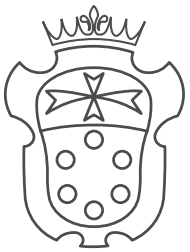


SCUOLA
NORMALE
SUPERIORE

Classe di Scienze

Corso di perfezionamento in
Metodi Computazionali e Modelli Matematici per le Scienze e la Finanza

XXXVI ciclo

Advances in polynomial and rational Krylov methods for matrix functions with applications

Settore Scientifico Disciplinare **MAT/08**

Candidato
dr. Igor Simunec

Supervisore
Prof. Michele Benzi

Co-supervisore
Prof. Leonardo Robol

Anno accademico 2023–2024

Abstract

In this thesis we explore the use of polynomial and rational Krylov subspace methods for the computation of the action of a matrix function $f(A)$ on a vector $\boldsymbol{b}$, a crucial linear algebra task in numerous applications, focusing on both theoretical analysis and algorithmic aspects.

We establish a priori and a posteriori bounds for the error in the approximation of $f(A)\boldsymbol{b}$ and $\boldsymbol{b}^T f(A)\boldsymbol{b}$ with polynomial and rational Krylov subspace methods, by exploiting integral representations of the function f. We also consider the use of rational Krylov methods for approximating the action of a generalized matrix function on a vector, extending previous approaches based on the Golub-Kahan bidiagonalization and providing a convergence analysis for this approach, which directly generalizes the established theory of rational Krylov methods for standard matrix functions.

We then develop a low-memory method for the computation of $f(A)\boldsymbol{b}$ for a Hermitian matrix A, by combining an outer Lanczos method with an inner rational Krylov subspace, which is used to compress the outer basis. The inner subspace is generated using small projected matrices, and it is thus inexpensive to construct compared to the cost of the outer iteration.

Next, we employ a rational Krylov method to solve a fractional diffusion equation on a graph, whose solution is given as $f(A)\boldsymbol{b}$, where A is the Laplacian of the graph and f is a function with a branch cut on $(-\infty, 0]$. We introduce techniques to cheaply remove the eigenvalue 0 of the graph Laplacian, which causes delays in the convergence of Krylov subspace methods.

We then consider two applications of Krylov subspace methods in the context of the approximation of $\mathrm{tr}(f(A))$ with stochastic trace estimators or probing methods. We first address the problem of approximating the von Neumann entropy of a symmetric positive semidefinite matrix A, defined as $S(A) = \mathrm{tr}(-A \log A)$, and we analyze in detail the convergence of the probing method, providing both theoretical error bounds and heuristic estimates. Next, we analyze the problem of identifying gaps in the spectrum of a symmetric matrix A, by computing traces of spectral projectors with a combination of a stochastic trace estimator and the Lanczos method. We give a rigorous analysis of the proposed algorithm to maximize its efficiency and provide guarantees on its results.

Acknowledgements

I am thankful to my supervisor Prof. Michele Benzi for his helpful guidance and advice throughout my PhD, and to Prof. Leonardo Robol for co-supervising my thesis. I am grateful to Prof. Stefan Güttel for his hospitality during a fruitful research visit to the University of Manchester.

I would also like to extend my thanks to the whole numerical linear algebra group at the Scuola Normale Superiore and the University of Pisa, and especially to my PhD student colleagues Angelo and Michele.

Finally, I thank all my friends for making my years in Pisa so enjoyable and memorable, and my parents for their constant support and encouragement.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Given a square matrix A and a function f defined on the spectrum of A, the computation of the matrix function $f(A)$ is a crucial problem in numerical linear algebra [96], which finds applications in several fields, ranging from engineering and physics to data science and machine learning. Notable examples of matrix functions include the matrix exponential $f(A) = \exp(A)$, the matrix logarithm $f(A) = \log(A)$, the matrix sign function $f(A) = \operatorname{sign}(A)$, and the matrix fractional powers $f(A) = A^\alpha$, with $\alpha \in (0, 1)$.

In many applications, it is not necessary to compute the whole matrix function $f(A)$, but rather its action on a given vector $\boldsymbol{b}$, i.e., the product $f(A)\boldsymbol{b}$ or the quadratic form $\boldsymbol{b}^T f(A)\boldsymbol{b}$. Direct methods for the computation of the whole matrix function $f(A)$, such as diagonalization, can be computationally prohibitive for large-scale problems. Using these methods may even be unfeasible in terms of storage costs when A is sparse, as the matrix function $f(A)$ is generally a dense matrix. As a result, there is an incentive to compute or approximate $f(A)\boldsymbol{b}$ or $\boldsymbol{b}^T f(A)\boldsymbol{b}$ directly without first forming $f(A)$ itself. Popular techniques for approximating these quantities are polynomial and rational Krylov subspace methods [59, 79, 90, 119]. These methods extract an approximation from a Krylov subspace, which is generated by computing matrix-vector products with A in the polynomial case or solving shifted linear systems with A in the rational case. The accuracy of approximations to $f(A)\boldsymbol{b}$ and $\boldsymbol{b}^T f(A)\boldsymbol{b}$ obtained through polynomial and rational Krylov subspaces is strongly linked to polynomial and, respectively, rational approximations of the scalar function f, so studying these approximations is crucial for understanding the convergence properties of Krylov subspace methods.

Polynomial and rational Krylov methods exhibit significantly different properties, and determining which one is the better approach heavily depends on the specific problem at hand. Each iteration of a polynomial Krylov method is relatively cheap, since it only involves a single matrix-vector product with A. However, the method may require many iterations to converge, especially when f has singularities near the spectrum of A, for instance if $f(z) = z^{-1/2}$ and A has eigenvalues close to 0. On the other hand, each iteration of a rational Krylov method requires the solution of a linear system with a shifted matrix $A - \xi I$, where ξ is one of the poles of the rational Krylov subspace. This operation can be significantly more expensive than a matrix-vector product, depending on the structure of A. Nevertheless, if the poles are selected carefully, a rational Krylov method can converge in a much smaller number of iterations, potentially also outperforming a polynomial Krylov method in terms of execution time. The structure of the matrix A is essential in determining the difference in cost between polynomial and rational Krylov iterations, while the properties of the function f and the location of the eigenvalues of A are critical factors for establishing whether a rational Krylov iteration can offer a substantial improvement in convergence speed over a polynomial iteration.

In this thesis we concentrate on analyzing polynomial and rational Krylov subspace methods for computing $f(A)\boldsymbol{b}$ from both a theoretical and algorithmic perspective, and we explore their use in conjunction with trace estimators for the approximation of $\operatorname{tr}(f(A))$ in specific applications. The thesis can be broadly divided into two main parts. In the first part, ranging from Chapter 2 to Chapter 6, we examine in detail various properties of polynomial and rational Krylov subspace methods, and we employ them to obtain new theoretical insights and algorithmic advancements aimed at improving the existing methods. In the second part, from Chapter 7 to Chapter 9, we combine Krylov subspace

methods with stochastic trace estimators [98,117] and probing techniques [77] to approximate $\mathrm{tr}(f(A))$, focusing on two specific problems: the computation of the von Neumann entropy and the detection of gaps in the spectrum of A. We provide an in-depth analysis tailored to these applications, which exploits the particular properties of the problems and leads to the development of efficient algorithms.

The rest of this thesis is structured as follows. In Chapter 2, we introduce some notation and we review basic concepts that will be used throughout the thesis. In particular, we present some integral expressions for functions, such as Cauchy-Stieltjes and Laplace-Stieltjes integral representations, which will be useful in the analysis of Krylov subspace methods. We then provide a few equivalent definitions of a matrix function and we introduce generalized matrix functions [94], defined using the singular value decomposition instead of the eigenvalue decomposition. Krylov subspace methods for the approximation of generalized matrix functions will be the topic of Chapter 4. We conclude Chapter 2 by defining polynomial and rational Krylov subspaces and the corresponding approximations to $f(A)\boldsymbol{b}$, and by discussing some of their fundamental properties which will serve as the building blocks for the rest of the thesis.

In Chapter 3 we establish bounds for the error in the approximation of $f(A)\boldsymbol{b}$ and $\boldsymbol{b}^T f(A)\boldsymbol{b}$ using a rational Krylov method, by following two distinct approaches. First, we review a posteriori error bounds for $f(A)\boldsymbol{b}$ from [89] obtained by using the Cauchy integral formula and the residue theorem, and we extend these bounds to quadratic forms $\boldsymbol{b}^T f(A)\boldsymbol{b}$. Then, we derive both a priori and a posteriori error bounds for $f(A)\boldsymbol{b}$ by exploiting an integral representation of the function f. This second approach generalizes the one used in [40] for the Lanczos method to the rational Krylov case, and involves overcoming some additional challenges that are not present in the polynomial case. Both techniques in this chapter rely on key properties of rational Krylov subspaces, such as the relations between rational Arnoldi decompositions and Krylov subspaces associated with shifted matrices, and the link between approximate solutions to shifted linear systems extracted from the same rational Krylov subspace. While these properties are well known for polynomial Krylov subspaces, not all of them are known in the rational context. Our analysis identifies some parallels between the polynomial and rational settings. Most of these results have been published in [143].

In Chapter 4 we introduce and analyze rational Krylov methods for approximating the action of a generalized matrix function on a vector. Previous literature has explored polynomial approximation methods, such as those based on the Golub-Kahan bidiagonalization [5] and on Chebyshev polynomial interpolation [6]. Here we extend the approach based on the Golub-Kahan bidiagonalization to the rational Krylov case, and we provide a convergence analysis of the resulting method that is a direct generalization of the established theory for rational Krylov methods applied to standard matrix functions. In addition, we obtain a short-term recurrence for updating the rational Krylov basis, by exploiting the quasiseparable structure of the projected matrix. We have published these results in [37].

In Chapter 5 we develop a new low-memory method for the computation of $f(A)\boldsymbol{b}$ for a Hermitian matrix A. This method combines an outer (polynomial or rational) Lanczos iteration with an inner rational Krylov subspace, which is used to compress the outer basis and reduce memory usage. The inner subspace is generated using a projected matrix of small size and its construction does not involve any operation with A, which makes it cheap compared to the cost of the outer iteration. In order to make the compression beneficial, the outer Krylov method should require many iteration to converge, and each iteration should be relatively cheap. Thus, suitable choices for the outer iteration are the polynomial Lanczos method, or alternatively the extended or shift-and-invert Krylov method, which use a small number of repeated poles. The resulting low-memory algorithm closely replicates the convergence of the outer Krylov subspace method, with an error that depends on the poles used in the inner rational Krylov subspace and is typically negligible. The method exhibits favorable performance when compared against several low-memory Krylov subspace methods from the literature. The contents of this chapter have appeared in the preprint [38].

In Chapter 6 we employ a rational Krylov method to solve a fractional diffusion equation on a graph, whose solution is expressed as $f(A)\boldsymbol{b}$, where $f(z) = \exp(-tz^\alpha)$, with $t > 0$ and $\alpha \in (0, 1)$. In this setting, the matrix A is the Laplacian associated with the graph, which is a singular M-matrix. Since the function f has a branch cut on $(-\infty, 0]$, convergence issues may arise with Krylov subspace methods. To deal with this challenge, we introduce techniques to remove the singularity of the graph Laplacian by utilizing the knowledge of the right and left eigenvectors of A associated with

the eigenvalue 0, thus improving the convergence rate of Krylov subspace methods for a negligible cost. This approach is applicable also for other problems where a known eigenvalue of A is close to a singularity of f; for instance, we apply the same technique in Chapter 8 for computing the von Neumann entropy of a network. The contents of this chapter have been published in [23].

Chapter 7 provides a brief introduction to numerical methods for estimating the trace of $f(A)$, which will be employed in the subsequent chapters alongside Krylov subspace methods. We begin by discussing stochastic trace estimators that approximate the trace using random sample vectors, such as Hutchinson's estimator [98], as well as more advanced variants such as Hutch++ [117, 129], that also integrate low-rank approximation. We also introduce probing methods [77], that exploit the sparsity structure of A to construct specific probing vectors which are then used to approximate the trace. Both approaches approximate the trace as a sum of a certain number of quadratic forms $\boldsymbol{x}_i^T f(A)\boldsymbol{x}_i$, which can be efficiently approximated using a Krylov subspace method.

In Chapter 8 we address the problem of approximating the von Neumann entropy [153] of a symmetric positive semidefinite matrix A, which is defined as $S(A) = \operatorname{tr}(-A \log A)$ and plays an important role in quantum mechanics and network science [11, 32, 105]. By exploiting an integral expression of the function $f(z) = z \log z$, we analyze the convergence of the probing method in this specific setting and we provide both theoretical bounds and heuristic estimates to predict its error. We also discuss implementation details for the Krylov subspace method used in the computation of quadratic forms, including the selection of poles and stopping criteria. The resulting probing algorithm is tested on several problems and compared with an adaptive implementation of Hutch++. Our experiments highlight advantages and disadvantages of both approaches, showing that each method performs better in different situations. These findings have been published in [21].

In Chapter 9 we employ Hutchinson's stochastic trace estimator in combination with the Lanczos algorithm to identify gaps in the spectrum of a real symmetric matrix A, by computing traces of spectral projectors associated with sections of the spectrum of A. This approach has been used in the literature for related tasks, such as estimating the number of eigenvalues of A within a given interval or approximating spectral densities [55,109]. A closely related problem is determining the gap between two consecutive eigenvalues of A, such as between $\lambda_k(A)$ and $\lambda_{k+1}(A)$, assuming that the eigenvalues $\lambda_j(A)$ are sorted in increasing order. Rather than focusing on finding the gap between two specific eigenvalues, we approach the problem from a broader perspective, and we aim to identify all gaps in the spectrum of A that are wider than a certain threshold. This shift in viewpoint allows for a rigorous theoretical analysis of the proposed algorithm, and enables us to determine the optimal parameters that maximize its efficiency. All the candidate gaps identified by the algorithm are certified to be actual gaps in the spectrum of A with high probability, and with an appropriate choice of the parameters we are able to detect all gaps with width above a certain threshold. We demonstrate the effectiveness of the resulting algorithm through several numerical experiments. The contents of this chapter will be included in a preprint that is currently in preparation [20].

