omega_2 = 10$	EKSM	812	202	168.2	1260	190	325.1	1092	217	232.4
	RKSM	245	202	76.4	252	172	59.5	336	208	84.9
	LR-ADI	315	225	60.2	588	189	88.9	574	218	93.9
FDM $\omega_1 = 10,\ \omega_2 = 10^3$	EKSM	800	345	246.9	60	38	9.5	no conv.		
	RKSM	305	305	144.9	90	38	26.7	860	294	450.2
	LR-ADI	340	318	108.1	800	42	179.6	650	299	147.6

with $\lambda/\lambda_{\max} \leq 10^{-12}$ are neglected. The required subspace dimensions $d = \dim(\mathcal{Q})$, the ranks g of the obtained low-rank solutions after this truncation, as well as the computations times t_c for all methods and test systems are summarized in Table 7.3. Due to the applied rank truncation, g also coincides with the column dimension of the low-rank solution factors. For the LR-ADI methods, d is the column dimension of the computed low-rank solution factor.

The different inhomogeneities of the frequency-limited GCALEs (7.30) and (7.50) clearly affect the performance of EKSM and RKSM. As expected in Section 7.3.2, the ranks g of the low-rank approximations of P_Ω are smaller compared to the approximations of P. However, the required subspace dimensions d for the low-rank approximations of P_Ω are only smaller for the example *FDM*, where the computation time is also smaller compared to P. For the *ifiss66k* and *rail79k* examples, these differences are less pronounced since the subspace dimensions $d = \dim(\mathcal{Q})$ and computation times t_c are higher for P_Ω. However, recalling that the inhomogeneities of (7.30) and (7.50) are of rank $2m$, one can see that the number of required iteration steps is actually less compared to P. For the *rail79k* system, the ranks of the approximations for P and P_Ω show a less pronounced difference which also explains the higher subspace dimensions. The LR-ADI iteration seems to be somewhat less affected by the different inhomogeneities since the computation times for P_Ω are in the majority of cases larger than for P. It achieves, however, the smallest computation times for the infinite Gramian P for the examples *FDM*, and *ifiss66k*. In most cases, computing the low-rank solution factors of the modified frequency-limited Gramians P_Ω^{mod} seems to be more demanding for all methods compared to P, P_Ω. Apparently, the ranks of the computed low-rank approximations for P_Ω^{mod} are very close to those for P, which is one explanation for the higher computational effort.

Table 7.4.: Results for approximating B_Ω, P_Ω by Algorithm 7.3 using extended and rational Krylov subspaces with different (adaptive) shifts.

Example	method	d	d_{B_Ω}	d_{P_Ω}	$\mathcal{R}_{\text{final}}$	$\mathcal{L}_{\text{final}}$	t_{c}
ifiss66k,	RKSM($\mathcal{D}$)	415	415	0	$1.73{\cdot}10^{-8}$	$1.25{\cdot}10^{-8}$	182.6
$t_{\text{quad}} = 788$	RKSM($\jmath\Omega$)	295	295	0	$5.92{\cdot}10^{-10}$	$3.85{\cdot}10^{-10}$	98.6
	EKSM	868	868	0	$8.06{\cdot}10^{-9}$	$7.79{\cdot}10^{-9}$	454.3
rail79k,	RKSM($\mathcal{D}$)	336	336	0	$7.25{\cdot}10^{-9}$	$1.51{\cdot}10^{-9}$	103.0
$t_{\text{quad}} = 556$	RKSM($\jmath\Omega$)	273	273	0	$1.93{\cdot}10^{-8}$	$1.66{\cdot}10^{-8}$	73.6
	RKSM($\jmath[\omega_1, \omega_2]$)	322	259	63	$1.19{\cdot}10^{-8}$	$1.55{\cdot}10^{-8}$	90.2
	EKSM	130	130	0	$2.44{\cdot}10^{-10}$	$2.26{\cdot}10^{-10}$	23.2
FDM,	RKSM($\mathcal{D}$)	320	320	0	$1.18{\cdot}10^{-8}$	$1.08{\cdot}10^{-8}$	169.2
$t_{\text{quad}} = 374$	RKSM($\jmath\Omega$)	65	65	0	$9.63{\cdot}10^{-12}$	$8.83{\cdot}10^{-12}$	23.6
	RKSM($\jmath[\omega_1, \omega_2]$)	65	65	0	$1.23{\cdot}10^{-11}$	$1.13{\cdot}10^{-11}$	23.6

Numerical Approximations for $f(A)b$ and the Gramians by Krylov Subspace Methods

Here, we evaluate the numerical strategies presented in Section 7.3.3 for obtaining approximations of B_Ω and P_Ω. We employ the projection approach given in Algorithm 7.3 for different choices of Krylov subspaces. The results are summarized in Table 7.4. For RKSM, the abbreviations ($\mathcal{D}$), ($\jmath\Omega$), ($\jmath[\omega_1, \omega_2]$) refer to the use of the adaptive shifts based on the convex hull, the imaginary interval $\jmath\Omega$, and the alternating usage of $\jmath\omega_1$, $\jmath\omega_2$ and their complex conjugates. The approximation of B_Ω is regarded as accurate enough when the basis criterion (7.45) is satisfied with $\tau_f = 10^{-8}$. After reaching this condition, Algorithm 7.3 continues with computing the low-rank solution factor of P_Ω until the criterion based on the GCALE residual norm in Line 10 with $\tau_P = 10^{-8}$ is satisfied. In case of RKSM($\jmath\Omega$) and RKSM($\jmath[\omega_1, \omega_2]$), these iteration steps devoted to the GCALE solution were carried out using the convex hull based adaptive shifts, i.e., RKSM($\mathcal{D}$). We also list the dimension $d = d_{B_\Omega} + d_{P_\Omega} = \dim(\mathcal{Q})$ of the generated subspaces, where d_{B_Ω} denotes the subspace dimension necessary to approximate B_Ω to the desired accuracy and d_{P_Ω} is the number of (additional) basis vectors required to achieve also the desired accuracy w.r.t. the GCALEs. The quadrature approximation of B_Ω from the previous examples serves as reference solution B_Ω^{ref}. The final relative error

$$\mathcal{R}_{\text{final}} := \|B_\Omega^{\text{ref}} - \tilde{B}_\Omega\| / \|\tilde{B}_\Omega\|,$$

the final normalized GCALE residual norm $\mathcal{L}_{\text{final}}$ obtained after a rank truncation, as well as the consumed computing times t_{quad} and t_{c} in seconds for the quadrature approximation B_Ω^{ref} and the Krylov method (Algorithm 7.3), respectively, are also given.

Apparently, the projection approaches need less time than the quadrature approximation. Also, once the approximation of B_Ω is found, in most situations there are no additional basis vectors necessary for generating the low-rank solution factor of P_Ω. Using RKSM with adaptively computed shifts based on $\jmath\Omega$ appears to be the best choice

regarding the required subspace dimensions (which directly reflect the required iteration steps) as well as the consumed computation time. RKSM($\mathcal{D}$) leads to higher subspace dimensions and longer computing times for all test systems. The simplification RKSM($\jmath[\omega_1, \omega_2]$) of RKSM($\jmath\Omega$) as well as EKSM are competitive candidates only for the *FDM* example. They required larger subspaces for the *rail79k* example and failed to compute accurate approximations for the *ifiss66k* example. Notice that EKSM did not manage to compute a low-rank solution factor of P_Ω (cf. Table 7.3 in the previous Section) for the *ifiss66k* system. To conclude, using the adaptive, purely imaginary shifts from $\jmath\Omega$, leads to a fast convergence of RKSM for approximating B_Ω and sufficiently accurate low-rank solutions of the frequency-limited GCALEs are obtained immediately from the generated rational Krylov basis.

Reduction Results

Now we carry out the standard BT, frequency-limited BT, and its stability preserving modification [117] on the basis of low-rank solutions of the respective GCALEs. For the infinite Gramians P, Q, we use the low-rank solution method that achieved the smallest time in the experiment in Section 7.3.5 (cf. Table 7.3). The low-rank factors of P_Ω are selected from the fastest method from Table 7.4 in Section 7.3.5. The observability Gramians as well as the modified frequency-limited Gramians P_Ω^{mod}, Q_Ω^{mod} are similarly dealt with by Algorithm 7.3 with the same settings of τ_f, τ_P. The obtained low-rank solution factors are used within Algorithm 7.1 to carry out the three balanced truncation variants to generate reduced order models of a prescribed order r. It is noteworthy that no significantly differing reduced order models were constructed when another method is employed to compute the low-rank solution factors.

For each test system, the Bode magnitude plots of the original and reduced transfer function H and $\tilde{H}$, respectively, as well as the relative errors $\mathcal{E}^{\text{rel}}(\omega)$ (cf. (7.26)) are shown in Figure 7.4, where the thick red lines indicate the frequency interval boundaries ω_1, ω_2. We also indicate the theoretical error bounds (7.8) and (7.55) of BT and FLBT$^{\text{mod}}$, respectively, for $\omega \in \mathbb{R}$. For the *ifiss* example, (7.8) and (7.55) overestimate the true error by several order of magnitude (cf. Table 7.5) and are, thus, not shown in the respective error plot in Figure 7.4. While the reduced systems from BT match H in the entire frequency range, those obtained with the frequency-limited variants FLBT, FLBT$^{\text{mod}}$ show slight deviations outside the considered frequency interval $[\omega_1,\ \omega_2]$. The relative error plots clarify this as they reveal smaller errors obtained by the frequency-limited approaches within $[\omega_1,\ \omega_2]$, where especially unmodified FLBT yields superior approximations. With the exception of example *rail79k*, it also achieves a somewhat higher accuracy for $\omega \leq \omega_1$.

To quantitatively measure the approximation quality, we consider the largest relative error within the relevant frequency region Ω via

$$\mathcal{E}^{\text{rel}}_{\max}(\Omega) := \max_{\omega \in [\omega_1, \omega_2]} \mathcal{E}^{\text{rel}}(\omega).$$

The results are given in Table 7.5 which also includes the overall time t_{MOR} which sums up the computation time to acquire the low-rank Gramian factors and the generation

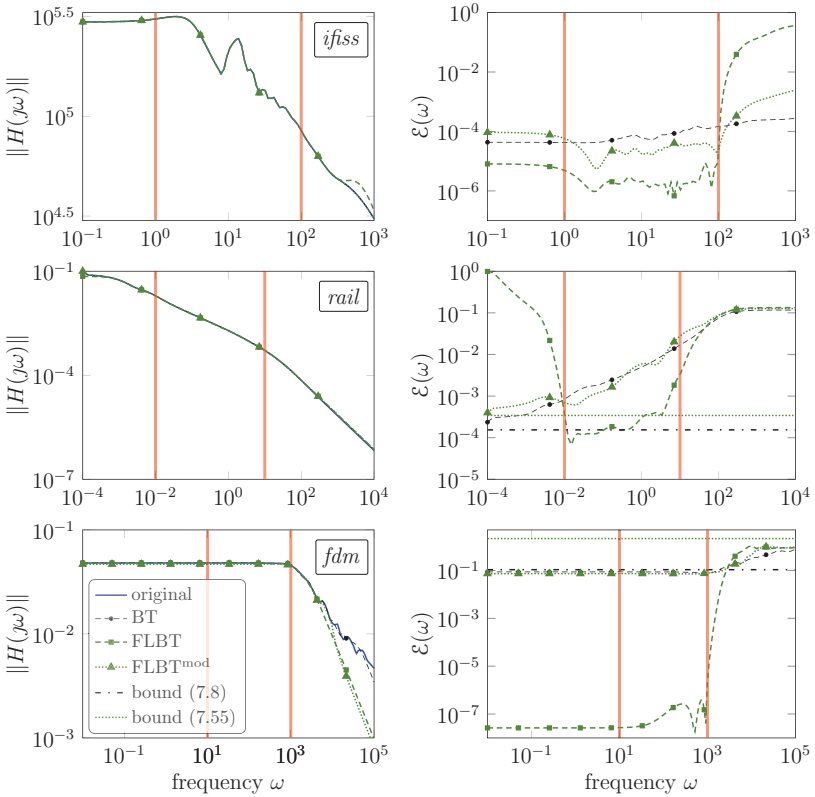


Figure 7.4.: Results obtained by different BT versions: Bode magnitude plot of original and reduced transfer functions (left), relative errors and theoretical error bounds (right). The vertical red bars indicate the frequency limits ω_1, ω_2.

Table 7.5.: Reduction result obtained by different BT versions.

Example, settings	BT type	algorithms	t_{MOR}	$\mathcal{E}_{\max}^{\mathrm{rel}}(\Omega)$	bound	stable?
ifiss66k	BT	LR-ADI	130.2	$1.48{\cdot}10^{-4}$	$1.89{\cdot}10^{2}$	1
$[\omega_1,\ \omega_2] = [1,\ 10^2]$,	FLBT	RKSM($\jmath\Omega$)	201.0	$8.82{\cdot}10^{-6}$	–	1
$r = 160$	$\mathrm{FLBT}^{\mathrm{mod}}$	RKSM($\jmath\Omega$)	320.9	$5.68{\cdot}10^{-5}$	$1.21{\cdot}10^{3}$	1
rail79k	BT	LR-ADI	105.4	$1.53{\cdot}10^{-2}$	$1.54{\cdot}10^{-4}$	1
$[\omega_1,\ \omega_2] = [10^{-2},\ 10]$,	FLBT	RKSM($\jmath\Omega$)	144.8	$2.59{\cdot}10^{-3}$	–	1
$r = 50$	$\mathrm{FLBT}^{\mathrm{mod}}$	RKSM($\jmath\Omega$)	187.9	$2.41{\cdot}10^{-2}$	$3.42{\cdot}10^{-4}$	1
FDM	BT	LR-ADI	221.1	$8.76{\cdot}10^{-2}$	$1.09{\cdot}10^{-1}$	1
$[\omega_1,\ \omega_2] = [10,\ 10^3]$,	FLBT	EKSM	54.9	$1.03{\cdot}10^{-7}$	–	0
$r = 30$	$\mathrm{FLBT}^{\mathrm{mod}}$	RKSM($\jmath\Omega$)	330.6	$7.61{\cdot}10^{-2}$	2.17	1

of the reduced order model by Algorithms 7.3 and 7.1, respectively. We also indicate if the constructed reduced system is asymptotically stable.

It can be clearly confirmed that FLBT provides reduced systems with the best approximation quality in $[\omega_1, \omega_2]$ for all test systems. Hence, the goal, mentioned in the beginning, to achieve better accuracies at the same reduced order is fulfilled. For comparison, BT achieves a comparable accuracy, e.g., for system *ifiss66k*, if the reduced order is increased to $r \approx 200$. Regarding the computation times t_{MOR}, FLBT is in some cases more expensive due to the required handling of the matrix function. If an efficient numerical low-rank approach is used, computing the required low-rank solution factors of the frequency-limited Gramians can, however, be cheaper compared to the infinite Gramians. This is especially the case when the numerical ranks of the frequency-limited Gramians are noticeable smaller than those of the infinite ones, e.g., for the *FDM* example. There, carrying out FLBT requires less time than BT. For the *FDM* example, however, FLBT returned an asymptotically unstable reduced system. This was also observed for the other systems for some smaller reduced dimensions r. The stability preserving modification $\mathrm{FLBT}^{\mathrm{mod}}$ does always provide a stable reduced order models but the computation times are significantly higher than for BT and FLBT. Similar to the experiments in Section 7.3.5, it seems to be harder to solve (7.54) for low-rank solutions. Moreover, this stability preservation appears to sacrifice the obtained accuracy in $[\omega_1, \omega_2]$. In particular, for the *rail79k* and *FDM* systems, the accuracies of $\mathrm{FLBT}^{\mathrm{mod}}$ and ordinary, unrestricted BT are very similar which renders the introduction of the frequency restrictions essentially redundant. If stability preservation is really crucial and FLBT fails in this direction, but if also the efficient numerical realization of the model order reduction approach is demanded, we suggest to use BT without frequency limitations.

Similar to the results in Section 7.2.5, in some cases a violation of at least one of the theoretical error bounds (7.8), (7.55) associated to BT, $\mathrm{FLBT}^{\mathrm{mod}}$ can be seen in Figure 7.4 and Table 7.5. In particular, this occurs for the *rail* and *fdm* example. This is again caused by the errors induced by the approximate GCALE solutions.

7.4. Conclusion and Outlook

We considered the execution of BT by means of approximate low-rank solutions of the Gramians which are computed by the G-LR-ADI iteration a dual version thereof. This admits to monitor the approximate Hankel singular values as the iteration proceeds and use these as goal-oriented stopping criterion instead of the GCALE residual norms. Numerical experiments show that this might lead to a earlier termination of the dual G-LR-ADI iteration which can save some numerical effort but the produced reduced order models can potentially be less accurate or even unstable. A structure preserving variant of BT for second order systems was also discussed.

As second main topic of this chapter, BT restricted to limited frequency intervals (FLBT) was investigated. Compared to standard BT in the first two sections, in FLBT a matrix-valued function occurs in the inhomogeneities of the Gramians which have to be computed. The effects of this matrix-valued function on the eigenvalues of the Gramians within FLBT have been investigated. The established eigenvalue bounds indicate faster decaying eigenvalues of the frequency-limited Gramians compared to the infinite Gramians which can result in smaller numerical ranks. Due to some of the employed eigenvalue inequalities, the found bounds are not very tight such that further research in that direction will yield further insights. A single rational Krylov subspace method with suitable shift parameters can be employed to intelligently deal with the matrix-valued functions, as well as the computation of low-rank solution factors of the occurring GCALEs. This suggested approach was, in the numerical tests, superior to other approaches, e.g., the extended Krylov subspace method or the G-LR-ADI iteration. Further improvements of RKSM [84, 166] might yield additional performance gains and are subject to future investigations. Carrying out FLBT using this method led to reduced order models of better accuracy compared to BT. In some cases, the numerical effort of FLBT was even smaller because of the lower effort to approximate the frequency-limited Gramians. The stability preserving modified FLBT approach [116] was also considered, but, although similar techniques can be employed, the occurring GCALEs appear to be significantly harder to deal with, independent of the used low-rank method. Improving the handling of these modified Gramians, as well as related approaches with restrictions on time or for discrete-time systems, are also further research perspectives.

CHAPTER 8

CONCLUSIONS AND OUTLOOK

Contents

8.1. Summary

This thesis investigated the numerical feasible computation of low-rank solution factors for large-scale algebraic matrix equations. It is motivated by the often observed as well as theoretically predicted fast decay of the singular values of the solutions [184, 4, 113, 222, 20, 23, 7]. The main focus of this thesis lay on approaches based on the low-rank ADI iteration for Lyapunov and Sylvester equations [183, 161, 43]. The first contribution in Chapter 3 was, that the residual during these iteration has a small, fixed rank and can be given explicitly in a low-rank factorization. The incorporated low-rank factors of the residual can be plugged into the method as intrinsic part which led to reformulated versions of the low-rank ADI iteration. Moreover, this also enabled to compute the residual norm very cheaply compared to older approaches.

Methods based on the ADI iteration require a number of shift parameters to achieve a fast convergence. In various applications, these parameters can be complex and we proposed in Chapter 4 a novel efficient approach to deal with these complex shifts and still generate real low-rank solution factors in the end.

The efficient numerical generation of high quality shift parameters was investigated in Chapter 5 which led to novel strategies, where the shifts are generated during the low-rank ADI iteration. This is done completely automatic during the course of the iteration such that the low-rank ADI methods essentially become independent of a pre-computed shift parameters and, to a large extent, also to setup data required for the shift generation. One key concept of these shifts is based on successively minimizing the residual norm which is considered as function dependent on the shift parameter as variable. These new shift generation strategies lead to a faster convergence of the ADI iteration compared to traditional approaches in the majority of cases and also made the

low-rank ADI iteration competitive with other low-rank methods, e.g., rational Krylov subspace methods [83, 82].

As application, the usage of the low-rank ADI iteration for Lyapunov and Sylvester equations within Newton methods for large-scale, symmetric [42, 201, 94] and, respectively, nonsymmetric algebraic Riccati equations [241] was considered in Chapter 6. The proposed efficiency enhancements also led to performance improvements and reduced computation times in this situation. The outer Newton iteration of these algorithms can be further drastically accelerated by performing a Galerkin projection using the space spanned by the current low-rank solution factors. By incorporating all these techniques, the low-rank Newton-ADI method performed well compared to other, existing algorithms. However, the rational Krylov subspace method [211] was superior in some experiments w.r.t. symmetric Riccati equations.

Balanced truncation model order reduction [175] was selected in Chapter 7 as second application of the investigated low-rank matrix equation solvers. Here, a pair of adjoint Lyapunov equations has to be solved by means of low-rank solution factors for which a dual low-rank ADI iteration was used. This admitted to monitor quantities relevant for the model order reduction application during the iteration and formulate goal-oriented stopping criteria. Using the low-rank solution factors generated by this dual low-rank ADI iteration led to accurate reduced order models. This accuracy was noticeably decreased when the iteration was terminated earlier because of the goal-oriented stopping and in some cases, unstable reduced order models were constructed. Balanced truncation can be restricted to finite frequency intervals which leads to frequency-limited balanced truncation [107]. There, the inhomogeneities of the occurring Lyapunov equations involve a complicated, matrix valued function which has to be dealt with in addition. The eigenvalue decay of the solutions of these Lyapunov equations was investigated in more detail and a different behavior compared to the Lyapunov equations in unrestricted balanced truncation was revealed. It is possible to tackle both the matrix valued functions as well as the Lyapunov equations by a single rational Krylov subspace algorithm with appropriate shift parameters and utilizing a subspace recycling idea. It is due to the presence of the matrix valued function that the low-rank ADI iteration is not competitive in this particular situation. The generated reduced order models were of smaller dimension and of better accuracy in the considered frequency interval in comparison to unrestricted balanced truncation. In some cases it was even possible to generate the required low-rank Lyapunov solution factors with smaller numerical effort.