Finally, Chapter 10 summarizes the results presented in this thesis, and provides concluding remarks and an outlook on potential directions for future research.

Chapter 2

Background

In this chapter we review some basic material concerning matrix functions and their computation by means of Krylov subspace methods. Throughout this work, we will frequently cite literature for definitions and theoretical results. These references are meant simply as a guide to where the information can be located, and they should not necessarily be interpreted as attributing the original results to the cited authors.

2.1 Notation and basic definitions

Let us begin by establishing some notation that we are going to use throughout this thesis and recalling some basic definitions. We use uppercase letters to denote matrices, and bold lowercase letters to denote vectors. Given a vector $\boldsymbol{v} \in \mathbb{C}^n$, we will sometimes denote its entries by v_j, for $j = 1, \ldots, n$. We will denote by I_n the $n \times n$ identity matrix, and by $\mathbf{1}_n$ and $\mathbf{0}_n$ vectors of all ones and all zeros of length n, respectively. We are going to omit the subscripts if they can be inferred without ambiguity. The vectors of the canonical basis of $\mathbb{C}^n$ or $\mathbb{R}^n$ will be denoted by $\boldsymbol{e}_j$, for $j = 1, \ldots, n$. We denote by $\|\boldsymbol{v}\|_2$ the Euclidean norm of a vector $\boldsymbol{v}$, and by $\|A\|_2$ the induced matrix norm, also called spectral norm. Recall that $\|A\|_2$ coincides with the largest singular value of A, and when A is Hermitian it coincides with the modulus of its largest eigenvalue.

Given a matrix $A \in \mathbb{C}^{n\times n}$, we denote its transpose by A^T, its conjugate transpose by A^H and its Moore-Penrose pseudoinverse by $A^\dagger$. We denote the eigenvalue spectrum of A by $\Lambda(A)$ and the set of its singular values by $\Sigma(A)$. The field of values (or numerical range) of A is given by

$$\mathbb{W}(A) := \{\boldsymbol{x}^H A \boldsymbol{x} \,:\, \|\boldsymbol{x}\|_2 = 1\}.$$

We recall that $\mathbb{W}(A)$ is a convex and compact subset of the complex plane that contains the convex hull of $\Lambda(A)$. If A is normal, that is, if $A^H A = A A^H$, then $\mathbb{W}(A)$ coincides with the convex hull of $\Lambda(A)$.

We denote by $\overline{\mathbb{C}}$ the extended complex plane $\mathbb{C} \cup \{\infty\}$, and similarly by $\overline{\mathbb{R}}$ the extended real line $\mathbb{R} \cup \{\infty\}$. Given a compact set $E \subset \mathbb{C}$ and a function $f : E \to \mathbb{C}$, we write $\|f\|_E$ to denote the supremum norm of f on E. The set of polynomials (with either real or complex coefficients depending on the context) with degree at most m is denoted by Π_m. For a given polynomial q, we denote by Π_m/q the set of rational functions with denominator q, that is,

$$\Pi_m/q = \{r = p/q \,:\, p \in \Pi_m\}.$$

For an invertible matrix $A \in \mathbb{C}^{n\times n}$ and two vectors $\boldsymbol{u}, \boldsymbol{v} \in \mathbb{C}^n$ such that $1 + \boldsymbol{v}^H A^{-1} \boldsymbol{u} \neq 0$, the Sherman-Morrison formula provides an expression for the inverse of the rank-one update $A + \boldsymbol{u}\boldsymbol{v}^H$, which is given by

$$(A + \boldsymbol{u}\boldsymbol{v}^H)^{-1} = A^{-1} - \frac{A^{-1}\boldsymbol{u}\boldsymbol{v}^H A^{-1}}{1 + \boldsymbol{v}^H A^{-1}\boldsymbol{u}}. \tag{2.1.1}$$

2.2 Functions with integral expressions

On multiple occasions, we will rely on an integral expression of a function f to derive certain theoretical results. In this section, we provide a brief introduction to the integral representations that will be employed in the following chapters.

We start by recalling that if the function f is analytic in a neighborhood of a point $a \in \mathbb{C}$, it can be written using the Cauchy integral formula as

$$f(a) = \frac{1}{2\pi i} \int_\Gamma \frac{f(z)}{z-a}\,\mathrm{d}z,$$

where Γ is a simple closed curve that winds counterclockwise around a, contained in the region where f is analytic.

The following definition introduces the class of Laplace-Stieltjes functions; see, for instance, [114, Definition 1].

Definition 2.2.1. A function $f : (0,\infty) \to \mathbb{R}$ is a Laplace-Stieltjes function if it can be expressed in the form

$$f(z) = \int_0^\infty e^{-tz} d\mu(t),$$

where μ is a positive measure on $[0,\infty)$.

The class of Laplace-Stieltjes functions coincides with the class of *completely monotonic functions* [135, Definition 1.3], i.e., infinitely differentiable functions defined on $(0,\infty)$ such that

$$(-1)^k f^{(k)}(z) \geq 0 \qquad \forall\, z > 0 \quad \text{and} \quad k \geq 0. \tag{2.2.1}$$

The equivalence between these two classes of functions is known as Bernstein's theorem [135, Theorem 1.4]. Examples of Laplace-Stieltjes functions are

$$e^{-z} = \int_1^\infty e^{-tz}\,\mathrm{d}t \qquad \text{and} \qquad z^{-\alpha} = \int_0^\infty e^{-tz} \frac{t^{\alpha-1}}{\Gamma(\alpha)}\,\mathrm{d}t, \quad \alpha \in (0,1).$$

A class of functions which is closely related to completely monotonic functions is the class of *Bernstein functions*, that consists of all functions $f : (0,\infty) \to \mathbb{R}$ of class C^∞ such that

$$f(z) \geq 0 \quad \text{and} \quad (-1)^{k-1} f^{(k)}(z) \geq 0, \qquad \forall\, k \geq 1 \quad \text{and} \quad \forall\, z > 0. \tag{2.2.2}$$

A nonnegative function $f : (0,\infty) \to \mathbb{R}$ is a Bernstein function if and only if its derivative f' is a completely monotonic function. The fractional power $f(z) = z^\alpha$, for $\alpha \in (0,1)$, is an example of a Bernstein function. Certain compositions of Bernstein functions and completely monotonic functions are completely monotonic [135, Theorem 3.7]. For instance, if f is a positive Bernstein function, then the function $g(z) = e^{-tf(z)}$ is completely monotonic for all $t > 0$. In particular, by considering the Bernstein function $f(z) = z^\alpha$, with $\alpha \in (0,1)$, we obtain that $g(z) = \exp(-tz^\alpha)$ is completely monotonic, or equivalently Laplace-Stieltjes. This fact will be useful in Chapter 6 for solving fractional diffusion equations on a graph.

We next introduce Cauchy-Stieltjes functions, which are a subclass of Laplace-Stieltjes functions; see, for instance, [24, Definition 3.1].

Definition 2.2.2. A function $f : \mathbb{C} \setminus (-\infty, 0] \to \mathbb{C}$ is a Cauchy-Stieltjes function if it can be expressed in the form

$$f(z) = \int_0^\infty \frac{1}{t+z} d\mu(t),$$

where μ is a positive measure on $[0,\infty)$.

Some examples of Cauchy-Stieltjes functions are

$$z^{-\alpha} = \frac{\sin(\alpha\pi)}{\pi} \int_0^\infty \frac{t^{-\alpha}}{t+z}\,\mathrm{d}t, \quad \alpha \in (0,1), \qquad \text{and} \qquad \frac{\log(1+z)}{z} = \int_1^\infty \frac{1}{t(t+z)}\,\mathrm{d}t.$$

Cauchy-Stieljes functions are also completely monotonic, so they are a subset of Laplace-Stieltjes functions. This can be also seen directly, by observing that

$$\int_0^\infty \frac{1}{t+z}\,\mathrm{d}\mu(t) = \int_0^\infty \int_0^\infty e^{-s(t+z)}\,\mathrm{d}s\,\mathrm{d}\mu(t) = \int_0^\infty e^{-sz}\left(\int_0^\infty e^{-st}\,\mathrm{d}\mu(t)\right)\mathrm{d}s,$$

where $\nu(s) := \int_0^\infty e^{-st}\,\mathrm{d}\mu(t)$ is the density of the measure associated with the Laplace-Stieltjes representation. Cauchy-Stieltjes functions are also known as Stieltjes functions [24].

Markov functions [10] are a slightly more general class of functions that are closely related to Cauchy-Stieltjes functions.

Definition 2.2.3. Given $-\infty \leq \alpha < \beta < \infty$, a function $f : \mathbb{C} \setminus [\alpha, \beta] \to \mathbb{C}$ is a Markov function if it can be expressed in the form

$$f(z) = \int_\alpha^\beta \frac{1}{z-t} d\mu(t),$$

where μ is a positive measure with support contained in $[\alpha, \beta]$.

A Markov function is analytic in $\mathbb{C} \setminus [\alpha, \beta]$. A Cauchy-Stieltjes function is also a Markov function with $[\alpha, \beta] = (-\infty, 0]$.

We refer the reader to [135] for further detail on the classes of functions introduced in this section.

2.3 Matrix functions

This section introduces the definition of a matrix function and reviews some basic properties of matrix functions that will be useful throughout this thesis. The contents of this section are mainly based on [96, Chapter 1].

Given a square matrix $A \in \mathbb{C}^{n\times n}$ and a scalar function f, there are three classical ways to define a matrix function $f(A) \in \mathbb{C}^{n\times n}$. The first way to define $f(A)$ is through its Jordan canonical form. Recall that any matrix A can be expressed as

$$A = ZJZ^{-1}, \qquad J = \operatorname{diag}(J_1, \ldots, J_p),$$

where Z is nonsingular and J is the Jordan canonical form of A, which is a block diagonal matrix with blocks

$$J_k = J_k(\lambda_k) = \begin{bmatrix} \lambda_k & 1 & & \\ & \lambda_k & \ddots & \\ & & \ddots & 1 \\ & & & \lambda_k \end{bmatrix} \in \mathbb{C}^{m_k \times m_k},$$

with $m_1 + \cdots + m_p = n$. The matrix $J_k(\lambda_k)$ is a Jordan block associated with the eigenvalue λ_k. The index $n(\lambda)$ of an eigenvalue $\lambda \in \Lambda(A)$ is defined as the largest size of a Jordan block associated with λ. Recall that a matrix is diagonalizable if and only if all of its eigenvalues have index 1. We say that a function is defined on the spectrum of A if all the values

$$f^{(j)}(\lambda), \qquad j = 0, \ldots, n(\lambda) - 1, \quad \lambda \in \Lambda(A)$$

exist [96, Definition 1.1]. When the matrix A is diagonalizable, this definition simply requires that the values $f(\lambda)$ are defined for all $\lambda \in \Lambda(A)$. When f is defined on the spectrum of A according to the definition above, we can define $f(A)$ via the Jordan canonical form of A as follows.

Definition 2.3.1 ([96, Definition 1.2])**.** Let $A \in \mathbb{C}^{n\times n}$ have a Jordan canonical form $A = ZJZ^{-1}$ and let f be defined on the spectrum of A. Then

$$f(A) := Zf(J)Z^{-1} = Z\operatorname{diag}(f(J_k(\lambda_k)))Z^{-1},$$

where

$$f(J_k(\lambda_k)) := \begin{bmatrix} f(\lambda_k) & f'(\lambda_k) & \cdots & \frac{f^{(m_k-1)}(\lambda_k)}{(m_k-1)!} \\ & f(\lambda_k) & \ddots & \vdots \\ & & \ddots & f'(\lambda_k) \\ & & & f(\lambda_k) \end{bmatrix} \in \mathbb{C}^{m_k \times m_k}.$$

When A is diagonalizable, the Jordan canonical form simplifies to the eigenvalue decomposition $A = Z\Lambda Z^{-1}$, with $\Lambda = \operatorname{diag}(\lambda_1, \ldots, \lambda_n)$, and Definition 2.3.1 reduces to

$$f(A) = Zf(\Lambda)Z^{-1}, \qquad f(\Lambda) = \operatorname{diag}(f(\lambda_1), \ldots, f(\lambda_n)).$$

A matrix function can be alternatively defined by using polynomial interpolation.

Definition 2.3.2 ([96, Definition 1.4])**.** Let f be defined on the spectrum of A. Then $f(A) := p(A)$, where p is the unique polynomial of smallest degree that satisfies the Hermite interpolation conditions

$$p^{(j)}(\lambda) = f^{(j)}(\lambda), \qquad j = 1, \ldots, n(\lambda) - 1, \quad \lambda \in \Lambda(A).$$

The polynomial p is called the Hermite interpolating polynomial.

The above definition is equivalent to Definition 2.3.1, following from the fact that for any polynomial p the matrix $p(A)$ is completely determined by the values of p on the spectrum of A. In particular, this implies that if two polynomials p and q take the same values on the spectrum of A, then $p(A) = q(A)$, and hence in Definition 2.3.2 we can equivalently take as p any polynomial that satisfies the Hermite interpolation conditions, not necessarily the one of smallest degree.

An elegant definition of a matrix function can be given via a generalization of the Cauchy integral formula to the matrix case. Unlike the previous definitions, this one requires f to be analytic in a neighbourhood of $\Lambda(A)$, but it is frequently useful for proving certain theoretical results.