8.2. Future Research Perspectives

With respect to the theoretical prediction of the singular value decay of solutions of matrix equations, there is still not much known about the influence of the inhomogeneities and the eigenvectors of the coefficient matrices. Some first investigations in this direction can found, e.g., in [160, 222, 7]. Non-normality of the coefficient matrices seems to lead to counter-intuitive effects as reported in [7]. For algebraic Riccati equations even less is known w.r.t. these topics. Some results w.r.t. Lyapunov equations can be generalized to

symmetric Riccati equations [23]. A better understanding of these issues could help to further improve the existing or develop new low-rank algorithms for matrix equations.

The developed low-rank factorization of the residual within low-rank ADI methods does only hold when the occurring linear systems are solved exactly. Some investigations regarding inexact solutions, e.g., by iterative Krylov methods for large linear systems can be found in [160, 202, 219], but there the original low-rank ADI methods were used without the explicitly incorporated low-rank residual factors. Further research w.r.t. those inexact linear solves is necessary, especially when sparse direct solvers cannot be applied in a computationally feasible way.

The developed novel automatic shift approaches in Chapter 5 led to a significant performance improvement of the low-rank ADI methods. However, further enhancements can be achieved by applying more sophisticated and specially tailored optimization routines, e.g., approaches designed intrinsically for the occurring eigenvalue minimization problems, see, e.g., [173, 174, 180]. Especially in the case of Sylvester equations, the optimization problems appear to be demanding for the optimization routines. Of course, knowledge or further theoretical investigations of the analytic solution of the occurring optimization problem might be invaluable to further enhance the residual norm-minimizing shifts.

All the just mentioned points naturally also apply to the considered Newton-ADI hybrids in Chapter 6 for large-scale algebraic Riccati equations. Further research topics include the use of inexact solves of the inner, linear matrix equations and using line-search strategies [25, 94, 135]. Both ideas are subject to current research in [125], where it is shown that they potentially yield additional, noticeable performance gains and even enable convergence when the standard Newton-ADI method fails.

Regarding the low-rank balanced truncation framework an open problem is how the inexact solutions of the Lyapunov equations influence the reduction process. The experiments in Chapter 7 reveal that a premature termination of the low-rank method can yield noticeable less accurate and unstable reduced order models. Hence, further research is needed in this direction, e.g., by determining bounds for a minimal required accuracy of the Gramian approximations in order to preserve stability. This, in turn, might provide more reliable goal-oriented termination criteria.

The bounds regarding the eigenvalue decay of the Lyapunov solutions occurring in frequency-limited balanced truncation are descriptive but somewhat conservative and not very tight. Obtaining better bounds might yield further insights how the frequency restrictions influence the eigenvalue decay. This might then be used to improve the methods for dealing with the involved matrix valued function and computing the low-rank Gramian factors. Computing low-rank solution factors of the stability preserving modification [116] of frequency-limited balanced truncation was computationally more demanding compared to original, frequency-limited as well as unrestricted balanced truncation. The accuracy of the obtained reduced order models appear to also suffer from this modification. Hence, further research effort might be devoted to alternative ways to maintain stability in frequency-limited balanced truncation.

APPENDIX A

ADDITIONAL ALGORITHMS

In Algorithm A.1, a basic implementation of the rational Krylov subspace method (RKSM) [82] for GCALEs is illustrated. There, in Lines 1,6,8, orth refers to any stable (block) orthogonalization routine. The matrix $\tilde{A}_j$ in Line 9 as well as $\|\mathcal{L}(Q_j\tilde{X}_jQ_j^T)\|$ in Line 11 can be computed more efficiently [83]. Moreover, E can also be incorporated using its Cholesky factors [209, 83] if $E = E^T \succ 0$. By adjusting Line 3, Algorithm A.1 can be rewritten into a standard [196, 137] or extended Krylov subspace method (EKSM) [209].

Algorithm A.1: Rational Krylov subspace method (RKSM) for GCALEs

Input : $A,\ E,\ F$ as in (3.12), shifts $\{s_1, \ldots, s_{j_{\max}}\} \subset \mathbb{C}$, tolerance $0 < \tau \ll 1$.
Output: $Z \in \mathbb{R}^{n\times k}$ such that $ZZ^T \approx X$ with $k \ll n$.

1 $F_E := E^{-1}F$, $Q_0 = \operatorname{orth}(F_E)$.
2 **for** $j = 1, 2, \ldots, j_{\max}$ **do**
3 Generate new basis vectors $q_j = (A - s_jE)^{-1}Eq_{j-1}$.
4 Orthogonally extend basis matrix Q_{j-1}:
5 **if** $s_j \in \mathbb{R}$ **then**
6 $Q_j = \operatorname{orth}([Q_{j-1},\ q_j])$.
7 **else**
8 $Q_j = \operatorname{orth}([Q_{j-1},\ \operatorname{Re}(q_j), \operatorname{Im}(q_j)])$, $j = j + 1$.
9 $\tilde{A}_j = Q_j^TE^{-1}AQ_j$, $\tilde{F}_j = Q_j^TF_E$.
10 Solve $\tilde{A}_j\tilde{X}_j + \tilde{X}_j\tilde{A}_j^T + \tilde{F}_j\tilde{F}_j^H = 0$ for $\tilde{X}_j$.
11 **if** $\|\mathcal{L}(Q_j\tilde{X}_jQ_j^T)\| < \tau\|FF^T\|$ **then**
12 Compute (and truncate) EVD $\tilde{X}_j = W_j\tilde{\Lambda}_jW_j^T$, $W_j^TW_j = I_{rj}$, $\tilde{\Lambda}_j = \operatorname{diag}\left(\tilde{\lambda}_1, \ldots, \tilde{\lambda}_{rj}\right)$.
13 Construct low-rank solution factor $Z = Q_jW_j\tilde{\Lambda}_j^{\frac{1}{2}}$.
14 Stop rational Krylov process.

Algorithm A.2: LR-NADI-N-r for generating real low-rank solutions of NAREs

Input : Matrices $A,\ B,\ F,\ G,\ U,\ P$ defining (6.17), initial guesses $K^{(0)},\ H^{(0)}$, and stopping tolerances $\tau_{\text{NM}}, \tau_{\text{ADI}} \ll 1$.

Output: $Z^{(k)} \in \mathbb{R}^{n\times t_j}$, $Y^{(k)} \in \mathbb{R}^{m\times t_j}$, $\Gamma^{(k)} \in \mathbb{R}^{t_j\times t_j}$ s.t. $Z^{(k)}\Gamma^{(k)}(Y^{(k)})^T \approx X$.

$k = 1$.
while $\|\mathcal{R}(X^{(k-1)})\| > \tau_{NM}\|UP^T\|$ **do**
  Get shifts $\{\alpha_i\}$, $\{\beta_i\}$ w.r.t. $A^{(k)} := A - K^{(k-1)}G^T$, $B^{(k)} := B - F(H^{(k-1)})^T$.
  $W_0 = [U, K^{(k-1)}]$, $T_0 = [P, H^{(k-1)}]$, $Z_0^{(k)} = \Gamma_0^{(k)} = Y_0^{(k)} = [\,]$, $j = 0$
  while $\|W_j(T_j)^H\| > \tau_{ADI}\|W_0(T_0)^H\|$ **do**
    $j = j + 1$, $\gamma_j = \beta_j + \alpha_j$.
    $V_j = (A^{(k)} + \beta_j I_n)^{-1}W_{j-1}$, $S_j = (B^{(k)} + \alpha_j I_m)^{-H}T_{j-1}$.
    if *Case 1:* $\beta_j \in \mathbb{R} \ \wedge \alpha_j \in \mathbb{R}$ **then**
      $W_j = W_{j-1} - \gamma_j V_j$, $T_j = T_{j-1} - \gamma_j S_j$.
      $Z_+ = V_j,\ Y_+ = S_j,\ \Gamma_+ = -\gamma_j I_{r+p}$, $h = 1$.
    if *Case 2:* $\beta_j \in \mathbb{C} \ \wedge \alpha_j \in \mathbb{C}$ **then**
      $W_{j+1} = W_{j-1} - 2\,\mathrm{Re}(\gamma_j)\,\mathrm{Re}(V_j) - \left(\frac{|\gamma_j|^2}{\mathrm{Im}(\beta_j)} - 2\,\mathrm{Im}(\gamma_j)\right)\mathrm{Im}(V_j)$.
      $T_{j+1} = T_{j-1} - 2\,\mathrm{Re}(\gamma_j)\,\mathrm{Re}(S_j) + \left(\frac{|\gamma_j|^2}{\mathrm{Im}(\alpha_j)} - 2\,\mathrm{Im}(\gamma_j)\right)\mathrm{Im}(S_j)$.
      $Z_+ = \left[\mathrm{Re}(V_j),\ \mathrm{Im}(V_j)\right],\ Y_+ = \left[\mathrm{Re}(S_j),\ \mathrm{Im}(S_j)\right]$.
      $\Gamma_+ = \hat{\Gamma}_j^{(2)}$ from (4.16), $j = j + 1$, $h = 2$.
    if *Case 3a:* $\beta_j \in \mathbb{C} \ \wedge \ \alpha_j \in \mathbb{R} \ \wedge \ \alpha_{j+1} \in \mathbb{R}$ **then**
      $\gamma_{j+1} = \overline{\beta_j} + \alpha_{j+1}$, $\delta_j := \alpha_{j+1}^2 + 2\,\mathrm{Re}(\beta_j)\alpha_{j+1} + |\beta_j|^2$.
      $W_{j+1} = W_{j-1} - (\mathrm{Re}(\gamma_j + \gamma_{j+1}))\,\mathrm{Re}(V_j) - \frac{\mathrm{Re}(\gamma_j)\,\mathrm{Re}(\gamma_{j+1}) - \mathrm{Im}(\beta_j)^2}{\mathrm{Im}(\beta_j)}\,\mathrm{Im}(V_j)$.
      $T_j = T_{j-1} - (\mathrm{Re}(\gamma_j) + \mathrm{Re}(\gamma_{j+1}))S_j$.
      $S_{j+1} = -(B^{(k)} + \alpha_{j+1}I_m)^{-T}S_j$, $T_{j+1} = T_j + \delta_j S_{j+1}$.
      $Z_+ = [\mathrm{Re}(V_j),\ \mathrm{Im}(V_j)]$, $Y_+ = [S_j,\ S_{j+1}]$.
      $\Gamma_+ = \hat{\Gamma}_j^{(3a)}$ from (4.19), $j = j + 1$, $h = 2$.
    if *Case 3b:* $\beta_j \in \mathbb{R} \ \wedge \ \beta_{j+1} \in \mathbb{R} \ \wedge \ \alpha_j \in \mathbb{C}$ **then**
      $\gamma_{j+1} = \beta_{j+1} + \overline{\alpha_j}$, $\delta_j := \beta_{j+1}^2 + 2\,\mathrm{Re}(\alpha_j)\beta_{j+1} + |\alpha_j|^2$.
      $W_j = W_{j-1} - (\mathrm{Re}(\gamma_j) + \mathrm{Re}(\gamma_{j+1}))V_j$.
      $V_{j+1} = (A^{(k)} + \beta_{j+1}I_n)^{-1}V_j,\ W_{j+1} = W_j + \delta_j V_{j+1}$.
      $T_{j+1} = T_{j-1} - (\mathrm{Re}(\gamma_j) + \mathrm{Re}(\gamma_{j+1}))\,\mathrm{Re}(S_j) - \frac{\mathrm{Im}(\alpha_j)^2 - \mathrm{Re}(\gamma_j)\,\mathrm{Re}(\gamma_{j+1})}{\mathrm{Im}(\alpha_j)}\,\mathrm{Im}(S_j)$.
      $Z_+ = [V_j,\ V_{j+1}]$,$Y_+ = [\mathrm{Re}(S_j),\ \mathrm{Im}(S_j)]$.
      $\Gamma_+ = \hat{\Gamma}_j^{(3b)}$ from (4.20), $j = j + 1$, $h = 2$.
    $Z_j^{(k)} = [Z_{j-h}^{(k)},\ Z_+]$, $Y_j^{(k)} = [Y_{j-h}^{(k)},\ Y_+]$, $\Gamma_j^{(k)} = \mathrm{diag}(\Gamma_{j-h}^{(k)},\ \Gamma_+)$, $k = k + 1$.
  $K^{(k)} = Z_j^{(k)}\Gamma_j^{(k)}((Y_j^{(k)})^T F)$, $H^{(k)} = Y_j^{(k)}(\Gamma_j^{(k)})^T(Z_j^{(k)})^H G$, $t_j = (r + p)j$.

APPENDIX B

THESES

1. This thesis deals with numerical methods for large-scale algebraic matrix equations. Linear matrix equations of Sylvester and Lyapunov type, but also quadratic equations of Riccati type, are considered.

2. The low-rank phenomenon exhibited by the solutions of large-scale algebraic matrix equations is reviewed. This leads to the concept of computing approximate solutions in the form of low-rank factorizations which is the backbone of the investigated algorithms.

3. For Sylvester and Lyapunov equations, the main emphasis lies on methods based on the low-rank version of the alternating directions implicit (LR-ADI) iteration.

4. It is shown that in each step of the LR-ADI iteration, the residual with respect to the current approximation of the solution can be exactly expressed as a low-rank factorization. The rank of the residual is bounded from above by the rank of the inhomogeneity of the matrix equation. This leads to a novel way to formulate the LR-ADI iteration and to efficient approaches for computing the residual norm.

5. The ADI iteration demands shift parameters which have a significant influence on the rate of the error reduction. If the considered matrix equation is defined by real nonsymmetric matrices, these shift parameters can occur in complex conjugated pairs. Based on newly discovered interconnections of the iteration data associated to these complex conjugated pairs, new strategies to reduce the amount of the resulting complex arithmetic operations and storage are presented.

6. The dependence on shift parameters is a conceptual disadvantage of the LR-ADI iteration, because shifts of high quality are usually hard to obtain for large-scale matrices. Novel shift generation and selection approaches are developed, where the shifts are efficiently and automatically computed during the course of the iteration, making the LR-ADI iteration basically parameter free.

7. The newly established low-rank factorization of the residual can be exploited to develop a shift generation strategy based on a minimization of the residual norm.

8. The proposed self-generating shifts outperform existing shift selection approaches in the majority of experiments, leading to fewer iteration numbers and less consumed computation time.

9. Equipped with the established numerical enhancements and the new shift strategies, the LR-ADI iteration compares well against competitive approaches, e.g., projection based methods using rational or extended Krylov subspaces.

10. Large-scale symmetric and nonsymmetric algebraic Riccati equations can be solved by Newton type methods, where the occurring Lyapunov and Sylvester equations can be dealt with the LR-ADI iteration, including all of the proposed performance improvements. Adding a Galerkin projection to the Newton iteration can further accelerate the algorithms.

11. Balanced truncation model order reduction requires the solution of two Lyapunov equations which are adjoint to each other. These can be handled simultaneously by a dual LR-ADI iteration, which can be equipped with specialized stopping criteria connected to the model reduction process.

12. Balanced truncation restricted to limited frequency regions leads to Lyapunov equations, where the inhomogeneity involves a nonlinear function of a matrix. It is investigated why the eigenvalues of the solutions of these special Lyapunov equations often decay faster compared to the equations arising in classical balanced truncation. Results explaining the influence of the frequency limits on this accelerated eigenvalue decay are established.

13. A projection approach employing rational Krylov subspaces is proposed which can handle both the matrix function and the Lyapunov equation by a single, computationally efficient algorithm. As result, frequency-limited balanced truncation can be carried with a numerical effort comparable to unrestricted balanced truncation and yields reduced order models of higher accuracy in the considered frequency region.

BIBLIOGRAPHY

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

[2] M. I. Ahmad, D. B. Szyld, and M. B. van Gijzen, *Preconditioned multi-shift BiCG for $\mathcal{H}_2$-optimal model reduction*, Tech. Rep. 12-06-15, Department of Mathematics, Temple University, June 2012. Revised March 2013.

[3] A. C. Antoulas, *Approximation of Large-Scale Dynamical Systems*, SIAM, Philadelphia, PA, 2005.

[4] A. C. Antoulas, D. C. Sorensen, and Y. Zhou, *On the Decay Rate of Hankel Singular Values and Related Issues*, Sys. Control Lett., 46 (2002), pp. 323–342.

[5] W. F. Arnold, III and A. J. Laub, *Generalized Eigenproblem Algorithms and Software for Algebraic Riccati Equations*, Proc. IEEE, 72 (1984), pp. 1746–1754.

[6] Z. Bai, J. Demmel, J. Dongarra, A. Ruhe, and H. van der Vorst, *Templates for the Solution of Algebraic Eigenvalue Problems: A Practical Guide*, SIAM, Philadelphia, PA, 2000.

[7] J. Baker, M. Embree, and J. Sabino, *Fast singular value decay for Lyapunov solutions with nonnormal coefficients*, SIAM J. Matrix Anal. Appl., 36 (2015), pp. 656–668.

[8] R. E. Bank and T. F. Chan, *An analysis of the composite step biconjugate gradient method*, Numer. Math., 66 (1993), pp. 295–319.

[9] ——, *A composite step bi-conjugate gradient algorithm for nonsymmetric linear systems*, Numer. Algorithms, 7 (1994), pp. 1–16.

[10] E. Bänsch, P. Benner, J. Saak, and H. K. Weichelt, *Riccati-based boundary feedback stabilization of incompressible Navier-Stokes flows*, SIAM J. Sci. Comput., 37 (2015), pp. A832–A858.

[11] L. Bao, Y. Lin, and Y. Wei, *Krylov subspace methods for the generalized Sylvester equation*, Appl. Math. and Comput., 175 (2006), pp. 557–573.

[12] ——, *A new projection method for solving large Sylvester equations*, Appl. Numer. Math., 57 (2007), pp. 521–532.

[13] S. Barrachina, P. Benner, and E. S. Quintana-Ortí, *Efficient algorithms for generalized algebraic Bernoulli equations based on the matrix sign function*, Numer. Algorithms, 46 (2007), pp. 351–368.

[14] R. Bartels and G. Stewart, *Solution of the Matrix Equation $AX + XB = C$: Algorithm 432*, Comm. ACM, 15 (1972), pp. 820–826.

[15] B. Beckermann and A. Gryson, *Extremal Rational Functions on Symmetric Discrete Sets and Superlinear Convergence of the ADI Method*, Constr. Approx., 32 (2010), pp. 393–428.

[16] P. Benner, *Contributions to the Numerical Solution of Algebraic Riccati Equations and Related Eigenvalue Problems*, Dissertation, Fakultät für Mathematik, TU Chemnitz–Zwickau, 09107 Chemnitz (Germany), Feb. 1997.

[17] ——, *Factorized Solution of Sylvester Equations with Applications in Control*, in Proc. Intl. Symp. Math. Theory Networks and Syst. MTNS 2004, `http://www.mtns2004.be`, 2004.

[18] ——, *Solving Large-Scale Control Problems*, IEEE Control Systems Magazine, 14 (2004), pp. 44–59.

[19] ——, *The Matrix Factorization Paradigm in Solving Matrix Equations.* Householder Symposium XVI, Seven Springs Mountain Resort, Champion, Pennsylvania, USA, see `http://www.mpi-magdeburg.mpg.de/mpcsc/benner/talks/hh05.pdf`., 2005.

[20] P. Benner and T. Breiten, *Low rank methods for a class of generalized Lyapunov equations and related issues*, Numer. Math., 124 (2013), pp. 441–470.

[21] ——, *On optimality of approximate low rank solutions of large-scale matrix equations*, Sys. Control Lett., 67 (2014), pp. 55–64.

[22] ——, *Rational interpolation methods for symmetric Sylvester equations*, Electr. Trans. Num. Anal., 42 (2014), pp. 147–164.

[23] P. Benner and Z. Bujanović, *On the solution of large-scale algebraic Riccati equations by using low-dimensional invariant subspaces*, Linear Algebra Appl., 488 (2016), pp. 430–459.

[24] P. Benner, Z. Bujanović, P. Kürschner, and J. Saak, *The RADI algorithm for solving large-scale algebraic Riccati equations*, e-prints 1510.00040, ArXiv, Sept. 2015. Available from http://arxiv.org/abs/1510.00040v2.

[25] P. Benner and R. Byers, *Newton's Method with Exact Line Search for Solving the Algebraic Riccati Equation*, Tech. Rep. SPC 95_24, Fakultät für Mathematik, TU Chemnitz–Zwickau, 09107 Chemnitz, FRG, 1995. Available from `http://www.tu-chemnitz.de/sfb393/spc95pr.html`.

[26] ——, *An Exact Line Search Method for Solving Generalized Continuous-Time Algebraic Riccati Equations*, IEEE Trans. Automat. Control, 43 (1998), pp. 101–107.

[27] P. Benner and H. Fassbender, *On the numerical solution of large-scale sparse discrete-time Riccati equations*, Adv. Comput. Math., 35 (2011), pp. 119–147.

[28] P. Benner, M. Hochstenbach, and P. Kürschner, *Model order reduction of large-scale dynamical systems with Jacobi-Davidson style eigensolvers*, in Proceedings of CCCA11, 2011. IEEE Catalog Number: CFP1154M-ART.