Definition 2.3.3 ([96, Definition 1.11])**.** Let f be analytic on and inside a closed contour Γ that winds around $\Lambda(A)$ exactly once. Then

$$f(A) := \frac{1}{2\pi i} \int_\Gamma f(z)(zI - A)^{-1} \, \mathrm{d}z.$$

This definition is equivalent to Definitions 2.3.1 and 2.3.2 when f is analytic in an open set that contains $\Lambda(A)$; see [96, Theorem 1.12] and [97, Theorem 6.2.28].

The matrix function $f(A)$ can be defined similarly to Definition 2.3.3 also for other integral representations of f, such as in the case of Cauchy-Stieltjes functions. Given a Cauchy-Stieltjes function f with integral representation

$$f(z) = \int_0^\infty \frac{1}{t+z} d\mu(t), \qquad z \in \mathbb{C} \setminus (-\infty, 0],$$

assuming that A has no eigenvalues in the set $(-\infty, 0]$, we have

$$f(A) = \int_0^\infty (A + tI)^{-1} \, \mathrm{d}\mu(t). \tag{2.3.1}$$

This identity can be proved by first showing that it holds for Jordan blocks, which can be done with a direct comparison of the entries of the right- and left-hand side using an integral representation of the derivatives of a Cauchy-Stieltjes function (see for example [24, Section 3]).

A significant result for the theory of matrix functions is Crouzeix's theorem [47], which links the spectral norm of $f(A)$ with the uniform norm of f over the numerical range of the matrix A.

Theorem 2.3.4. Assume that f is analytic in a neighbourhood of $\mathbb{W}(A)$. Then

$$\|f(A)\|_2 \leq C \|f\|_{\mathbb{W}(A)}, \qquad C \leq 1 + \sqrt{2},$$

This result was originally proved in [47] with a constant $C = 11.08$, which was later improved in [49] to $C = 1 + \sqrt{2}$. If A is normal, we have $\|f(A)\|_2 \leq \|f\|_{\mathbb{W}(A)}$, recalling that in this case $\mathbb{W}(A)$ coincides with the convex hull of $\Lambda(A)$. A conjecture by Crouzeix states that the inequality $\|f(A)\|_2 \leq C\|f\|_{\mathbb{W}(A)}$ holds with $C = 2$ for any matrix A.

2.4 Generalized matrix functions

Generalized matrix functions (GMFs) are an extension of the notion of matrix functions based on the singular value decomposition (SVD) instead of the spectral decomposition. They were introduced for the first time in [94], with the purpose of extending the definition of matrix functions to rectangular matrices.

Let $A \in \mathbb{C}^{m\times n}$ be a possibly rectangular matrix, and let $A = U\Sigma V^H$ be its SVD, where $U \in \mathbb{C}^{m\times m}$ and $V \in \mathbb{C}^{n\times n}$ are unitary and $\Sigma \in \mathbb{R}^{m\times n}$ is defined as

$$\Sigma_{i,j} = \begin{cases} \sigma_i & \text{if } i = j \leq r \\ 0 & \text{otherwise,} \end{cases}$$

where $r \leq \min\{m, n\}$ is the rank of A and $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_r > 0$ are the nonzero singular values of A. A generalized matrix function of A is defined as follows.

Definition 2.4.1. Given a function f defined on the set $\{\sigma_1, \ldots, \sigma_r\}$, the generalized matrix function of f applied on A is denoted by $f^\diamond(A)$ and is defined as

$$f^\diamond(A) := U f^\diamond(\Sigma) V^H,$$

where

$$f^\diamond(\Sigma)_{i,j} := \begin{cases} f(\sigma_i) & \text{if } i = j \leq r \\ 0 & \text{otherwise.} \end{cases}$$

Observe that a GMF can be expressed in terms of the compact SVD of the matrix A, that is $A = U_r \Sigma_r V_r^H$, where $U_r \in \mathbb{C}^{m\times r}$ and $V \in \mathbb{C}^{n\times r}$ have orthonormal columns and $\Sigma_r = \operatorname{diag}(\sigma_1, \ldots, \sigma_r) \in \mathbb{R}^{r\times r}$. In this case, we have

$$f^\diamond(A) = U_r f(\Sigma_r) V_r^H.$$

Since the definition of a GMF only depends on the values of f on the (positive) nonzero singular values of A, we can always assume that f is an odd function, and in particular that $f(0) = 0$.

Generalized matrix functions have become an active area of research only in recent years. For instance, theoretical aspects of GMFs have been investigated in [3, 18, 125], while efficient numerical methods have been developed in [5,6]. For some applications of generalized matrix functions, we direct the reader to [3,5,6] and the references therein. GMFs have also been recently considered in the context of quantum algorithms [107]. Generalized matrix functions are going to be the focus of Chapter 4, in which we analyze the use of rational Krylov subspace methods for the approximation of $f^\diamond(A)\boldsymbol{b}$.

2.5 Krylov subspace methods

The computation of the matrix function $f(A)$ is usually a computationally demanding problem, since $f(A)$ is in general a dense matrix even when the starting matrix A is sparse, and hence most algorithms require $\mathcal{O}(n^3)$ arithmetic operations [96]. For very large and sparse A, the computation of $f(A)$ is extremely expensive and potentially even unfeasible due to storage limitations. On the other hand, in many situations of practical interest it is enough to compute the action of $f(A)$ on a given vector $\boldsymbol{b} \in \mathbb{C}^n$, i.e., the product $f(A)\boldsymbol{b}$. In this case, it is possible to compute or approximate $f(A)\boldsymbol{b}$ without first computing $f(A)$, which can lead to significant savings in terms of both computational and storage costs. In this section we introduce Krylov subspace methods, which are a fundamental tool for computing approximations to $f(A)\boldsymbol{b}$.

2.5.1 Polynomial Krylov subspaces

We start by introducing polynomial Krylov subspaces, which are widely used for computing approximate solutions to linear algebra problems, from linear systems and eigenvalue problems to matrix functions and matrix equations.

Algorithm 2.1 Arnoldi algorithm

Input: $A \in \mathbb{C}^{n\times n}, \boldsymbol{b} \in \mathbb{C}^n$, $m \in \mathbb{N}$.

Output: $V_{m+1} \in \mathbb{C}^{n\times(m+1)}$ orthonormal basis of $\mathcal{K}_{m+1}(A, \boldsymbol{b})$, $\underline{H}_m \in \mathbb{C}^{(m+1)\times m}$ upper Hessenberg, satisfying the Arnoldi relation (2.5.1).

1: $\boldsymbol{v}_1 = \boldsymbol{b}/\|\boldsymbol{b}\|_2$
2: **for** $j = 1, \dots, m$ **do**
3: $\quad \boldsymbol{w}_j = A\boldsymbol{v}_j$
4: $\quad$ **for** $i = 1, \dots, j$ **do**
5: $\quad\quad h_{i,j} = \boldsymbol{v}_i^H \boldsymbol{w}_j$
6: $\quad\quad \boldsymbol{w}_j = \boldsymbol{w}_j - h_{i,j}\boldsymbol{v}_i$
7: $\quad$ **end for**
8: $\quad h_{j+1,j} = \|\boldsymbol{w}_j\|_2$
9: $\quad$ **if** $h_{j+1,j} = 0$ **then**
10: $\quad\quad$ **stop**
11: $\quad$ **end if**
12: $\quad \boldsymbol{v}_{j+1} = \boldsymbol{w}_j/h_{j+1,j}$
13: **end for**

Definition 2.5.1. Given a matrix $A \in \mathbb{C}^{n\times n}$ and a vector $\boldsymbol{b} \in \mathbb{C}^n$, the m-th polynomial Krylov subspace associated with A and $\boldsymbol{b}$ is defined as

$$\mathcal{K}_m(A, \boldsymbol{b}) := \operatorname{span}\{\boldsymbol{b}, A\boldsymbol{b}, \dots, A^{m-1}\boldsymbol{b}\} = \{p(A)\boldsymbol{b} \,:\, p \in \Pi_{m-1}\}.$$

The subspaces $\mathcal{K}_j(A, \boldsymbol{b})$, $j = 1, \dots, m$ form a nested sequence of strictly increasing dimension, provided that m is smaller than the invariance index M of the Krylov subspace, i.e., the smallest integer such that $\mathcal{K}_M(A, \boldsymbol{b}) = \mathcal{K}_{M+1}(A, \boldsymbol{b})$. The invariance index M coincides with the degree of the minimal polynomial of $\boldsymbol{b}$ with respect to A, that is, the monic polynomial ψ of smallest degree such that $\psi(A)\boldsymbol{b} = 0$, see for example [134, Proposition 6.1]. For simplicity, throughout this thesis we are going to assume that $m \leq M$ so that we always have $\dim \mathcal{K}_m(A, \boldsymbol{b}) = m$.

An orthonormal basis $V_m = [\boldsymbol{v}_1 \,\dots\, \boldsymbol{v}_m]$ of $\mathcal{K}_m(A, \boldsymbol{b})$ can be computed using the Arnoldi algorithm [134, Algorithm 6.1], which begins by setting $\boldsymbol{v}_1 = \boldsymbol{b}/\|\boldsymbol{b}\|_2$ and iteratively constructs the other basis vectors by multiplying the last computed basis vector by A and orthogonalizing against all previous ones. This procedure is described in Algorithm 2.1. In addition to the basis V_{m+1} of the Krylov subspace, the Arnoldi algorithm additionally computes an $(m+1) \times m$ upper Hessenberg matrix $\underline{H}_m$ that contains the coefficients used in the orthogonalization of the basis vectors, which together with V_{m+1} satisfies the Arnoldi relation

$$AV_m = V_{m+1}\underline{H}_m. \tag{2.5.1}$$

In the following, we are going to denote by H_m the $m \times m$ leading principal block of $\underline{H}_m$, which coincides with the orthogonal projection of A onto the Krylov subspace $\mathcal{K}_m(A, \boldsymbol{b})$, that is $H_m = V_m^H A V_m$. This follows immediately from (2.5.1).

The Arnoldi algorithm breaks down at step j (i.e., the condition at line 9 of Algorithm 2.1 is fulfilled) if and only if the invariance index of the Krylov subspace is j, see for instance [134, Proposition 6.6]. Such a breakdown occurs when we have found an invariant subspace of A, so it is usually called a lucky breakdown.

The polynomial Krylov subspace $\mathcal{K}_m(A, \boldsymbol{b})$ can be used to approximate $f(A)\boldsymbol{b}$ with the Arnoldi approximation

$$f(A)\boldsymbol{b} \approx \boldsymbol{f}_m := V_m f(H_m) V_m^H \boldsymbol{b} = \|\boldsymbol{b}\|_2 V_m f(H_m)\boldsymbol{e}_1. \tag{2.5.2}$$

The use of this approximation can be motivated by Definition 2.3.2. Indeed, it turns out that $\boldsymbol{f}_m = p_{m-1}(A)\boldsymbol{b}$, where p_{m-1} is the Hermite interpolation polynomial of the function f at the eigenvalues of H_m, also known as Ritz values associated with $\mathcal{K}_m(A, \boldsymbol{b})$ [96, Theorem 13.5]. Recalling that $f(A)$ coincides with $p(A)$, where p is the Hermite interpolation polynomial of f at the eigenvalues of A, approximating $f(A)\boldsymbol{b}$ with $\boldsymbol{f}_m$ can be interpreted as approximating the polynomial p with the lower-degree polynomial p_{m-1}.

The interpretation of (2.5.2) in terms of polynomial approximations of f is also useful to understand the magnitude of the error $f(A)\boldsymbol{b} - \boldsymbol{f}_m$. First of all, the approximation (2.5.2) is exact when f is a polynomial of degree at most $m-1$.

Lemma 2.5.2 ([96, Lemma 13.4])**.** Let V_m and H_m be as in (2.5.1). Then, for any polynomial p with $\deg p \leq m-1$ we have

$$p(A)\boldsymbol{b} = \|\boldsymbol{b}\|_2 V_m p(H_m)\boldsymbol{e}_1.$$

Using Lemma 2.5.2 one can prove the following theorem, which links the error in the Arnoldi appoximation of $f(A)\boldsymbol{b}$ with the error of the polynomial approximation of the scalar function f in the uniform norm.

Theorem 2.5.3 ([10, Proposition 3.1])**.** Let $\boldsymbol{f}_m$ be the Arnoldi approximation (2.5.2) to $f(A)$. Then, if f is analytic in a neighborhood of $\mathbb{W}(A)$, we have

$$\|f(A)\boldsymbol{b} - \boldsymbol{f}_m\|_2 \leq 2C\|\boldsymbol{b}\|_2 \min_{p\in\Pi_{m-1}} \|f-p\|_{\mathbb{W}(A)},$$

with $C = \sqrt{2}+1$. If A is normal, the same result holds with $C = 1$.