[29] P. Benner, G. E. Khoury, and M. Sadkane, *On the Squared Smith Method for Large-Scale Stein Equations*, Numer. Lin. Alg. Appl., 21 (2014), pp. 645–665.

[30] P. Benner, M. Köhler, and J. Saak, *Sparse-Dense Sylvester Equations in $\mathcal{H}_2$-Model Order Reduction*, Tech. Rep. MPIMD/11-11, Max Planck Institute Magdeburg Preprints, 2011. Available from `http://www.mpi-magdeburg.mpg.de/preprints/2011/11/`.

[31] P. Benner, B. Kranz, J. Saak, and M. M. Uddin, *Efficient Reduced Order State Space Model Computation for a Class of Second Order Index One Systems*, Proc. Appl. Math. Mech., 12 (2012), pp. 699–700.

[32] P. Benner and P. Kürschner, *Computing Real Low-rank Solutions of Sylvester equations by the Factored ADI Method*, Comput. Math. Appl., 67 (2014), pp. 1656–1672.

[33] P. Benner, P. Kürschner, and J. Saak, *A Goal-Oriented Dual LRCF-ADI for Balanced Truncation*, vol. 7 of Proceedings of the MathMod 2012, IFAC-PapersOnline, Mathematical Modelling, Vienna Univ. of Technology, 2012, pp. 752–757.

[34] ——, *Avoiding complex arithmetic in the low-rank ADI method efficiently*, Proc. Appl. Math. Mech., 12 (2012), pp. 639–640.

[35] ——, *Real versions of low-rank ADI methods with complex shifts*, Tech. Rep. MPIMD/12-11, Max Planck Institute Magdeburg Preprints, 2012. Available from `http://www.mpi-magdeburg.mpg.de/preprints/`.

[36] ——, *A Reformulated Low-Rank ADI Iteration with Explicit Residual Factors*, Proc. Appl. Math. Mech., 13 (2013), pp. 585–586.

[37] ——, *An Improved Numerical Method for Balanced Truncation for Symmetric Second Order Systems*, Math. Comput. Model. Dyn. Sys., 19 (2013), pp. 593–615.

[38] ——, *Efficient Handling of Complex Shift Parameters in the Low-Rank Cholesky Factor ADI Method*, Numer. Algorithms, 62 (2013), pp. 225–251.

[39] ——, *Self-Generating and Efficient Shift Parameters in ADI Methods for Large Lyapunov and Sylvester Equations*, Electr. Trans. Num. Anal., 43 (2014), pp. 142–162.

[40] ——, *Frequency-Limited Balanced Truncation with Low-Rank Approximations*, SIAM J. Sci. Comput., 38 (2016), pp. A471–A499.

[41] ——, *Low-Rank Newton-ADI methods for Large Nonsymmetric Algebraic Riccati Equations*, J. Frankl. Inst., 353 (2016), pp. 1147–1167.

[42] P. Benner, J.-R. Li, and T. Penzl, *Numerical Solution of Large Lyapunov equations, Riccati Equations, and Linear-Quadratic Control Problems*, Numer. Lin. Alg. Appl., 15 (2008), pp. 755–777.

[43] P. Benner, R.-C. Li, and N. Truhar, *On the ADI Method for Sylvester Equations*, J. Comput. Appl. Math., 233 (2009), pp. 1035–1045.

[44] P. Benner, V. Mehrmann, and D. C. Sorensen, *Dimension Reduction of Large-Scale Systems*, vol. 45 of Lect. Notes Comput. Sci. Eng., Springer-Verlag, Berlin/Heidelberg, Germany, 2005.

[45] P. Benner, H. Mena, and J. Saak, *On the Parameter Selection Problem in the Newton-ADI Iteration for Large-Scale Riccati Equations*, Electr. Trans. Num. Anal., 29 (2008), pp. 136–149.

[46] P. Benner, E. Quintana-Ortí, and G. Quintana-Ortí, *Solving Stable Sylvester Equations via Rational Iterative Schemes*, J. Sci. Comp., 28 (2005 (electronic)), pp. 51–83.

[47] P. Benner and J. Saak, *Efficient numerical solution of the LQR-problem for the heat equation*, Proc. Appl. Math. Mech., 4 (2004), pp. 648–649.

[48] ——, *A Semi-Discretized Heat Transfer Model for Optimal Cooling of Steel Profiles*, in Dimension Reduction of Large-Scale Systems, P. Benner, V. Mehrmann, and D. Sorensen, eds., vol. 45 of Lect. Notes Comput. Sci. Eng., Springer-Verlag, Berlin/Heidelberg, Germany, 2005, pp. 353–356.

[49] ——, *Efficient solution of large scale Lyapunov and Riccati equations arising in model order reduction problems*, Proc. Appl. Math. Mech., 8 (2008), pp. 10085–10088.

[50] ——, *A Galerkin-Newton-ADI Method for Solving Large-Scale Algebraic Riccati Equations*, Preprint SPP1253-090, SPP1253, January 2010. `http://www.am.uni-erlangen.de/home/spp1253/wiki/index.php/Preprints`.

[51] ——, *Efficient Balancing based MOR for Large Scale Second Order Systems*, Math. Comput. Model. Dyn. Sys., 17 (2011), pp. 123–143.

[52] ——, *Numerical solution of large and sparse continuous time algebraic matrix Riccati and Lyapunov equations: a state of the art survey*, GAMM Mitteilungen, 36 (2013), pp. 32–52.

[53] P. Benner, J. Saak, and M. M. Uddin, *Second Order to Second Order Balancing for Index-1 Vibrational Systems*, in 7th International Conference on Electrical & Computer Engineering (ICECE) 2012, IEEE, 2012, pp. 933–936.

[54] P. Benner, J. Saak, and M. M. Uddin, *Structure Preserving MOR for Large Sparse Second Order Index-1 Systems and Application to a Mechatronic Model*, Preprint MPIMD/14-23, Max Planck Institute Magdeburg, Dec. 2014. Available from `http://www.mpi-magdeburg.mpg.de/preprints/`.

[55] D. A. Bini, B. Iannazzo, and B. Meini, *Numerical Solution of Algebraic Riccati Equations*, SIAM, 2011.

[56] D. A. Bini, B. Iannazzo, B. Meini, and F. Poloni, *Nonsymmetric algebraic Riccati equations associated with an M-matrix: recent advances and algorithms.*, in Matrix Methods: Theory, Algorithms and Applications, V. Olshevsky and E. Tyrtyshnikov, eds., World Scientific Publishing, 2010, ch. 10, pp. 176–209.

[57] D. A. Bini, B. Iannazzo, and F. Poloni, *A Fast Newton's Method for a Nonsymmetric Algebraic Riccati Equation*, SIAM J. Matrix Anal. Appl., 30 (2008), pp. 276–290.

[58] C. H. Bischof and G. Quintana-Ortí, *Algorithm 782: Codes for rank-revealing QR factorizations of dense matrices.*, ACM Trans. Math. Software, 24 (1998), pp. 254–257.

[59] M. Bollhöfer and A. K. Eppler, *A Structure Preserving FGMRES Method for Solving Large Lyapunov Equations*, in Progress in Industrial Mathematics at ECMI 2010, M. Günther, A. Bartel, M. Brunk, S. Schöps, and M. Striebel, eds., Mathematics in Industry, Springer Berlin Heidelberg, 2012, pp. 131–136.

[60] A. Bouhamidi, M. Hached, M. Heyouni, and K. Jbilou, *A preconditioned block Arnoldi method for large Sylvester matrix equations*, Numer. Lin. Alg. Appl., 20 (2011), pp. 208–219.

[61] M. Brand, *Fast online SVD revisions for lightweight recommender systems*, in SIAM International Conference on Data Mining, 2003.

[62] J. Brandts, *The Riccati algorithm for eigenvalues and invariant subspaces of matrices with inexpensive action*, Linear Algebra Appl., 358 (2003), pp. 335–365.

[63] A. Bunse-Gerstner and H. Fassbender, *Breaking Van Loan's Curse: A Quest forStructure-Preserving Algorithms for Dense Structured Eigenvalue Problems*, in Numerical Algebra, Matrix Theory, Differential-Algebraic Equations and

Control Theory, P. Benner, M. Bollhöfer, D. Kressner, C. Mehl, and T. Stykel, eds., Springer International Publishing, 2015, pp. 3–23.

[64] D. Calvetti and L. Reichel, *Application of ADI iterative methods to the restoration of noisy images*, SIAM J. Matrix Anal. Appl., 17 (1996), pp. 165–186.

[65] Y. Chahlaoui, D. Lemonnier, A. Vandendorpe, and P. Van Dooren, *Second-order balanced truncation*, Linear Algebra Appl., 415 (2006), pp. 373–384.

[66] Y. Chahlaoui and P. Van Dooren, *A collection of benchmark examples for model reduction of linear time invariant dynamical systems*, SLICOT Working Note 2002–2, University of Manchester, Feb. 2002. Available from `www.slicot.org`.

[67] D. Chu, X. Liu, and V. Mehrmann, *A numerical method for computing the Hamiltonian Schur form*, Numer. Math., 105 (2006), pp. 375–412.

[68] E. K.-W. Chu, *The solution of the matrix equations $AXB - CXD = E$ and $(YA - DZ, YC - BZ) = (E, F)$*, Linear Algebra Appl., 93 (1987), pp. 93–105.

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

[70] E. K.-W. Chu, H.-Y. Fan, W.-W. Lin, and C.-S. Wang, *Structure-preserving algorithms for periodic discrete-time algebraic Riccati equations*, Internat. J. Control, 77 (2004), pp. 767–788.

[71] T. Damm, *Direct methods and ADI-preconditioned Krylov subspace methods for generalized Lyapunov equations*, Numer. Lin. Alg. Appl., 15 (2008), pp. 853–871.

[72] B. N. Datta, *Numerical Methods for Linear Control Systems*, Elsevier Academic Press, 2004.

[73] P. I. Davies and N. J. Higham, *Computing $f(A)b$ for Matrix Functions f*, vol. 47 of Lecture Notes in Computational Science and Engineering, Springer Berlin Heidelberg, 2005, pp. 15–24.

[74] T. A. Davis, *Direct Methods for Sparse Linear Systems (Fundamentals of Algorithms 2)*, SIAM, Philadelphia, PA, 2006.

[75] D. Day and M. A. Heroux, *Solving Complex-Valued Linear Systems via Equivalent Real Formulations*, SIAM J. Sci. Comput., 23 (2001), pp. 480–498.

[76] B. De Moor and J. David, *Total Linear Least Squares and the Algebraic Riccati Equation*, Sys. Control Lett., 18 (1992), pp. 329–337.

[77] R. S. Dembo, S. C. Eisenstat, and T. Steihaug, *Inexact Newton Methods*, SIAM J. Numer. Anal., 19 (1982), pp. 400–408.

[78] J. W. Demmel, *Three methods for refining estimates of invariant subspaces*, Computing, 38 (1987), pp. 43–57.

[79] T. A. Driscoll, N. Hale, and L. N. Trefethen, *Chebfun guide*, Pafnuty Publications, Oxford, 2014.

[80] Z. Drmac, *Accurate computation of the product-induced singular value decomposition with applications.*, SIAM J. Numer. Anal., 35 (1998), pp. 1969–1994.

[81] V. Druskin and L. A. Knizhnerman, *Extended Krylov Subspaces: Approximation of the Matrix Square Root and Related Functions*, SIAM J. Matrix Anal. Appl., 19 (1998), pp. 755–771.

[82] V. Druskin, L. A. Knizhnerman, and V. Simoncini, *Analysis of the rational Krylov subspace and ADI methods for solving the Lyapunov equation*, SIAM J. Numer. Anal., 49 (2011), pp. 1875–1898.

[83] V. Druskin and V. Simoncini, *Adaptive rational Krylov subspaces for large-scale dynamical systems*, Systems & Control Letters, 60 (2011), pp. 546–560.

[84] V. Druskin, V. Simoncini, and M. Zaslavsky, *Adaptive Tangential Interpolation in Rational Krylov Subspaces for MIMO Dynamical Systems*, SIAM J. Matrix Anal. Appl., 35 (2014), pp. 476–498.

[85] L. Du, T. Sogabe, and S.-L. Zhang, *IDR(s) for solving shifted nonsymmetric linear systems*, J. Comput. Appl. Math., 274 (2015), pp. 35–43.

[86] X. Du, P. Benner, G. Yang, and D. Ye, *Balanced truncation of linear time-invariant systems at a single frequency*, Preprint MPIMD/13-02, Max Planck Institute Magdeburg, Jan. 2013. Available from `http://www.mpi-magdeburg.mpg.de/preprints/`.

[87] I. Duff, A. Erisman, and J. Reid, *Direct methods for sparse matrices*, Clarendon Press, Oxford, UK, 1989.

[88] S. C. Eisenstat and H. F. Walker, *Choosing the Forcing Terms in an Inexact Newton Method*, SIAM J. Sci. Comput., 17 (1996), pp. 16–32.

[89] A. El Guennouni, K. Jbilou, and A. Riquet, *Block Krylov Subspace Methods for Solving Large Sylvester Equations*, Numer. Algorithms, 29 (2002), pp. 75–96.

[90] M. Epton, *Methods for the solution of $AXD - BXC = E$ and its application in the numerical solution of implicit ordinary differential equations*, BIT, 20 (1980), pp. 341–345.

[91] J. Fehr, *Automated and Error Controlled Model Reduction in Elastic Multibody Systems*, PhD thesis, Institut für Technische und Numerische Mechanik, Universität Stuttgart, 2011.

[92] J. Fehr, P. Eberhard, and M. Lehner, *Improving the Reduction Process in Flexible Multibody Dynamics by the Use of 2nd Order Position Gramian Matrices*, in ENOC-2008, St. Petersburg, Russia, 2008.

[93] J. Fehr, M. Fischer, B. Haasdonk, and P. Eberhard, *Greedy-based approximation of frequency-weighted Gramian matrices for model reduction in multibody dynamics*, ZAMM - J. Appl. Math. Mech., 93 (2013), pp. 501–519.

[94] F. Feitzinger, T. Hylla, and E. W. Sachs, *Inexact Kleinman-Newton Method for Riccati Equations*, SIAM J. Matrix Anal. Appl., 31 (2009), pp. 272–288.

[95] M. Fiedler, *Hankel and Loewner matrices*, Linear Algebra Appl., 58 (1984), pp. 75–95.

[96] G. M. Flagg and S. Gugercin, *On the ADI method for the Sylvester equation and the optimal $\mathcal{H}_2$ points*, Appl. Numer. Math., 64 (2013), pp. 50–58.

[97] R. Fletcher, *Conjugate gradient methods for indefinite systems*, in Numerical Analysis, G. Watson, ed., vol. 506 of Lecture Notes in Mathematics, Springer Berlin / Heidelberg, 1976, pp. 73–89.

[98] F. Freitas, J. Rommes, and N. Martins, *Gramian-Based Reduction Method Applied to Large Sparse Power System Descriptor Models*, IEEE Trans. Power Systems, 23 (2008), pp. 1258–1270.

[99] R. W. Freund, M. Gutknecht, and N. M. Nachtigal, *An Implementation of the Look-Ahead Lanczos Algorithm for Non-Hermitian Matrices*, SIAM J. Sci. Comput., 14 (1993), pp. 137–158.

[100] R. W. Freund and M. Malhotra, *A block QMR algorithm for non-Hermitian linear systems with multiple right-hand sides*, Linear Algebra Appl., 254 (1997), pp. 119–157.

[101] R. W. Freund and N. M. Nachtigal, *QMR: a quasi-minimal residual method for non-Hermitian linear systems*, Numer. Math., 60 (1991), pp. 315–339.

[102] A. Frommer, *BiCGStab(ℓ) for Families of Shifted Linear Systems*, Computing, 70 (2003), pp. 87–109.

[103] A. Frommer and U. Glässner, *Restarted GMRES for Shifted Linear Systems*, SIAM J. Sci. Comput., 19 (1998), pp. 15–26.

[104] A. Frommer and V. Simoncini, *Matrix Functions*, in Model Order Reduction: Theory, Research Aspects and Applications, W. Schilders, H. A. van der Vorst, and J. Rommes, eds., vol. 13 of Mathematics in Industry, Springer Berlin Heidelberg, 2008, pp. 275–303.

[105] J. Gardiner, A. J. Laub, J. Amato, and C. Moler, *Solution of the Sylvester Matrix Equation $AXB + CXD = E$*, ACM Trans. Math. Software, 18 (1992), pp. 223–231.

[106] J. D. Gardiner and A. J. Laub, *A Generalization of the Matrix-Sign-Function Solution for Algebraic Riccati Equations*, Internat. J. Control, 44 (1986), pp. 823–832.

[107] W. Gawronski and J. Juang, *Model reduction in limited time and frequency intervals*, Int. J. Syst. Sci., 21 (1990), pp. 349–376.

[108] I. Gohberg and I. Koltracht, *Triangular factors of Cauchy and Vandermonde matrices*, Integr. Equat. Oper. Th., 26 (1996), pp. 46–59.

[109] G. H. Golub, S. Nash, and C. F. Van Loan, *A Hessenberg–Schur method for the problem $AX + XB = C$*, IEEE Trans. Automat. Control, AC-24 (1979), pp. 909–913.

[110] G. H. Golub, M. Stoll, and A. Wathen, *Approximation of the Scattering Amplitude and Linear Systems*, Electr. Trans. Num. Anal., 31 (2008), pp. 178–203.

[111] G. H. Golub and C. F. Van Loan, *Matrix Computations*, Johns Hopkins University Press, Baltimore, fourth ed., 2013.

[112] L. Grasedyck, *Existence and computation of low Kronecker-rank approximations for large linear systems of tensor product structure*, Computing, 72 (2004), pp. 247–265.

[113] ——, *Existence of a low rank or H-matrix approximant to the solution of a Sylvester equation*, Numer. Lin. Alg. Appl., 11 (2004), pp. 371–389.

[114] E. J. Grimme, *Krylov projection methods for model reduction*, PhD thesis, Univ. of Illinois at Urbana-Champaign, USA, 1997.

[115] M. Gu and S. C. Eisenstat, *A Stable And Fast Algorithm For Updating The Singular Value Decomposition*, technical report, Department of Computer Science, Yale University, 1994.

[116] S. Gugercin and A. C. Antoulas, *A survey of model reduction by balanced truncation and some new results*, Internat. J. Control, 77 (2004), pp. 748–766.

[117] S. Gugercin, A. C. Antoulas, and C. Beattie, *$\mathcal{H}\_2$ Model Reduction for Large-Scale Dynamical Systems*, SIAM J. Matrix Anal. Appl., 30 (2008), pp. 609–638.

[118] S. Gugercin and J.-R. Li, *Smith-type methods for balanced truncation of large systems*, in Dimension Reduction of Large-Scale Systems, P. Benner, V. Mehrmann, and D. Sorensen, eds., vol. 45 of Lect. Notes Comput. Sci. Eng., Springer-Verlag, Berlin/Heidelberg, Germany, 2005, pp. 49–82.

[119] S. Gugercin, D. C. Sorensen, and A. C. Antoulas, *A modified low-rank Smith method for large-scale Lyapunov equations*, Numer. Algorithms, 32 (2003), pp. 27–55.

[120] C. Guo and N. Higham, *Iterative Solution of a Nonsymmetric Algebraic Riccati Equation*, SIAM J. Matrix Anal. Appl., 29 (2007), pp. 396–412.

[121] C.-H. Guo, *Efficient methods for solving a nonsymmetric algebraic Riccati equation arising in stochastic fluid models*, J. Comput. Appl. Math, 192 (2006), pp. 353–373.

[122] C.-H. Guo and A. J. Laub, *On the Iterative Solution of a Class of Nonsymmetric Algebraic Riccati Equations*, SIAM J. Matrix Anal. Appl., 22 (2000), pp. 376–391.

[123] S. Güttel, *Rational Krylov approximation of matrix functions: Numerical methods and optimal pole selection*, GAMM Mitteilungen, 36 (2013), pp. 8–31.

[124] S. Hammarling, *Numerical Solution of the Stable, Non-negative Definite Lyapunov Equation*, IMA J. Numer. Anal., 2 (1982), pp. 303–323.

[125] M. Heinkenschloss, H. K. Weichelt, P. Benner, and J. Saak, *Inexact low-rank Newton-ADI method for large-scale algebraic Riccati equations*, Tech. Rep. MPIMD/15-06, Max Planck Institute Magdeburg Preprints, 2015. Available from `http://www.mpi-magdeburg.mpg.de/preprints/`.

[126] G. A. Hewer, *An Iterative Technique for the Computation of Steady State Gains for the Discrete Optimal Regulator*, IEEE Trans. Automat. Control, AC-16 (1971), pp. 382–384.

[127] M. Heyouni and K. Jbilou, *An extended block Arnoldi algorithm for large-scale solutions of the continuous-time algebraic Riccati equation*, Electr. Trans. Num. Anal., 33 (2009), pp. 53–62.

[128] N. J. Higham, *Functions of Matrices: Theory and Computation*, Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2008.

[129] D. Hinrichsen and A. J. Pritchard, *Mathematical Systems Theory I*, Springer-Verlag, Berlin, 2005.

[130] M. Hochbruck and M. Hochstenbach, *Subspace extraction for matrix functions*, tech. rep., 2005.

[131] R. Horn and C. Johnson, *Matrix Analysis*, Cambridge University Press, Cambridge, 1985.

[132] ——, *Topics in Matrix Analysis*, Cambridge University Press, Cambridge, 1991.