Proof. The main tool for the proof is the exactness property of (2.5.2) on polynomials of degree up to $m-1$, given in Lemma 2.5.2. For any polynomial p with $\deg p \leq m-1$, we have

$$f(A)\boldsymbol{b} - \|\boldsymbol{b}\|_2 V_m f(H_m)\boldsymbol{e}_1 = f(A)\boldsymbol{b} - p(A)\boldsymbol{b} + \|\boldsymbol{b}\|_2 V_m p(H_m)\boldsymbol{e}_1 - \|\boldsymbol{b}\|_2 V_m f(H_m)\boldsymbol{e}_1,$$

so we get

$$\|f(A)\boldsymbol{b} - \|\boldsymbol{b}\|_2 V_m f(H_m)\boldsymbol{e}_1\|_2 \leq \|\boldsymbol{b}\|_2 \big(\|f(A)-p(A)\|_2 + \|f(H_m)-p(H_m)\|_2\big).$$

If f is analytic on $\mathbb{W}(A)$, by Crouzeix's theorem we have

$$\|f(A)-p(A)\|_2 \leq C\|f-p\|_{\mathbb{W}(A)} \qquad \text{and} \qquad \|f(H_m)-p(H_m)\|_2 \leq C\|f-p\|_{\mathbb{W}(H_m)},$$

with $C = 1+\sqrt{2}$. Using also $\mathbb{W}(H_m) \subset \mathbb{W}(A)$, we conclude

$$\|f(A)\boldsymbol{b} - \|\boldsymbol{b}\|_2 V_m f(H_m)\boldsymbol{e}_1\|_2 \leq 2C\|\boldsymbol{b}\|_2 \|f-p\|_{\mathbb{W}(A)}.$$

The thesis is obtained by taking the minimum over all polynomials p with $\deg p \leq m-1$.

When A is normal, the same result follows with $C = 1$ by taking an eigenvalue decomposition $A = U\Lambda U^H$ and observing that for any function g defined on the spectrum of A we have

$$\|g(A)\|_2 = \|g(\Lambda)\|_2 = \max_{\lambda\in\Lambda(A)} |g(\lambda)|. \qquad \square$$

In the proof of Theorem 2.5.3 we used the property $\mathbb{W}(H_m) \subset \mathbb{W}(A)$. Note that when A is not normal, the eigenvalues of H_m are not necessarily contained in the convex hull of the spectrum of A. In particular, this implies that a function f that is defined on the spectrum of A but not on the whole numerical range $\mathbb{W}(A)$ may not be defined on the spectrum of H_m. This situation may happen, for instance, when computing the square root of a non-Hermitian matrix A with eigenvalues contained in the right-half plane. If some eigenvalues of A are close to the origin, it is likely that $\mathbb{W}(A)$ intersects $(-\infty, 0]$, where the square root has a branch cut.

Remark 2.5.4. If follows from the proof of Theorem 2.5.3 that when A is normal we also have the stronger bound

$$\|f(A)\boldsymbol{b} - \boldsymbol{f}_m\|_2 \leq \|\boldsymbol{b}\|_2 \min_{p\in\Pi_{m-1}} \big(\|f-p\|_{\Lambda(A)} + \|f-p\|_{\Lambda(H_m)}\big).$$

Depending on the distribution of the eigenvalues of A, this bound can be significantly more accurate than the one in Theorem 2.5.3, because it takes into account more detailed information on the spectrum such as the presence of isolated eigenvalues.

In the Hermitian case, the Arnoldi algorithm simplifies to the Lanczos algorithm [134, Algorithm 6.15], which can be used to construct an orthonormal basis V_m by means of a three-term recurrence. In each iteration of the Lanczos algorithm, the next basis vector $\boldsymbol{v}_{j+1}$ is obtained by orthogonalizing $A\boldsymbol{v}_j$ against the two previous basis vectors $\boldsymbol{v}_{j-1}$ and $\boldsymbol{v}_j$. The constructed basis is orthonormal in exact arithmetic, although there may be loss of orthogonality when finite precision is used, see for instance [88, Section 10.3] and the references therein. The three-term recurrence significantly reduces the computational cost of orthogonalization, and allows us to construct the basis V_m without the need to keep in memory all the previous basis vectors. This feature is exploited in several low-memory algorithms, such as the conjugate gradient algorithm for solving linear systems [134] and the two-pass Lanczos method for computing $f(A)\boldsymbol{b}$ [31]. We are going to focus in more detail on low-memory methods for the computation of $f(A)\boldsymbol{b}$ in Chapter 5, where we develop a memory efficient method by using an inner rational Krylov subspace to compress the Lanczos basis.

2.5.2 Rational Krylov subspaces

Similarly to how polynomial Krylov subspaces are related to polynomials in the matrix A, we can define a rational Krylov subspace in terms of rational functions in the matrix A.

Definition 2.5.5. Given a sequence of poles $\boldsymbol{\xi}_m = \{\xi_1, \dots, \xi_{m-1}\} \in \overline{\mathbb{C}} \setminus \Lambda(A)$, the rational Krylov subspace associated with A and $\boldsymbol{b}$ with poles $\boldsymbol{\xi}_m$ is defined as

$$\mathcal{Q}_m(A, \boldsymbol{b}, \boldsymbol{\xi}_m) = q_{m-1}(A)^{-1}\mathcal{K}_m(A, \boldsymbol{b}) = \{q_{m-1}(A)^{-1}p(A)\boldsymbol{b} \,:\, p \in \Pi_{m-1}\},$$

where

$$q_{m-1}(z) = \prod_{\substack{j=1 \\ \xi_j \neq \infty}}^{m-1} (z - \xi_j).$$

We are going to refer to q_{m-1} as the denominator polynomial associated with $\mathcal{Q}_m(A, \boldsymbol{b}, \boldsymbol{\xi}_m)$.

Note that Definition 2.5.5 also allows for infinite poles, and if all poles in $\boldsymbol{\xi}_m$ are equal to ∞, then we have $q_{m-1} \equiv 1$ and the rational Krylov subspace $\mathcal{Q}_m(A, \boldsymbol{b}, \boldsymbol{\xi}_m)$ reduces to the polynomial Krylov subspace $\mathcal{K}_m(A, \boldsymbol{b})$ as a special case. When the list of poles $\boldsymbol{\xi}_m$ is clear from the context, we will sometimes omit it from the definition of the rational Krylov subspace, and simply write $\mathcal{Q}_m(A, \boldsymbol{b}) := \mathcal{Q}_m(A, \boldsymbol{b}, \boldsymbol{\xi}_m)$ without explicitly referencing the poles.

Rational Krylov subspaces constructed using the same sequence of poles are nested as long as $m \leq M$, where M denotes the invariance index of $\mathcal{K}_m(A, \boldsymbol{b})$, that is, if $\boldsymbol{\xi}_j = \{\xi_1, \dots, \xi_{j-1}\}$ for $j \leq m$ we have

$$\operatorname{span}\{\boldsymbol{b}\} = \mathcal{Q}_1(A, \boldsymbol{b}, \boldsymbol{\xi}_1) \subset \mathcal{Q}_2(A, \boldsymbol{b}, \boldsymbol{\xi}_2) \subset \cdots \subset \mathcal{Q}_m(A, \boldsymbol{b}, \boldsymbol{\xi}_m), \tag{2.5.3}$$

see for instance [89, Lemma 4.3]. An orthonormal basis V_m of $\mathcal{Q}_m(A, \boldsymbol{b}, \boldsymbol{\xi}_m)$ can be constructed with the rational Arnoldi algorithm [89, Algorithm 1], introduced by Ruhe in [133] in the context of eigenvalue problems. The rational Arnoldi algorithm is described in Algorithm 2.2.

The matrices V_{m+1} and $\underline{H}_m$ computed by Algorithm 2.2 satisfy the rational Arnoldi relation

$$AV_{m+1}\underline{K}_m = V_{m+1}\underline{H}_m, \tag{2.5.4}$$

with the upper Hessenberg matrix

$$\underline{K}_m := \begin{bmatrix} 1 & & \\ & \ddots & \\ & & 1 \\ 0 & \cdots & 0 \end{bmatrix} + \underline{H}_m \begin{bmatrix} \xi_1^{-1} & & \\ & \ddots & \\ & & \xi_m^{-1} \end{bmatrix} \in \mathbb{C}^{(m+1)\times m}, \tag{2.5.5}$$

where we are using the convention $1/\infty = 0$. In particular, when all poles are infinite we have $\underline{K}_m = \begin{bmatrix} I_m \\ \boldsymbol{0}_m^T \end{bmatrix}$ and we recover the Arnoldi relation (2.5.1) for $\mathcal{K}_m(A, \boldsymbol{b})$. Similarly to the polynomial Krylov case,

Algorithm 2.2 Rational Arnoldi algorithm

Input: $A \in \mathbb{C}^{n\times n}, \boldsymbol{b} \in \mathbb{C}^n$, $m \in \mathbb{N}, \boldsymbol{\xi}_{m+1} = [\xi_1, \dots, \xi_m] \subset \overline{\mathbb{C}} \setminus \Lambda(A)$.
Output: $V_{m+1} \in \mathbb{C}^{n\times(m+1)}$ orthonormal basis of $\mathcal{Q}_{m+1}(A, \boldsymbol{b}, \boldsymbol{\xi}_{m+1})$, $\underline{H}_m \in \mathbb{C}^{(m+1)\times m}$ upper Hessenberg, satisfying the rational Arnoldi relation (2.5.4).

1: $\boldsymbol{v}_1 = \boldsymbol{b}/\|\boldsymbol{b}\|_2$
2: **for** $j = 1, \dots, m$ **do**
3: $\quad \boldsymbol{w}_j = (I - A/\xi_j)^{-1} A\boldsymbol{v}_j$ ▷ here we use the convention $A/\infty = 0$
4: $\quad$ **for** $i = 1, \dots, j$ **do**
5: $\quad\quad h_{i,j} = \boldsymbol{v}_i^H \boldsymbol{w}_j$
6: $\quad\quad \boldsymbol{w}_j = \boldsymbol{w}_j - h_{i,j}\boldsymbol{v}_i$
7: $\quad$ **end for**
8: $\quad h_{j+1,j} = \|\boldsymbol{w}_j\|_2$
9: $\quad$ **if** $h_{j+1,j} = 0$ **then**
10: $\quad\quad$ **stop**
11: $\quad$ **end if**
12: $\quad \boldsymbol{v}_{j+1} = \boldsymbol{w}_j / h_{j+1,j}$
13: **end for**

we are going to denote by H_m and K_m the $m \times m$ leading principal blocks of $\underline{H}_m$ and $\underline{K}_m$, respectively. The upper Hessenberg matrices $\underline{H}_m$ and $\underline{K}_m$ both have full rank m, see for instance [89, Lemma 5.6]. Further details on rational Arnoldi decompositions can be found in [26].

Remark 2.5.6. The rational Arnoldi implementation in Algorithm 2.2 does not allow for poles equal to zero. However, this is not a limitation in practice, because one can still construct V_m by choosing a nonzero scalar σ that does not coincide with any of the poles and running Algorithm 2.2 to generate a basis of $\mathcal{Q}_m(A - \sigma I, \boldsymbol{b}, [\xi_j - \sigma]_{j=1}^{m-1})$, which is the same subspace as $\mathcal{Q}_m(A, \boldsymbol{b}, \boldsymbol{\xi}_m)$ but does not have any poles at zero. We mention in addition that the $\boldsymbol{w}_j$ at line 3 of Algorithm 2.2 is not guaranteed to expand the rational Krylov subspace, i.e., it may in principle happen that $\boldsymbol{w}_j \in \mathcal{Q}_j(A, \boldsymbol{b})$ and hence the algorithm would break down at line 10. In contrast with the (polynomial) Arnoldi algorithm, this is not a lucky breakdown, in the sense that it does not imply that $\mathcal{Q}_j(A, \boldsymbol{b})$ is an invariant subspace for A. However, such a breakdown is quite unlikely to happen in practice, so it is rarely an issue and hence in this thesis we simply focus on using the implementation described in Algorithm 2.2. The algorithm can be modified to ensure that the subspace is expanded at each iteration by changing how $\boldsymbol{w}_j$ is defined in line 3; we refer for instance to [27, Section 2] for more details.

Denoting by $A_m := V_m^H A V_m$ the projection of A onto the subspace $\mathcal{Q}_m(A, \boldsymbol{b}, \boldsymbol{\xi}_m)$, we can approximate $f(A)\boldsymbol{b}$ vith the rational Arnoldi approximation

$$f(A)\boldsymbol{b} \approx \boldsymbol{f}_m := V_m f(A_m) V_m^H \boldsymbol{e}_1 = \|\boldsymbol{b}\|_2 V_m^H f(A_m)\boldsymbol{e}_1. \tag{2.5.6}$$

Each iteration of the rational Arnoldi algorithm requires the solution of a shifted linear system with the matrix $A - \xi_j I$ for one of the poles ξ_j. One iteration can thus be significantly more expensive than a polynomial Krylov iteration, but the convergence of (2.5.6) to $f(A)\boldsymbol{b}$ can be much faster than (2.5.2) provided that the poles are chosen well. When spectral information of A is available, the poles for the rational Krylov subspace can be chosen by exploiting rational approximations of f on the spectrum or field of values of A [10, 90]. Greedy adaptive strategies have also been proposed in the literature, see for instance [60]. Some specific sequences of poles lead to special cases of the rational Krylov subspace $\mathcal{Q}_m(A, \boldsymbol{b}, \boldsymbol{\xi}_m)$: if all the poles in $\boldsymbol{\xi}_m$ are equal to $\xi \in \mathbb{C} \setminus \sigma(A)$, then $\mathcal{Q}_m(A, \boldsymbol{b}, \boldsymbol{\xi}_m)$ is a *shift-and-invert Krylov subspace*,

$$Q_m(A, \boldsymbol{b}, \boldsymbol{\xi}_m) = \mathcal{K}_m((I - A/\xi)^{-1}, \boldsymbol{b}),$$

which was first investigated for the computation of matrix functions in [119, 151]. If the poles alternate between 0 and ∞, we obtain the *extended Krylov subspace*, introduced in [59], which is of the form

$$\mathcal{Q}_{2m}(A, \boldsymbol{b}, \boldsymbol{\xi}) = A^{-m}\mathcal{K}_{2m}(A, \boldsymbol{b}) = \mathcal{K}_{2m}(A, A^{-m}\boldsymbol{b}).$$

When linear systems can be solved with a direct method, note that using the same pole ξ multiple times allows us to reuse the factorization of $A - \xi I$ without recomputing it, thereby reducing the cost of subsequent linear system solves.