[133] W. D. Hoskins, D. S. Meek, and D. J. Walton, *The Numerical Solution of $A'Q + QA = -C$*, IEEE Trans. Automat. Control, AC-22 (1977), pp. 882–883.

[134] D. Hu and L. Reichel, *Krylov-subspace methods for the Sylvester equation*, Linear Algebra Appl., 172 (1992), pp. 283–313.

[135] T. Hylla, *Extension of inexact Kleinman-Newton methods to a general monotonicity preserving convergence theory*, dissertation, Universität Trier, 2011.

[136] M. Imran and A. Ghafoor, *Model reduction of descriptor systems using frequency limited Gramians*, J. Frankl. Inst., 352 (2015), pp. 33–51.

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

[138] K. Jbilou, *Low rank approximate solutions to large Sylvester matrix equations*, Appl. Math. Comput., 177 (2006), pp. 365–376.

[139] Z. Jia and H. Lv, *A posteriori error estimates of Krylov subspace approximations to matrix functions*, Numer. Algorithms, 69 (2015), pp. 1–28.

[140] J. Juang and I. D. Chen, *Iterative solution for a certain class of algebraic matrix Riccati equations arising in transport theory*, Transport Theor. Stat., 22 (1993), pp. 65–80.

[141] D. Kahaner, C. Moler, S. Nash, and G. E. Forsythe, *Numerical Methods and Software*, Prentice-Hall, Englewood Cliffs, NJ, 1988.

[142] C. T. Kelley, *Iterative Methods for Linear and Nonlinear Equations*, SIAM, Philadelphia, PA, 1995.

[143] C. Kenney and A. J. Laub, *The Matrix Sign Function*, IEEE Trans. Automat. Control, 40 (1995), pp. 1330–1348.

[144] C. Kenney, A. J. Laub, and P. M. Papadopoulos, *Matrix-sign algorithms for Riccati equations*, IMA J. Math. Contr. Info., 3 (1992), pp. 331–344.

[145] D. Kleinman, *On an Iterative Technique for Riccati Equation Computations*, IEEE Trans. Automat. Control, AC-13 (1968), pp. 114–115.

[146] L. A. Knizhnerman, *Calculation of functions of unsymmetric matrices using Arnoldi's method*, Comput. Math. Math. Phys., 31 (1992), pp. 1–9.

[147] L. A. Knizhnerman and V. Simoncini, *A new investigation of the extended Krylov subspace method for matrix function evaluations*, Numer. Lin. Alg. Appl., 17 (2010), pp. 615–638.

[148] P. Kunkel and V. Mehrmann, *Differential-Algebraic Equations: Analysis and Numerical Solution*, Textbooks in Mathematics, EMS Publishing House, 2006.

[149] P. Lancaster, *On eigenvalues of matrices dependent on a parameter*, Numer. Math., 6 (1964), pp. 377–387.

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

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

[152] N. Lang, H. Mena, and J. Saak, *An LDL^T factorization based ADI algorithm for solving large scale differential matrix equations*, Proc. Appl. Math. Mech., 14 (2014), pp. 827–828.

[153] ——, *On the benefits of the LDL^T factorization for large-scale differential matrix equation solvers*, Linear Algebra Appl., 480 (2015), pp. 44–71.

[154] A. J. Laub, *A Schur Method for Solving Algebraic Riccati Equations*, IEEE Trans. Automat. Control, AC-24 (1979), pp. 913–921.

[155] ——, *Invariant Subspace Methods for the Numerical Solution of Riccati Equations*, in The Riccati Equation, S. Bittanti, A. J. Laub, and J. C. Willems, eds., Springer-Verlag, Berlin, 1991, pp. 163–196.

[156] A. J. Laub, M. T. Heath, C. C. Paige, and R. C. Ward, *Computation of system balancing transformations and other application of simultaneous diagonalization algorithms*, IEEE Trans. Automat. Control, 32 (1987), pp. 115–122.

[157] M. Lehner, *Modellreduktion in elastischen Mehrkörpersystemen (in German)*, Schriften aus dem Institut für Technische und Numerische Mechanik der Universität Stuttgart, 10 (2007). Dissertation.

[158] N. Levenberg and L. Reichel, *A generalized ADI iterative method*, Numer. Math., 66 (1993), pp. 215–233.

[159] A. Lewis, *The mathematics of eigenvalue optimization*, Mathem. Prog., 97 (2003), pp. 155–176.

[160] J.-R. Li, *Model Reduction of Large Linear Systems via Low Rank System Gramians*, PhD thesis, Massachusettes Institute of Technology, September 2000.

[161] J.-R. Li and J. White, *Low Rank Solution of Lyapunov Equations*, SIAM J. Matrix Anal. Appl., 24 (2002), pp. 260–280.

[162] R.-C. Li and N. Truhar, *On the ADI Method for Sylvester Equations*, Technical Report 2008-2, Department of Mathematics, University of Texas at Arlington, 2008. Available at `http://www.uta.edu/math/preprint/rep2008_02.pdf`.

[163] T. Li, E. K.-w. Chu, J. Juang, and W.-W. Lin, *Solution of a Nonsymmetric Algebraic Riccati Equation from a Two-dimensional Transport Model*, Linear Algebra Appl., 434 (2011), pp. 201–214.

[164] T. Li, E. K.-w. Chu, Y. Kuo, and W.-W. Lin, *Solving Large-Scale Nonsymmetric Algebraic Riccati Equations by Doubling*, SIAM J. Matrix Anal. Appl., 34 (2013), pp. 1129–1147.

[165] T. Li, P. C.-Y. Weng, E. K.-w. Chu, and W.-W. Lin, *Large-scale Stein and Lyapunov equations, Smith method, and applications*, Numer. Algorithms, 63 (2013), pp. 727–752.

[166] Y. Lin and V. Simoncini, *Minimal residual methods for large scale Lyapunov equations*, Appl. Numer. Math., 72 (2013), pp. 52–71.

[167] ——, *A new subspace iteration method for the algebraic Riccati equation*, Numer. Lin. Alg. Appl., 22 (2015), pp. 26–47.

[168] J. Lu and D. L. Darmofal, *A Quasi-Minimal Residual Method for Simultaneous Primal-Dual Solutions and Superconvergent Functional Estimates*, SIAM J. Sci. Comput., 24 (2002), pp. 1693–1709.

[169] C. C. Mac Duffee, *The Theory of Matrices*, Chelsea, New York, 1946.

[170] A. Massoudi, M. Opmeer, and T. Reis, *The ADI method for algebraic Riccati equations*, Preprint 2014-16, Hamburger Beiträge zur Angewandten Mathematik, 2014.

[171] V. Mehrmann and T. Stykel, *Balanced Truncation Model Reduction for Large-Scale Systems in Descriptor Form*, in Dimension Reduction of Large-Scale Systems, P. Benner, V. Mehrmann, and D. Sorensen, eds., vol. 45 of Lecture Notes in Computational Science and Engineering, Springer-Verlag, Berlin/Heidelberg, Germany, 2005, pp. 83–115.

[172] V. Mehrmann and H. Xu, *Explicit Solutions for a Riccati Equation from Transport Theory*, SIAM J. Matrix Anal. Appl., 30 (2009), pp. 1339–1357.

[173] E. Mengi, *A Support Based Algorithm for Optimization with Eigenvalue Constraints*, tech. rep., Oct. 2013.

[174] E. Mengi, E. A. Yildirim, and M. Kiliç, *Numerical Optimization of Eigenvalues of Hermitian Matrix Functions*, SIAM J. Matrix Anal. Appl., 35 (2014), pp. 699–724.

[175] B. C. Moore, *Principal component analysis in linear systems: controllability, observability, and model reduction*, IEEE Trans. Automat. Control, AC-26 (1981), pp. 17–32.

[176] I. Moret and P. Novati, *RD-Rational Approximations of the Matrix Exponential*, BIT, 44 (2004), pp. 595–615.

[177] J. Nocedal and S. J. Wright, *Numerical optimization.*, Springer-Verlag, 1999.

[178] C. Nowakowski, P. Kürschner, P. Eberhard, and P. Benner, *Model reduction of an elastic crankshaft for elastic multibody simulations*, ZAMM - J. Appl. Math. Mech., 93 (2013), pp. 198–216.

[179] D. P. O'Leary, *The block conjugate gradient algorithm and related methods*, Linear Algebra Appl., 29 (1980), pp. 293–322.

[180] M. L. Overton, *Large-Scale Optimization of Eigenvalues*, SIAM J. Optimiz., 2 (1992), pp. 88–120.

[181] D. Peaceman and H. Rachford, *The numerical solution of elliptic and parabolic differential equations*, J. Soc. Indust. Appl. Math., 3 (1955), pp. 28–41.

[182] T. Penzl, *Numerische Lösung großer Lyapunov-Gleichungen*, Logos–Verlag, Berlin, Germany, 1998. Dissertation, Fakultät für Mathematik, TU Chemnitz, 1998.

[183] ——, *A cyclic low rank Smith method for large sparse Lyapunov equations*, SIAM J. Sci. Comput., 21 (2000), pp. 1401–1418.

[184] ——, *Eigenvalue decay bounds for solutions of Lyapunov equations: the symmetric case*, Sys. Control Lett., 40 (2000), pp. 139–144.

[185] ——, Lyapack *Users Guide*, Tech. Rep. SFB393/00-33, Sonderforschungsbereich 393 *Numerische Simulation auf massiv parallelen Rechnern*, TU Chemnitz, 09107 Chemnitz, Germany, 2000. Available from `http://www.tu-chemnitz.de/sfb393/sfb00pr.html`.

[186] D. Petterson, *A Nonlinear Optimization Approach to H2-Optimal Modeling and Control*, PhD thesis, Linköping University, 2013. Available from `http://www.diva-portal.org/smash/get/diva2:647068/FULLTEXT01.pdf`.

[187] V. Ramaswami, *Matrix analytic methods for stochastic fluid flows*, in Teletraffic Engineering in a Competitive World, D. Smith and P. Key, eds., Proc. of the 16th International Teletraffic Congress, Edinburgh, UK, 1999, Elsevier Science, pp. 1019–1030.

[188] L. Reichel and Q. Ye, *A generalized LSQR algorithm*, Numerical Linear Algebra with Applications, 15 (2008), pp. 643–660.

[189] T. Reis and T. Stykel., *Balanced truncation model reduction of second-order systems*, Math. Comput. Model. Dyn. Sys., 14 (2008), pp. 391–406.

[190] H. Richter, *Zum Logarithmus einer Matrix*, Archiv der Mathematik, 2 (1949), pp. 360–363.

[191] J. Roberts, *Linear Model Reduction and Solution of the Algebraic Riccati Equation by Use of the Sign Function*, Internat. J. Control, 32 (1980), pp. 677–687. (Reprint of Technical Report No. TR-13, CUED/B-Control, Cambridge University, Engineering Department, 1971).