In contrast with the polynomial Krylov case, the projected matrix A_m is not computed while running Algorithm 2.2, and it requires performing some additional computations. However, we can easily recover A_m from $\underline{H}_m$ and $\underline{K}_m$ if we assume that $\xi_m = \infty$. Indeed, if $\xi_m = \infty$ then the last row of $\underline{K}_m$ is zero and hence K_m is invertible, so we have the reduced Arnoldi decomposition

$$AV_m K_m = V_{m+1}\underline{H}_m. \tag{2.5.7}$$

Multiplying (2.5.7) by V_m^H on the left and by K_m^{-1} on the right we obtain

$$A_m = V_m^H A V_m = H_m K_m^{-1}.$$

Note that although H_m and K_m depend on ξ_m in their last column, the basis V_m does not, since ξ_m is only involved in the computation of $\boldsymbol{v}_{m+1}$. This means that we can use the approach described above to compute the projected matrix A_m even if $\xi_m \neq \infty$, by temporarily pretending that $\xi_m = \infty$ to compute A_m as $H_m K_m^{-1}$, and then using the actual ξ_m to compute $\boldsymbol{v}_{m+1}$ and the last columns of $\underline{H}_m$ and $\underline{K}_m$ satisfying (2.5.4).

Similarly to the polynomial case, the approximation (2.5.6) to $f(A)\boldsymbol{b}$ is closely related to the approximation of f with rational functions. The following exactness property is a generalization of Lemma 2.5.2 to the rational case.

Lemma 2.5.7 ([90, Lemma 3.1])**.** Let V_m and A_m as in (2.5.6), and let q_{m-1} be the denominator polynomial of the associated rational Krylov subspace. Then, for any rational function of the form $r = p/q_{m-1}$ with $\deg p \leq m-1$ we have

$$r(A)\boldsymbol{b} = \|\boldsymbol{b}\|_2 V_m r(A_m)\boldsymbol{e}_1.$$

With Lemma 2.5.7 one can prove the following theorem that bounds the error $f(A)\boldsymbol{b} - \boldsymbol{f}_m$ in the rational Krylov case.

Theorem 2.5.8 ([90, Corollary 3.4])**.** Let $\boldsymbol{f}_m$ be the rational Arnoldi approximation (2.5.6) to $f(A)\boldsymbol{b}$, and let q_{m-1} be the denominator polynomial of the rational Krylov subspace. Then, if f is analytic in a neighborhood of $\mathbb{W}(A)$, we have

$$\|f(A)\boldsymbol{b} - \boldsymbol{f}_m\|_2 \leq 2C\|\boldsymbol{b}\|_2 \min_{p \in \Pi_{m-1}} \|f - q_{m-1}^{-1} p\|_{\mathbb{W}(A)},$$

with $C = \sqrt{2} + 1$. If A is normal, the same result holds with $C = 1$.

Proof. The proof is essentially the same as the one of Theorem 2.5.3, using the exactness result of Lemma 2.5.7 in place of Lemma 2.5.2. See also the proof of [90, Corollary 3.4]. □

When A is Hermitian and the poles $\boldsymbol{\xi}_{m+1}$ are real we can construct a basis V_{m+1} of the rational Krylov subspace $\mathcal{Q}_m(A, \boldsymbol{b}, \boldsymbol{\xi}_m)$ with a short-term recurrence by employing the rational Lanczos algorithm (Algorithm 2.3, see also [53], [89, Section 5.2] and [128] for additional details), which also computes the tridiagonal matrices

$$\underline{H}_m = \begin{bmatrix} \alpha_1 & \beta_1 & & \\ \beta_1 & \ddots & \ddots & \\ & \ddots & \ddots & \beta_{m-1} \\ & & \beta_{m-1} & \alpha_m \\ & & & \beta_m \end{bmatrix} \quad \text{and} \quad \underline{K}_m = \begin{bmatrix} 1 & & & \\ & \ddots & & \\ & & & 1 \\ 0 & \dots & & 0 \end{bmatrix} + \begin{bmatrix} \xi_0^{-1} & & & \\ & \ddots & & \\ & & & \xi_m^{-1} \end{bmatrix} \underline{H}_m, \tag{2.5.8}$$

where $\xi_0 := \infty$, with the convention that the inverse of an infinite pole equals zero. Together with V_{m+1}, the matrices $\underline{H}_m$ and $\underline{K}_m$ satisfy the rational Arnoldi relation

$$AV_{m+1}\underline{K}_m = V_{m+1}\underline{H}_m.$$

Algorithm 2.3 Rational Lanczos

Input: $A \in \mathbb{C}^{n\times n}, \boldsymbol{b} \in \mathbb{C}^n, \boldsymbol{\xi}_{m+1} = \{\xi_1, \dots, \xi_m\} \subset \overline{\mathbb{R}} \setminus \Lambda(A)$
Output: $V_{m+1} \in \mathbb{C}^{n\times(m+1)}$, $\alpha_1, \dots, \alpha_m$, $\beta_1, \dots \beta_m \in \mathbb{R}$
1: $\xi_{-1} = \infty, \quad \xi_0 = \infty, \quad \beta_0 = 0, \quad \boldsymbol{v}_0 = \boldsymbol{0}$
2: $\boldsymbol{w} = (I - A/\xi_0)^{-1}\boldsymbol{b}$ ▷ with the convention $A/\infty = 0$
3: $\boldsymbol{v}_1 = \boldsymbol{w}/\|\boldsymbol{w}\|_2$
4: **for** $j = 1, \dots, m$ **do**
5: $\quad [\boldsymbol{v}_{j+1}, \alpha_j, \beta_j] =$ Algorithm 2.4$(A, \boldsymbol{v}_{j-1}, \boldsymbol{v}_j, \xi_{j-2}, \xi_{j-1}, \xi_j, \beta_{j-1})$
6: **end for**
7: $V_{m+1} = [\boldsymbol{v}_1, \dots, \boldsymbol{v}_{m+1}]$

Algorithm 2.4 Single iteration of rational Lanczos

Input: $A \in \mathbb{C}^{n\times n}, \boldsymbol{v}_1, \boldsymbol{v}_2 \in \mathbb{C}^n, \xi_0, \xi_1, \xi_2 \in \overline{\mathbb{R}} \setminus \Lambda(A), \beta_1 \in \mathbb{R}$
Output: $\boldsymbol{v}_3 \in \mathbb{C}^n$, $\alpha_2, \beta_2 \in \mathbb{R}$
1: $\boldsymbol{w} = A(\boldsymbol{v}_2 + \boldsymbol{v}_1\beta_1/\xi_0) - \boldsymbol{v}_1\beta_1$ ▷ with the convention $\beta_1/\infty = 0$
2: $\boldsymbol{s} = (I - A/\xi_1)\boldsymbol{v}_2$ ▷ with the convention $A/\infty = 0$
3: $[\boldsymbol{w}, \boldsymbol{s}] = (I - A/\xi_2)^{-1}[\boldsymbol{w}, \boldsymbol{s}]$
4: $\alpha_2 = (\boldsymbol{w}^H\boldsymbol{v}_2)/(\boldsymbol{s}^H\boldsymbol{v}_2)$
5: $\boldsymbol{w} = \boldsymbol{w} - \boldsymbol{s}\alpha_2$
6: $\beta_2 = \|\boldsymbol{w}\|_2$
7: $\boldsymbol{v}_3 = \boldsymbol{w}/\beta_2$

Note that the key difference between (2.5.8) and (2.5.5) is in the definition of the matrix $\underline{K}_m$, where the diagonal factor and $\underline{H}_m$ are in reversed order.

Another important difference with respect to rational Arnoldi is that in each iteration of the rational Lanczos algorithm we need to solve a linear system in which the right-hand side has two columns instead of one (see line 3 of Algorithm 2.4). However, if the linear systems are solved with a direct method, this does not represent a significant increase in computational cost with respect to a linear system with a single right-hand side, since the matrix factorization has to be computed only once. Moreover, it has been observed that short-term recurrences can produce a numerical loss of orthogonality in the columns of V_m in finite precision arithmetic; see [88, Section 10.3] and [128] for more details.

We are going to employ several other properties of rational Krylov subspaces throughout this thesis. For ease of exposition, we will introduce each property in the chapter where it is first applied.

2.5.3 Pole selection for rational Krylov subspaces

In this section we briefly introduce some pole selection strategies for rational Krylov subspaces that we will employ in the following chapters.

The authors of [114] consider the case of a positive definite Hermitian matrix A with spectrum in $[a, b]$ and a Cauchy-Stieltjes or Laplace-Stieltjes function f, and relate the error for the computation of $f(A)\boldsymbol{b}$ with a rational Krylov method to the *third Zolotarev problem* in approximation theory. The solution to this problem is known explicitly and it can be used to find poles on $(-\infty, 0)$ that provide in some sense an optimal convergence rate for the rational Krylov method for the two classes of functions. The optimal poles for Laplace-Stieltjes functions satisfy the following convergence bound.

Theorem 2.5.9 ([114, Corollary 3])**.** Let f be a Laplace-Stieljes function and A a Hermitian positive definite matrix with $\Lambda(A) \subset [a, b]$. Then the approximation $\boldsymbol{f}_m$ from (2.5.6) with the optimal poles from [114, Theorem 3] is such that

$$\|f(A)\boldsymbol{b} - \boldsymbol{f}_m\|_2 \leq 8\gamma f(0^+)\|\boldsymbol{b}\|_2\rho^{\ell/2},$$

where

$$\gamma = 2.23 + \frac{2}{\pi}\log\left(4\ell\sqrt{\frac{b}{\pi a}}\right) \qquad \text{and} \qquad \rho = \exp\left(-\frac{\pi^2}{\log(4b/a)}\right).$$

Remark 2.5.10. We mention that the authors of [114] use a slightly different definition of a rational Krylov subspace, in which the vector $\boldsymbol{b}$ is not included, so the rational functions associated with a subspace of dimension m have denominator of degree m instead of $m-1$, and the bound in Theorem 2.5.9 holds when this alternative definition is used. However, we are mainly interested in the asymptotic convergence rate of a related equidistributed sequence of poles (which we discuss below), so we can use this result regardless.

The optimal Zolotarev poles used in Theorem 2.5.9 are not nested: indeed, the poles for the optimal approximant of degree $m+1$ are usually completely different from those of the optimal approximant of degree m. Thus, these poles cannot be used to construct the Krylov subspace incrementally, and they are useful in practice only if one fixes in advance the number of iterations to perform, for instance by relying on an a priori error bound. This drawback can be overcome by constructing a nested sequence of poles that is equidistributed according to the limit measure identified by the optimal poles, which can be done with the method of equidistributed sequences described in [114, Section 3.5]. These poles have the same asymptotic convergence rate as the optimal Zolotarev poles and are usually better for practical purposes. Knowledge of a positive interval $[a, b]$ that contains $\Lambda(A)$ is needed to compute these poles. However, the integral expression of the Cauchy-Stieltjes or Laplace-Stieltjes function is used only for the theoretical analysis, and it is not explicitly required to compute the poles.

When f is a Cauchy-Stieltjes function, an optimal sequence of poles and an associated error bound are given in [114, Corollary 4]. The pole selection strategy described in [114] for Cauchy-Stieltjes functions is a special case of the one introduced in [10, Section 6] for Markov functions, where the authors consider the more general problem of rational approximation in the non-Hermitian case. The pole sequence and the associated error bound are given in [10, Theorem 6.6]. These poles are also related to the third Zolotarev problem, and they are defined in terms of Jacobi elliptic functions; we refer to the proof of [10, Theorem 6.6] for additional details.

2.5.4 Approximation of quadratic forms

When computing approximations to quadratic forms involving $f(A)$ for a Hermitian matrix A, polynomial and rational Krylov subspace methods have a faster convergence compared to the approximation of $f(A)\boldsymbol{b}$. Here we focus directly on the rational case, as the polynomial one can be obtained as a special case. Similarly to (2.5.6), we can approximate the quadratic form $\boldsymbol{b}^H f(A)\boldsymbol{b}$ with

$$\boldsymbol{b}^H f(A)\boldsymbol{b} \approx \psi_m := \boldsymbol{b}^H V_m f(A_m) V_m^H \boldsymbol{b} = \|\boldsymbol{b}\|_2^2 \boldsymbol{e}_1^T f(A_m)\boldsymbol{e}_1. \tag{2.5.9}$$

It is immediately clear from (2.5.9) that only the projected matrix A_m is needed to compute ψ_m, while the basis V_m is not explicitly required. This means that one can construct the basis V_m with the rational Lanczos algorithm by exploiting a short-term recurrence and keeping in storage only the last three basis vectors at any given time, without requiring a second pass to recompute the basis V_m as in the case of $f(A)\boldsymbol{b}$; see for instance [128].

The approximation (2.5.9) is exact on rational functions with a degree that is roughly doubled with respect to the dimension of the Krylov subspace, as shown in the following lemma.

Lemma 2.5.11. Let A be Hermitian, and let V_m be the basis of $\mathcal{Q}_m(A, \boldsymbol{b}, \boldsymbol{\xi}_m)$ constructed by Algorithm 2.2. Let $A_m = V_m^H A V_m$, and let q_{m-1} be the denominator polynomial of the associated rational Krylov subspace. Then, for any rational function $r = p/q_{m-1}^2$ with $\deg p \leq 2m-1$ we have

$$\boldsymbol{b}^H r(A)\boldsymbol{b} = \|\boldsymbol{b}\|_2^2 \boldsymbol{e}_1^T r(A_m)\boldsymbol{e}_1$$

Proof. See for instance [90, Remark 3.2] or [128, Proposition 3.1]. □

The exactness result of Lemma 2.5.11 leads to the following error bound, which can be proved similarly to Theorem 2.5.8.

Proposition 2.5.12. Let A be Hermitian, and let ψ_m be the rational Arnoldi approximation (2.5.9) to $\boldsymbol{b}^H f(A)\boldsymbol{b}$, and let q_{m-1} be the denominator polynomial of the rational Krylov subspace. Then we have

$$|\boldsymbol{b}^H f(A)\boldsymbol{b} - \psi_m| \leq \|\boldsymbol{b}\|_2^2 \min_{p \in \Pi_{2m-1}} \left(\|f - q_{m-1}^{-2} p\|_{\Lambda(A)} + \|f - q_{m-1}^{-2} p\|_{\Lambda(A_m)} \right).$$

The approximation (2.5.9) can be also interpreted in terms of rational Gauss quadrature rules, see for instance [2, 130].

Chapter 3

Error bounds for rational Krylov methods

In applications it is often essential to have bounds or estimates on the error of the approximation of $f(A)\boldsymbol{b}$ with a Krylov subspace method, in order to either predict the number of iterations needed to reach a given accuracy or to have a reliable stopping criterion. A simple way to estimate the error $\|f(A)\boldsymbol{b} - \boldsymbol{f}_m\|_2$ is to simply approximate it with $\|\boldsymbol{f}_{m+1} - \boldsymbol{f}_m\|_2$, which can be easily evaluated by running the Krylov subspace method for one additional iteration. This error estimate is often quite reliable as a stopping criterion, but it does not provide any theoretical guarantee on the error and it may sometimes significantly underestimate it, especially when convergence is slow or stagnating for a few iterations. In the literature there have been several papers devoted to developing a priori and a posteriori error bounds and estimates, both in the polynomial [40, 78, 80] and in the rational Krylov setting [10, 114, 120].

In this chapter we present some error bounds for the approximation of $f(A)\boldsymbol{b}$ with rational Krylov methods, obtained by exploiting an integral representation of the function f. In Section 3.2 we review a bound for the approximation of $f(A)\boldsymbol{b}$ that was given in [89, Section 6.6.2], and we extend it to the quadratic form $\boldsymbol{b}^H f(A)\boldsymbol{b}$ in the case of Hermitian A in Section 3.2.1. In Section 3.3 we obtain an integral expression for the error in the approximation of $f(A)\boldsymbol{b}$ by exploiting an integral expression of f, which we then use to derive both a priori and a posteriori bounds when A is Hermitian. To obtain these bounds we first need to analyze the relation between rational Krylov subspaces associated to shifted matrices $A - zI$. The contents of this chapter are based in part on [21] and [143].

3.1 Krylov subspaces associated with shifted matrices

In this section we give some results on Krylov subspaces associated to shifted matrices and their relation to the approximate solutions of shifted linear systems, which are going to be useful in the derivation of the error bounds later in this chapter. Unless explicitly stated, we do not assume that the matrix A is Hermitian.

3.1.1 Polynomial Krylov subspaces

Let us start by recalling some basic facts regarding shifted polynomial Krylov subspaces and the solution of shifted linear systems using the full orthogonalization method (FOM) [134], which will serve as an introduction to the next part, in which we will consider rational Krylov subspaces.

Recall that after m iterations FOM constructs a basis $V_m \in \mathbb{C}^{n\times m}$ of the polynomial Krylov subspace $\mathcal{K}_m(A, \boldsymbol{b})$ and approximates the solution to the linear system $A\boldsymbol{x} = \boldsymbol{b}$ with

$$\boldsymbol{x}_m = V_m A_m^{-1} \boldsymbol{e}_1 \|\boldsymbol{b}\|_2, \qquad \text{where } A_m = V_m^H A V_m.$$

This is equivalent to imposing that the residual $\boldsymbol{r}_m = \boldsymbol{b} - A\boldsymbol{x}_m$ is orthogonal to the basis V_m. The residual $\boldsymbol{r}_m$ can be elegantly expressed in terms of the characteristic polynomial of A_m. If we denote

by χ_m the characteristic polynomial of A_m, that is $\chi_m(\lambda) = \det(\lambda I - A_m)$, by [127, eq. (3.8)] we have

$$\boldsymbol{r}_m = \frac{1}{\chi_m(0)}\chi_m(A)\boldsymbol{b}. \tag{3.1.1}$$

Moreover, since $\boldsymbol{r}_m \in \mathcal{K}_{m+1}(A, \boldsymbol{b})$ and $\boldsymbol{r}_m \perp \mathcal{K}_m(A, \boldsymbol{b})$, the residual $\boldsymbol{r}_m$ is proportional to the next basis vector $\boldsymbol{v}_{m+1}$ (see, e.g., [134, Proposition 6.7]).

A similar argument can be used for the solution of a shifted linear system $(A - zI)\boldsymbol{x}(z) = \boldsymbol{b}$, for any $z \in \mathbb{C}$. Due to the shift invariance of polynomial Krylov subspaces, i.e. $\mathcal{K}_m(A - zI, \boldsymbol{b}) = \mathcal{K}_m(A, \boldsymbol{b})$, after m iterations of FOM applied to the shifted system we have constructed the same basis V_m as in the case of the linear system $A\boldsymbol{x} = \boldsymbol{b}$, and therefore the approximate solution to the shifted linear system is given by

$$\boldsymbol{x}_m(z) = V_m(A_m^z)^{-1}\boldsymbol{e}_1\|\boldsymbol{b}\|_2, \qquad \text{where } A_m^z := V_m^H(A - zI)V_m = A_m - zI.$$

The residual can be expressed using (3.1.1) with A_m^z in place of A_m, yielding

$$\boldsymbol{r}_m(z) = \boldsymbol{b} - (A - zI)\boldsymbol{x}_m(z) = \frac{1}{\chi_m^z(0)}\chi_m^z(A - zI)\boldsymbol{b},$$

where

$$\chi_m^z(\lambda) := \det(\lambda I - A_m^z) = \chi_m(\lambda + z).$$

With simple algebraic manipulations, the above expression can be rewritten as

$$\boldsymbol{r}_m(z) = \frac{1}{\chi_m(z)}\chi_m(A)\boldsymbol{b} = \frac{\chi_m(0)}{\chi_m(z)}\boldsymbol{r}_m(0). \tag{3.1.2}$$

In other words, the residuals of shifted linear systems are all collinear, and they are proportional to $\boldsymbol{v}_{m+1}$, the next vector in the Krylov basis. The constant in (3.1.2) can be written explicitly in terms of the eigenvalues of the projected matrix A_m. These facts are well known in the literature, see for instance [141, Proposition 2.1] and [150, Lemma 5].

3.1.2 Rational Krylov subspaces

The results that we present in this section have strong similarities with the ones presented in Section 3.1.1 for the case of shifted polynomial Krylov subspaces and FOM. Let us consider the family of shifted linear systems

$$(A - zI)\boldsymbol{x} = \boldsymbol{b}, \qquad z \in \mathbb{C}.$$

In analogy to FOM and to the matrix function approximation (2.5.6), we can extract from the rational Krylov subspace $\mathcal{Q}_m(A, \boldsymbol{b}, \boldsymbol{\xi}_m)$ the approximate linear system solution

$$\mathtt{x}_m(z) := V_m(A_m - zI)^{-1}\boldsymbol{e}_1\|\boldsymbol{b}\|_2. \tag{3.1.3}$$

Let us introduce the notation

$$\begin{aligned} \mathtt{res}_m(z) &:= \boldsymbol{b} - (A - zI)\mathtt{x}_m(z), \\ \mathtt{err}_m(z) &:= (A - zI)^{-1}\mathtt{res}_m(z), \end{aligned} \tag{3.1.4}$$

which are the residual and error of the shifted linear system, respectively.

Lemma 3.1.1. Assuming that K_m is nonsingular, the shifted linear system residual $\mathtt{res}_m(z)$ can be written as

$$\mathtt{res}_m(z) = -(I - V_m V_m^H)(h_{m+1,m}I - k_{m+1,m}A)\boldsymbol{v}_{m+1}\boldsymbol{e}_m^T K_m^{-1}(A_m - zI)^{-1}\boldsymbol{e}_1\|\boldsymbol{b}\|_2, \tag{3.1.5}$$

where $h_{m+1,m}$ and $k_{m+1,m}$ are the last subdiagonal elements of $\underline{H}_m$ and $\underline{K}_m$, respectively.

Proof. Let us consider the rational Arnoldi decomposition

$$AV_{m+1}\underline{K}_m = V_{m+1}\underline{H}_m$$

obtained after m iterations of the rational Arnoldi algorithm. We can write this relation more explicitly as

$$AV_mK_m + A\boldsymbol{v}_{m+1}k_{m+1,m}\boldsymbol{e}_m^T = V_mH_m + \boldsymbol{v}_{m+1}h_{m+1,m}\boldsymbol{e}_m^T. \tag{3.1.6}$$

The ratios of corresponding subdiagonal elements of $\underline{H}_m$ and $\underline{K}_m$ are the poles $\xi_1, \dots, \xi_m$ of the rational Krylov subspace; this follows from the definitions of $\underline{H}_m$ and $\underline{K}_m$ given in Section 2.5.2, see also [89, Section 5.1] and [26]. In particular, $h_{m+1,m}k_{m+1,m}^{-1} = \xi_m$, so $k_{m+1,m} = 0$ when $\xi_m = \infty$. For all $z \in \mathbb{C}$ we have

$$(A - zI)V_m = V_m(H_mK_m^{-1} - zI) + (h_{m+1,m}I - k_{m+1,m}A)\boldsymbol{v}_{m+1}\boldsymbol{e}_m^TK_m^{-1}. \tag{3.1.7}$$

Recalling that $A_m = V_m^HAV_m$, we see from (3.1.6) that

$$A_m = H_mK_m^{-1} - k_{m+1,m}V_m^HA\boldsymbol{v}_{m+1}\boldsymbol{e}_m^TK_m^{-1},$$

and combining with (3.1.7) we get for A_m the identity

$$\begin{aligned}(A - zI)V_m &= V_m(A_m - zI) + \left(h_{m+1,m}I - k_{m+1,m}(I - V_mV_m^H)A\right)\boldsymbol{v}_{m+1}\boldsymbol{e}_m^TK_m^{-1} \\ &= V_m(A_m - zI) + (I - V_mV_m^H)(h_{m+1,m}I - k_{m+1,m}A)\boldsymbol{v}_{m+1}\boldsymbol{e}_m^TK_m^{-1},\end{aligned} \tag{3.1.8}$$

where we used the fact that $\boldsymbol{v}_{m+1} \perp \operatorname{range}(V_m)$ for the last equality. Using (3.1.8), the residual $\mathtt{res}_m(z)$ can be written as

$$\mathtt{res}_m(z) = -(I - V_mV_m^H)(h_{m+1,m}I - k_{m+1,m}A)\boldsymbol{v}_{m+1}\boldsymbol{e}_m^TK_m^{-1}(A_m - zI)^{-1}\boldsymbol{e}_1\|\boldsymbol{b}\|_2,$$

concluding the proof. □

This shows that the residuals $\mathtt{res}_m(z)$ are collinear for all $z \in \mathbb{C}$. In particular, when $\xi_m = \infty$ we have $k_{m+1,m} = 0$ and (3.1.5) simplifies to

$$\mathtt{res}_m(z) = -h_{m+1,m}\boldsymbol{v}_{m+1}\boldsymbol{e}_m^TK_m^{-1}(A_m - zI)^{-1}\boldsymbol{e}_1\|\boldsymbol{b}\|_2, \tag{3.1.9}$$

i.e., the residuals are collinear to the next basis vector $\boldsymbol{v}_{m+1}$, as in the polynomial case discussed Section 3.1.1. The identity (3.1.9) has been used in past literature, see for instance [89, Section 6.6.2] and [21, Section 4.3].