[192] J. Rommes, N. Martins, and F. Freitas, *Computing Rightmost Eigenvalues for Small-Signal Stability Assessment of Large-Scale Power Systems*, IEEE Trans. Power Systems, 25 (2010), pp. 929–938.

[193] A. Ruhe, *Rational Krylov Sequence Methods for Eigenvalue Computation*, Linear Algebra Appl., 58 (1984), pp. 391–405.

[194] ——, *Rational Krylov algorithms for nonsymmetric Eigenvalue problems, II: Matrix pairs*, Linear Algebra Appl., 197/198 (1994), pp. 283–296.

[195] ——, *The Rational Krylov algorithm for nonsymmetric Eigenvalue problems. III: Complex shifts for real matrices*, BIT, 34 (1994), pp. 165–176.

[196] Y. Saad, *Numerical Solution of Large Lyapunov Equation*, in Signal Processing, Scattering, Operator Theory and Numerical Methods, M. A. Kaashoek, J. H. van Schuppen, and A. C. M. Ran, eds., Birkhäuser, 1990, pp. 503–511.

[197] ——, *Analysis of Some Krylov Subspace Approximations to the Matrix Exponential Operator*, SIAM J. Numer. Anal., 29 (1992), pp. 209–228.

[198] ——, *Numerical Methods for Large Eigenvalue Problems*, Manchester University Press, Manchester, UK, 1992.

[199] ——, *Iterative Methods for Sparse Linear Systems*, SIAM, Philadelphia, PA, 2003.

[200] J. Saak, *Effiziente numerische Lösung eines Optimalsteuerungsproblems für die Abkühlung von Stahlprofilen*, Diplomarbeit, Fachbereich 3/Mathematik und Informatik, Universität Bremen, D-28334 Bremen, Sept. 2003.

[201] ——, *Efficient Numerical Solution of Large Scale Algebraic Matrix Equations in PDE Control and Model Order Reduction*, PhD thesis, TU Chemnitz, July 2009. Available from `http://nbn-resolving.de/urn:nbn:de:bsz:ch1-200901642`.

[202] J. Sabino, *Solution of Large-Scale Lyapunov Equations via the Block Modified Smith Method*, PhD thesis, Rice University, Houston, Texas, June 2007. Available from: `http://www.caam.rice.edu/tech_reports/2006/TR06-08.pdf`.

[203] M. Sadkane, *A low-rank Krylov squared Smith method for large-scale discrete-time Lyapunov equations*, Linear Algebra and its Applications, 436 (2012), pp. 2807–282.

[204] M. A. Saunders, H. D. Simon, and E. L. Yip, *Two Conjugate-Gradient-Type Methods for Unsymmetric Linear Equations*, SIAM J. Numer. Anal., 25 (1988), pp. pp. 927–940.

[205] W. H. A. Schilders, H. A. van der Vorst, and J. Rommes, *Model Order Reduction: Theory, Research Aspects and Applications*, Springer-Verlag, Berlin, Heidelberg, 2008.

[206] H. Shaker and M. Tahavori, *Frequency-interval model reduction of bilinear systems*, IEEE Trans. Automat. Control, 59 (2014), pp. 1948–1953.

[207] D. Silvester, H. Elman, and A. Ramage, *Incompressible Flow and Iterative Solver Software (IFISS) version 3.2*, May 2012.

[208] V. Simoncini, *Restarted Full Orthogonalization Method for Shifted Linear Systems*, BIT, 43 (2003), pp. 459–466.

[209] ——, *A new iterative method for solving large-scale Lyapunov matrix equations*, SIAM J. Sci. Comput., 29 (2007), pp. 1268–1288.

[210] ——, *Computational methods for linear matrix equations*. Available at `http://www.dm.unibo.it/~simoncin/`, March 2013.

[211] V. Simoncini, D. B. Szyld, and M. Monsalve, *On two numerical methods for the solution of large-scale algebraic Riccati equations*, IMA J. Numer. Anal., 34 (2014), pp. 904–920.

[212] R. A. Smith, *Matrix Equation $XA + BX = C$*, SIAM J. Appl. Math., 16 (1968), pp. 198–201.

[213] D. Sorensen and Y. Zhou, *Bounds on Eigenvalue Decay Rates and Sensitivity of Solutions to Lyapunov Equations*, Tech. Rep. TR02-07, Dept. of Comp. Appl. Math., Rice University, Houston, TX, June 2002. Available online from `http://www.caam.rice.edu/caam/trs/tr02.html`.

[214] ——, *Direct methods for matrix Sylvester and Lyapunov equations*, J. Appl. Math, 2003 (2003), pp. 277–303.

[215] D. C. Sorensen and A. C. Antoulas, *The Sylvester equation and approximate balanced reduction*, Linear Algebra and its Applications, 351–352 (2002), pp. 671–700.

[216] G. Starke, *Optimal Alternating Directions Implicit Parameters for nonsymmetric systems of linear equations*, SIAM J. Numer. Anal., 28 (1991), pp. 1431–1445.

[217] T. Stykel, *Analysis and Numerical Solution of Generalized Lyapunov Equations*, dissertation, TU Berlin, 2002.

[218] T. Stykel and V. Simoncini, *Krylov subspace methods for projected Lyapunov equations*, Appl. Numer. Math., 62 (2012), pp. 35–50.

[219] K. Sun, *Model order reduction and domain decomposition for large-scale dynamical systems*, PhD thesis, Rice University, Houston, 2008. Available from `http://search.proquest.com/docview/304507831`.

[220] F. Tisseur and K. Meerbergen, *The Quadratic Eigenvalue Problem*, SIAM Rev., 43 (2001), pp. 235–286.

[221] M. S. Tombs and I. Postlethwaite, *Truncated balanced realization of a stable nonminimal state-space system*, Internat. J. Control, 46 (1987), pp. 1319–1330.

[222] N. Truhar and K. Veselić, *Bounds on the trace of a solution to the Lyapunov equation with a general stable matrix*, Sys. Control Lett., 56 (2007), pp. 493–503.

[223] ——, *An efficient method for estimating the optimal dampers' viscosity for linear vibrating systems using Lyapunov equation*, SIAM J. Matrix Anal. Appl., 31 (2009), pp. 18–39.

[224] J. van den Eshof and M. Hochbruck, *Preconditioning Lanczos Approximations to the Matrix Exponential*, SIAM J. Sci. Comput., 27 (2005), pp. 1438–1457.

[225] H. A. Van der Vorst, *Iterative Krylov Methods for Large Linear Systems*, Cambridge University Press, Cambridge, 2003.

[226] M. B. van Gijzen, G. L. G. Sleijpen, and J.-P. M. Zemke, *Flexible and multi-shift induced dimension reduction algorithms for solving large sparse linear systems*, Numer. Lin. Alg. Appl., 22 (2015), pp. 1–25.

[227] M. B. van Gijzen, C. B. Vreugdenhil, and H. Oksuzoglu, *The Finite Element Discretization for Stream-Function Problems on Multiply Connected Domains*, Journal of Computational Physics, 140 (1998), pp. 30–46.

[228] B. Vandereycken and S. Vandewalle, *A Riemannian Optimization Approach for Computing Low-Rank Solutions of Lyapunov Equations*, SIAM J. Matrix Anal. Appl., 31 (2010), pp. 2553–2579.

[229] E. L. Wachspress, *Iterative solution of the Lyapunov matrix equation*, Appl. Math. Lett., 107 (1988), pp. 87–90.

[230] ——, *Optimum parameters for two-variable ADI iteration*, Ann. Nuc. Ener., 19 (1992), pp. 765–778.

[231] ——, *The ADI Model Problem*, 1995. Available from the author.

[232] ——, *ADI Iteration Parameters for the Sylvester Equation*, 2000. Available from the author.

[233] ——, *The ADI Model Problem*, Springer New York, 2013.

[234] T. Wolf, $\mathcal{H}_2$ *Pseudo-Optimal Model Order Reduction*, PhD thesis, Technische Universität München, 2015.

[235] T. Wolf and H. K.-F. Panzer, *The ADI iteration for Lyapunov equations implicitly performs H2 pseudo-optimal model order reduction*, Internat. J. Control, 89 (2016), pp. 481–493.

[236] T. Wolf, H. K.-F. Panzer, and B. Lohmann, *ADI iteration for Lyapunov equations: a tangential approach and adaptive shift selection*, arXiv e-prints 1312.1142v1, Cornell University, Dec. 2013. math.NA.

[237] ——, *Model Order Reduction by Approximate Balanced Truncation: A Unifying Framework*, at-Automatica, 61 (2013), pp. 545–556.

[238] N. Wong and V. Balakrishnan, *Quadratic Alternating Direction Implicit Iteration for the Fast Solution of Algebraic Riccati Equations*, in Proc. Int. Symposium on Intelligent Signal Processing and Communication Systems, 2005, pp. 373–376.

[239] B. Yan, S. X.-D. Tan, and B. McGaughy, *Second-Order Balanced Truncation for Passive-Order Reduction of RLCK Circuits*, 55 (2008), pp. 942–946.

[240] K. Yosida, *Fourier Transform and Differential Equations*, in Functional Analysis, vol. 123 of Classics in Mathematics, Springer Berlin Heidelberg, 1995, pp. 145–193.

[241] B. Yu, D.-H. Li, and N. Dong, *Low memory and low complexity iterative schemes for a nonsymmetric algebraic Riccati equation arising from transport theory*, J. Comput. Appl. Math., 250 (2013), pp. 175–189.

[242] Y. Zhou, *Numerical Methods for Large Scale Matrix Equations with Applications in LTI System Model Reduction*, PhD thesis, Rice University, Houston, Texas, May 2002.

[243] Y. Zhou and D. C. Sorensen, *Approximate implicit subspace iteration with alternating directions for LTI system model reduction*, Numer. Lin. Alg. Appl., 15 (2008), pp. 873–886.

EHRENERKLÄRUNG

Ich versichere hiermit, dass ich die vorliegende Arbeit ohne unzulässige Hilfe Dritter und ohne Benutzung anderer als der angegebenen Hilfsmittel angefertigt habe; verwendete fremde und eigene Quellen sind als solche kenntlich gemacht. Ich habe insbesondere nicht wissentlich:

- Ergebnisse erfunden oder widersprüchliche Ergebnisse verschwiegen,
- statistische Verfahren absichtlich missbraucht, um Daten in ungerechtfertigter Weise zu interpretieren,
- fremde Ergebnisse oder Veröffentlichungen plagiiert oder verzerrt wiedergegeben.

Mir ist bekannt, dass Verstöße gegen das Urheberrecht Unterlassungs- und Schadenersatzansprüche des Urhebers sowie eine strafrechtliche Ahndung durch die Strafverfolgungsbehörden begründen kann.

Die Arbeit wurde bisher weder im Inland noch im Ausland in gleicher oder ähnlicher Form als Dissertation eingereicht und ist als Ganzes auch noch nicht veröffentlicht.

Magdeburg, 15.10.2015

Patrick Kürschner