The residual $\mathtt{res}_m(z)$ can be also linked to the characteristic polynomial $\chi_m(\lambda) = \det(\lambda I - A_m)$. For simplicity, let us focus first on the case $z = 0$, and let us use the notation $\mathtt{res}_m := \mathtt{res}_m(0)$ and similarly $\mathtt{x}_m := \mathtt{x}_m(0)$. The residual $\mathtt{res}_m$ is orthogonal to $\mathcal{Q}_m(A, \boldsymbol{b})$ because of the factor $(I - V_mV_m^H)$ in (3.1.5); recalling that $\xi_m = h_{m+1,m}/k_{m+1,m}$, we also see that $\mathtt{res}_m$ belongs to the subspace

$$(I - \xi_m^{-1}A)\mathcal{Q}_{m+1}(A, \boldsymbol{b}) = q_{m-1}(A)^{-1}\mathcal{K}_{m+1}(A, \boldsymbol{b}) \supset \mathcal{Q}_m(A, \boldsymbol{b}).$$

If we denote by $\chi_m(\lambda) = \det(\lambda I - A_m)$ the characteristic polynomial of A_m, by [89, Lemma 4.5] we have

$$\chi_m(A)q_{m-1}(A)^{-1}\boldsymbol{b} \perp \mathcal{Q}_m(A, \boldsymbol{b}),$$

and moreover $\chi_m(A)q_{m-1}(A)^{-1}\boldsymbol{b} \in q_{m-1}(A)^{-1}\mathcal{K}_{m+1}(A, \boldsymbol{b})$. The vectors $\chi_m(A)q_{m-1}(A)^{-1}\boldsymbol{b}$ and $\mathtt{res}_m$ belong to the same subspace of dimension $m+1$ which contains $\mathcal{Q}_m(A, \boldsymbol{b})$, and they are both orthogonal to $\mathcal{Q}_m(A, \boldsymbol{b})$, so they must be collinear, i.e., there exists a constant $\alpha \in \mathbb{C}$ such that

$$\mathtt{res}_m = \alpha\chi_m(A)q_{m-1}(A)^{-1}\boldsymbol{b}.$$

The value of the constant α can be determined by looking at the constant terms in the identity

$$q_{m-1}(A)\boldsymbol{b} - q_{m-1}(A)A\mathtt{x}_m = \alpha\chi_m(A)\boldsymbol{b},$$

which gives us

$$q_{m-1}(0) = \alpha\chi_m(0).$$

As a consequence, we have the following elegant expression for the residual with $z = 0$,

$$\mathtt{res}_m(0) = \frac{q_{m-1}(0)}{\chi_m(0)}\chi_m(A)q_{m-1}(A)^{-1}\boldsymbol{b}. \tag{3.1.10}$$

This result can now be easily generalized to all $z \in \mathbb{C}$. Consider the shifted linear system $(A-zI)\boldsymbol{x} = \boldsymbol{b}$, which has the approximate solution

$$\mathtt{x}_m(z) = V_m(A_m^z)^{-1}\boldsymbol{e}_1\|\boldsymbol{b}\|_2, \qquad \text{where } A_m^z = V_m^H(A - zI)V_m = A_m - zI.$$

Recall that the subspace $\mathcal{Q}_m(A, \boldsymbol{b}, \boldsymbol{\xi}_m)$ coincides with the subspace $\mathcal{Q}_m(A-zI, \boldsymbol{b}, [\xi_j - z]_{j=1}^{m-1})$, in which the poles have been shifted by z. The denominator polynomial associated to this rational Krylov subspace is therefore

$$q_{m-1}^z(\lambda) = \prod_{j=1}^{m-1}(\lambda - (\xi_j - z)) = q_{m-1}(\lambda + z),$$

and the characteristic polynomial χ_m^z of the projected matrix A_m^z is

$$\chi_m^z(\lambda) = \det(\lambda I - A_m^z) = \chi_m(\lambda + z).$$

If we write the residual $\mathtt{res}_m(z) = \boldsymbol{b} - (A - zI)\mathtt{x}_m(z)$ using (3.1.10) with A replaced by $A - zI$, we get

$$\begin{aligned}\mathtt{res}_m(z) &= \frac{q_{m-1}^z(0)}{\chi_m^z(0)}\chi_m^z(A - zI)q_{m-1}^z(A - zI)^{-1}\boldsymbol{b} \\ &= \frac{q_{m-1}(z)}{\chi_m(z)}\chi_m(A)q_{m-1}(A)^{-1}\boldsymbol{b} \\ &= \frac{q_{m-1}(z)}{q_{m-1}(0)} \cdot \frac{\chi_m(0)}{\chi_m(z)}\,\mathtt{res}_m(0).\end{aligned} \tag{3.1.11}$$

Letting $\theta_j^{(m)}$ with $j = 1, \ldots, m$ denote the eigenvalues of A_m, we also obtain the following explicit expression for the shifted residual:

$$\mathtt{res}_m(z) = \varphi_m(z)\mathtt{res}_m(0), \qquad \text{where } \varphi_m(z) = \prod_{j=1}^{m-1}\frac{\xi_j - z}{\xi_j}\prod_{j=1}^{m}\frac{\theta_j^{(m)}}{\theta_j^{(m)} - z}.$$

3.2 A posteriori error bounds via the residue theorem

In this section we obtain some a posteriori error bounds for analytic functions, derived by exploiting the Cauchy integral formula and the residue theorem. We first review an error bound given in [89, Section 6.6.2] for the rational Krylov approximation of $f(A)\boldsymbol{b}$, and we then extend it to the approximation of the quadratic form $\boldsymbol{b}^H f(A)\boldsymbol{b}$ in Section 3.2.1, modifying it in order to account for the faster convergence rate for quadratic forms in the case of Hermitian A.

Recall that after m iterations of the rational Arnoldi algorithm we have the rational Arnoldi relation

$$AV_{m+1}\underline{K}_m = V_{m+1}\underline{H}_m,$$

where $\mathrm{range}(V_{m+1}) = \mathcal{Q}_{m+1}(A, \boldsymbol{b}, \boldsymbol{\xi}_{m+1})$ and $\underline{K}_m$, $\underline{H}_m$ are $(m+1)\times m$ upper Hessenberg matrices with full rank m. Let us focus on the situation when $\xi_m = \infty$: in this case the last row of $\underline{K}_m$ is zero, and the Arnoldi relation simplifies to

$$AV_mK_m = V_mH_m + \boldsymbol{v}_{m+1}h_{m+1,m}\boldsymbol{e}_m^T,$$

where $h_{m+1,m}\boldsymbol{e}_m^T$ is the last row of $\underline{H}_m$. Since K_m is nonsingular, we can rewrite the above identity as

$$AV_m = V_mA_m + \boldsymbol{v}_{m+1}h_{m+1,m}\boldsymbol{e}_m^TK_m^{-1}, \qquad \text{where} \quad A_m = V_m^HAV_m = H_mK_m^{-1}.$$

Remark 3.2.1. The assumption that $\xi_m = \infty$ gives us a simpler rational Arnoldi relation and hence it allows us to derive the error bound more easily. Such an assumption is not restrictive, since the value of ξ_m does not have any impact on V_m and A_m, but only on $\boldsymbol{v}_{m+1}$ and the last column of $\underline{H}_m$ and $\underline{K}_m$. We will discuss in Section 3.2.2 some techniques that can be used to compute the bound for all m, without having to set the corresponding poles $\xi_m = \infty$.

Assuming that f is analytic in a neighbourhood of the spectrum of A and denoting by Γ a contour that surrounds $\Lambda(A)$, by the Cauchy integral formula we have

$$f(A)\boldsymbol{b} = \frac{1}{2\pi i}\int_\Gamma f(z)(zI - A)^{-1}\boldsymbol{b}\,\mathrm{d}z.$$

If Γ also encloses the eigenvalues of the projected matrix A_m, we also have

$$\boldsymbol{f}_m = \|\boldsymbol{b}\|_2 V_m f(A_m)\boldsymbol{e}_1 = \frac{\|\boldsymbol{b}\|_2}{2\pi i}\int_\Gamma f(z)V_m(zI - A_m)^{-1}\boldsymbol{e}_1\,\mathrm{d}z,$$

and by combining the two above identities we obtain the following expression for the error

$$f(A)\boldsymbol{b} - \boldsymbol{f}_m = \frac{1}{2\pi i}\int_\Gamma f(z)(zI - A)^{-1}\mathtt{res}_m(z)\,\mathrm{d}z, \tag{3.2.1}$$

where we recognize $\mathtt{res}_m(z)$ as the residual of the shifted linear system $(A - zI)\boldsymbol{x} = \boldsymbol{b}$ associated with the approximate solution $\mathtt{x}_m(z)$ extracted from the rational Krylov subspace $\mathcal{Q}_m(A, \boldsymbol{b})$. It follows from (3.1.9) that

$$\mathtt{res}_m(z) = \|\boldsymbol{b}\|_2\varphi_m(z)\boldsymbol{v}_{m+1}, \qquad \text{with} \quad \varphi_m(z) = h_{m+1,m}\boldsymbol{e}_m^T K_m^{-1}(zI - A_m)^{-1}\boldsymbol{e}_1. \tag{3.2.2}$$

Following [89, Section 6.6.2], we assume that A_m is diagonalizable and we denote its spectral decomposition by $A_m = X_m D_m X_m^{-1}$, with $D_m = \mathrm{diag}(\theta_1, \ldots, \theta_m)$. Defining the vectors

$$\boldsymbol{\alpha}^T = [\alpha_1, \ldots, \alpha_m] = h_{m+1,m}\boldsymbol{e}_m^T K_m^{-1} X_m \qquad \text{and} \qquad \boldsymbol{\beta} = [\beta_1, \ldots, \beta_m]^T = X_m^{-1}\boldsymbol{e}_1,$$

we have

$$\varphi_m(z) = \boldsymbol{\alpha}^T(zI - D_m)\boldsymbol{\beta} = \sum_{j=1}^m \alpha_j\beta_j\frac{1}{z - \theta_j}.$$

Plugging this expression for $\varphi_m(z)$ into (3.2.1), we obtain

$$\begin{aligned} f(A)\boldsymbol{b} - \boldsymbol{f}_m &= \frac{\|\boldsymbol{b}\|_2}{2\pi i}\sum_{j=1}^m \alpha_j\beta_j\int_\Gamma \frac{f(z)}{z - \theta_j}(zI - A)^{-1}\boldsymbol{v}_{m+1}\,\mathrm{d}z \\ &= \|\boldsymbol{b}\|_2\sum_{j=1}^m \alpha_j\beta_j\big(f(A) - f(\theta_j)I\big)(A - \theta_j I)^{-1}\boldsymbol{v}_{m+1}, \end{aligned}$$

where the second equality follows from the residue theorem [95, Theorem 4.7a]. The expression in the sum is a matrix function of A, with the function

$$G_m(z) := \sum_{j=1}^m \alpha_j\beta_j \begin{cases} \dfrac{f(z) - f(\theta_j)}{z - \theta_j} & \text{if } z \neq \theta_j, \\ f'(\theta_j) & \text{if } z = \theta_j, \end{cases}$$

where the expression for $z = \theta_j$ is obtained by taking the limit $z \to \theta_j$ in the one for $z \neq \theta_j$. Summarizing, we have written the error in the approximation of $f(A)\boldsymbol{b}$ as

$$f(A)\boldsymbol{b} - \boldsymbol{f}_m = \|\boldsymbol{b}\|_2 G_m(A)\boldsymbol{v}_{m+1}.$$

This error expression holds for a general matrix A. If we assume that A is Hermitian, we obtain the following upper and lower a posteriori bounds on the error, which correspond to [89, eq. (6.15)].

Theorem 3.2.2. Assuming that A is a Hermitian matrix with spectrum contained in the interval $[a, b]$, we have

$$\|\boldsymbol{b}\|_2 \min_{z\in[a,b]} |G_m(z)| \leq \|f(A)\boldsymbol{b} - \boldsymbol{f}_m\| \leq \|\boldsymbol{b}\|_2 \max_{z\in[a,b]} |G_m(z)|. \tag{3.2.3}$$

If we do not assume that A is Hermitian, we still have the upper bound

$$\|f(A)\boldsymbol{b} - \boldsymbol{f}_m\| \leq (1+\sqrt{2})\|\boldsymbol{b}\|_2 \max_{z\in\mathbb{W}(A)} |G_m(z)|.$$

However, this general bound is more difficult to evaluate compared to (3.2.3) since the numerical range $\mathbb{W}(A)$ is hard to compute in general, and it may even provide no meaningful information on convergence when f has singularities contained in $\mathbb{W}(A)$.

3.2.1 Error bounds for quadratic forms

In this section we obtain a refined a posteriori error bound for the quadratic form $\boldsymbol{b}^H f(A)\boldsymbol{b}$ in the case of a Hermitian matrix A, that accounts for the faster convergence rate predicted, e.g., by Proposition 2.5.12. The contents of this section are partially based on [21, Section 4.3].

When A is Hermitian, we have

$$\begin{aligned}\boldsymbol{b}^H(zI-A)^{-1}\mathtt{res}_m(z) &= \mathtt{res}_m(z)^H(zI-A)^{-1}\mathtt{res}_m(z) - \mathtt{x}_m(z)^H\mathtt{res}_m(z)\\ &= \mathtt{res}_m(z)^H(zI-A)^{-1}\mathtt{res}_m(z),\end{aligned}$$

where we exploited the fact that $\mathtt{x}_m(z) \in \mathcal{Q}_m(A,\boldsymbol{b}) \perp \mathtt{res}_m(z)$. By using this property in conjunction with (3.2.1), we can write the error for the quadratic form $\boldsymbol{b}^H f(A)\boldsymbol{b}$ as

$$\begin{aligned}\boldsymbol{b}^H f(A)\boldsymbol{b} - \psi_m &= \frac{1}{2\pi i}\int_\Gamma f(z)\mathtt{res}_m(z)^H(zI-A)^{-1}\mathtt{res}_m(z)\,\mathrm{d}z\\ &= \frac{1}{2\pi i}\|\boldsymbol{b}\|_2^2\int_\Gamma f(z)\varphi_m(z)^2\boldsymbol{v}_{m+1}^H(zI-A)^{-1}\boldsymbol{v}_{m+1}\,\mathrm{d}z,\end{aligned} \tag{3.2.4}$$

where $\psi_m = \boldsymbol{b}^H\boldsymbol{f}_m$ is as defined in (2.5.9) and $\varphi_m(z)$ is as defined in (3.2.2). We can now follow the same steps used in the derivation of the bound for $f(A)\boldsymbol{b}$ to bound the integral in (3.2.4). Since A is Hermitian, it has a spectral decomposition $A_m = U_m D_m U_m^H$ with U_m unitary and $D_m = \mathrm{diag}(\theta_1,\ldots,\theta_m)$, and the vectors $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ are given by

$$\boldsymbol{\alpha}^T = [\alpha_1,\ldots,\alpha_m] = h_{m+1,m}\boldsymbol{e}_m^T K_m^{-1}U_m \qquad \text{and} \qquad \boldsymbol{\beta} = [\beta_1,\ldots,\beta_m]^T = U_m^H\boldsymbol{e}_1,$$

so that we have

$$\varphi_m(z) = \sum_{j=1}^m \alpha_j\beta_j\frac{1}{z-\theta_j},$$

and

$$\varphi_m(z)^2 = \sum_{j=1}^m \alpha_j^2\beta_j^2\frac{1}{(z-\theta_j)^2} + 2\sum_{j=1}^m \alpha_j\beta_j\gamma_j\frac{1}{z-\theta_j},$$

where we defined $\gamma_j = \sum_{\ell\,:\,\ell\neq j} \alpha_\ell\beta_\ell \dfrac{1}{\theta_j-\theta_\ell}$. By plugging this expression into (3.2.4) we get

$$\begin{aligned}
\frac{1}{\|\boldsymbol{b}\|_2^2}(\boldsymbol{b}^H f(A)\boldsymbol{b} - \psi_m) &= \frac{1}{2\pi i}\int_\Gamma f(z)\varphi_m(z)^2 \boldsymbol{v}_{m+1}^H (zI-A)^{-1}\boldsymbol{v}_{m+1}\,\mathrm{d}z \\
&= \sum_{j=1}^m \alpha_j^2\beta_j^2 \frac{1}{2\pi i}\int_\Gamma \frac{f(z)}{(z-\theta_j)^2}\boldsymbol{v}_{m+1}^H (zI-A)^{-1}\boldsymbol{v}_{m+1}\,\mathrm{d}z \\
&\quad + 2\sum_{j=1}^m \alpha_j\beta_j\gamma_j \frac{1}{2\pi i}\int_\Gamma \frac{f(z)}{z-\theta_j}\boldsymbol{v}_{m+1}^H (zI-A)^{-1}\boldsymbol{v}_{m+1}\,\mathrm{d}z \\
&= \sum_{j=1}^m \alpha_j^2\beta_j^2 \boldsymbol{v}_{m+1}^H \Big((f(A)-f(\theta_j)I)(A-\theta_j I)^{-2} - f'(\theta_j)(A-\theta_j I)^{-1}\Big)\boldsymbol{v}_{m+1} \\
&\quad + 2\sum_{j=1}^m \alpha_j\beta_j\gamma_j \boldsymbol{v}_{m+1}^H (f(A)-f(\theta_j)I)(A-\theta_j I)^{-1}\boldsymbol{v}_{m+1},
\end{aligned}$$

where for the last equality we again used the residue theorem [95, Theorem 4.7a].

Let us define

$$g_m(z) := \sum_{j=1}^m \begin{cases} \alpha_j^2\beta_j^2\left(\dfrac{f(z)-f(\theta_j)}{(z-\theta_j)^2} - \dfrac{f'(\theta_j)}{z-\theta_j}\right) + 2\alpha_j\beta_j\gamma_j\dfrac{f(z)-f(\theta_j)}{z-\theta_j} & \text{if } z\neq\theta_j, \\ \dfrac{1}{2}\alpha_j^2\beta_j^2 f''(\theta_j) + 2\alpha_j\beta_j\gamma_j f'(\theta_j) & \text{if } z=\theta_j, \end{cases} \tag{3.2.5}$$

where the expression for $z=\theta_j$ is obtained by taking the limit for $z\to\theta_j$ in the definition for $z\neq\theta_j$. We have written the error in the approximation of $\boldsymbol{b}^H f(A)\boldsymbol{b}$ as

$$\boldsymbol{b}^H f(A)\boldsymbol{b} - \psi_m = \|\boldsymbol{b}\|_2^2 \boldsymbol{v}_{m+1}^H g_m(A)\boldsymbol{v}_{m+1},$$

and this expression immediately leads to the following a posteriori bound for the quadratic form error.

Theorem 3.2.3. Let A be a Hermitian matrix with $\Lambda(A)\subset[a,b]$. Using the same notation as above, we have

$$\|\boldsymbol{b}\|_2^2 \min_{z\in[a,b]}|g_m(z)| \le |\boldsymbol{b}^H f(A)\boldsymbol{b} - \psi_m| \le \|\boldsymbol{b}\|_2^2 \max_{z\in[a,b]}|g_m(z)|. \tag{3.2.6}$$

Remark 3.2.4. We are mainly interested in the upper bound in (3.2.6) to have a reliable stopping criterion for the rational Krylov method, although the lower bound can also be of interest. Under certain assumptions, it is also possible to obtain upper and lower bounds for the quadratic form $\boldsymbol{b}^H f(A)\boldsymbol{b}$ using pairs of rational Gauss quadrature rules, such as Gauss and Gauss-Radau quadrature rules. We refer to [2] for more details.

Example 3.2.1. In this example we test the accuracy of the lower and upper bounds given in (3.2.6) for the approximation of $\boldsymbol{b}^H f(A)\boldsymbol{b}$ with a rational Krylov method, using the poles for Cauchy-Stieltjes functions described in Section 2.5.3. We consider a 2000×2000 real symmetric matrix A with logspaced eigenvalues in the interval $\Sigma = [10^{-3}, 10^3]$, a random vector $\boldsymbol{b}$ and the functions $f_1(z) = z\log(z)$ and $f_2(z) = \sqrt{z}$. The upper and lower bounds are computed numerically by evaluating g_m on a discretization of the interval $[\lambda_{\min}, \lambda_{\max}]$. In addition to the lower and upper bounds, we also consider a heuristic estimate of the error given by the geometric mean of the upper and lower bound in (3.2.6), i.e.,

$$\texttt{est}_m = \|\boldsymbol{b}\|_2^2 \sqrt{\min_{z\in\Sigma}|g_m(z)| \max_{z\in\Sigma}|g_m(z)|}. \tag{3.2.7}$$

We also include a simple error estimate obtained as a difference of consecutive iterates, $|\boldsymbol{b}^H f(A)\boldsymbol{b} - \psi_m| \approx |\psi_{m+1} - \psi_m|$. The results are shown in Figure 3.1. Both the upper and lower bounds from Theorem 3.2.3 match the shape of the convergence curve quite well. Quite surprisingly, the geometric mean of the bounds gives a very accurate estimate for the error, even in the case when the bounds themselves are less accurate. On the other hand, the consecutive difference error estimate often underestimates the error, especially when convergence is stagnating for a few iterations (this is particularly evident in the left plot of Figure 3.1).

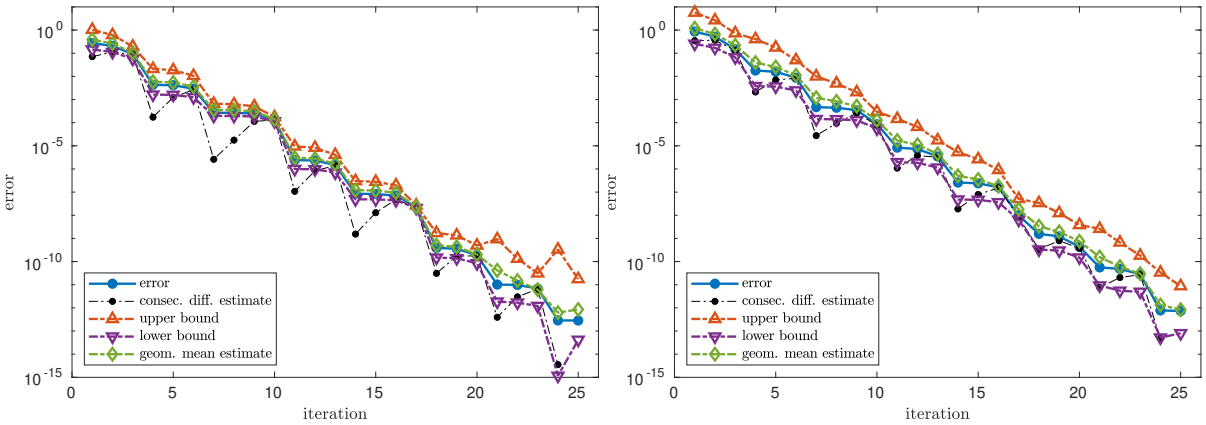


Figure 3.1 – Accuracy of error bounds and estimates for the relative error in the computation of $\boldsymbol{b}^H f(A)\boldsymbol{b}$ with a rational Krylov method, where A is a 2000×2000 matrix with logspaced eigenvalues in the interval $[10^{-3}, 10^3]$. Left: $f(z) = z \log z$. Right: $f(z) = \sqrt{z}$.

Remark 3.2.5. We do not have a rigorous explanation for the accuracy of the estimate based on the geometric mean of the bounds in (3.2.6), but from further experiments it seems to be very accurate also for other functions. Unfortunately, if the spectral interval Σ is only known approximately, the bounds become less tight and the geometric mean estimate usually ends up underestimating the actual error by one or two orders of magnitude.

3.2.2 Computation of the bounds

Recall that the a posteriori bounds in (3.2.6) hold after the m-th iteration only if $\xi_m = \infty$. In a practical scenario, i.e., when using the bounds as a stopping criterion for a rational Krylov method, it is desirable to evaluate the bounds after each iteration, without being forced to set the corresponding pole to ∞. As anticipated in Remark 3.2.1, we provide here two approaches to evaluate the bounds in (3.2.6) even when $\xi_m \neq \infty$.

One way to avoid setting poles to ∞, proposed in [89, Section 6.1], is to use an auxiliary basis vector $\boldsymbol{v}_\infty$, which is initialized as $\boldsymbol{v}_\infty^{(1)} = A\boldsymbol{v}_1$ at the beginning of the rational Arnoldi algorithm, and maintained orthonormal to the basis vectors $\{\boldsymbol{v}_1, \dots, \boldsymbol{v}_j\}$ at each iteration j, at the cost of only one additional orthogonalization per iteration. The basis $[V_j\ \boldsymbol{v}_\infty^{(j)}]$ is an orthonormal basis of the rational Krylov subspace $\mathcal{Q}_{j+1}(A, \boldsymbol{b})$ with poles $\{\xi_1, \dots, \xi_{j-1}, \infty\}$, and it is associated to the auxiliary Arnoldi decomposition

$$AV_j\widetilde{K}_j = [V_j\ \boldsymbol{v}_\infty^{(j)}]\underline{\widetilde{H}}_j,$$

where $\widetilde{K}_j\boldsymbol{e}_j = \boldsymbol{e}_1$ and the last column of $\underline{\widetilde{H}}_j$ contains the orthogonalization coefficients for $\boldsymbol{v}_\infty^{(j)}$. This decomposition can be used to compute the bound (3.2.6) since the last row of $\underline{\widetilde{K}}_j$ is zero by construction.

Remark 3.2.6. We point out that if $\xi_j = \infty$ for some j, then the approach described above will not work from iteration $j+1$ onward, since $A\boldsymbol{v}_1 \in \mathcal{Q}_{j+1}(A, \boldsymbol{b})$ and therefore $\boldsymbol{v}_\infty^{(j+1)} = \boldsymbol{0}$ after orthogonalization. This is easily fixed by switching to a different auxiliary vector at iteration $j+1$, such as $\boldsymbol{v}_\infty = A\boldsymbol{v}_{j+1}$, or by setting directly at the start $\boldsymbol{v}_\infty = A^{\ell+1}\boldsymbol{v}_1$, where ℓ is the number of poles at ∞ used to construct the rational Krylov subspace.

In our experience, the technique described above can be sometimes subject to instability due to a large condition number of the matrix $\widetilde{K}_j$. Therefore we also propose another approach, which is inspired by the methods for moving the poles of a rational Krylov subspace presented in [26, Section 4]. The idea is to add a pole at ∞ at the beginning of the pole sequence, and reorder the poles at each iteration in order to always have the last pole equal to ∞.

First of all, we recall how to swap poles in a rational Arnoldi decomposition. This procedure is a special case of the algorithm described in [26], but we still describe it in some detail for completeness. Recall that the poles of a rational Krylov subspace are the ratios of the entries below the main diagonals of $\underline{H}_j$ and $\underline{K}_j$, i.e., we have $\xi_j = h_{j+1,j}/k_{j+1,j}$. In other words, the poles $\{\xi_1, \dots, \xi_j\}$ are the eigenvalues of the upper triangular pencil $(\widehat{H}_j, \widehat{K}_j)$, where we denote by $\widehat{H}_j$ the bottom $j \times j$ block of $\underline{H}_j$, and similarly for $\widehat{K}_j$. So we can obtain a transformation that swaps two adjacent poles in the same way as orthogonal transformations that reorder eigenvalues in a generalized Schur form [102]. Let U_j and W_j be $j \times j$ unitary matrices such that the pencil $U_j^H(\widehat{H}_j, \widehat{K}_j)W_j$ is still in upper triangular form and has the last two eigenvalues in reversed order; the matrices U_j and W_j only involve 2×2 rotations, and they can be computed and applied cheaply as described in [102]. Defining $\widehat{U}_j = \mathrm{blkdiag}(1, U_j) \in \mathbb{C}^{(j+1)\times(j+1)}$, we have the new rational Arnoldi decomposition

$$A\widetilde{V}_{j+1}\underline{\widetilde{K}}_j = \widetilde{V}_{j+1}\underline{\widetilde{H}}_j,$$

where

$$\widetilde{V}_{j+1} = V_{j+1}\widehat{U}_j, \qquad \underline{\widetilde{K}}_j = \widehat{U}_j^H \underline{K}_j W_j \qquad \text{and} \qquad \underline{\widetilde{H}}_j = \widehat{U}_j^H \underline{H}_j W_j.$$

This decomposition has the same poles as $AV_{j+1}\underline{K}_j = V_{j+1}\underline{H}_j$, with the difference that the last two poles ξ_{j-1} and ξ_j are now swapped. In particular, if $\xi_{j-1} = \infty$, the last pole of the new decomposition is now ∞, and hence the last row of $\underline{\widetilde{K}}_j$ is equal to zero.

Given the pole sequence $\{\xi_1, \xi_2, \dots\}$, let us consider the rational Krylov subspace associated to the modified pole sequence $\{\infty, \xi_1, \xi_2, \dots\}$. Clearly, after the first iteration both pole sequences identify the same subspace $\mathcal{Q}_1(A, \boldsymbol{b}) = \mathrm{span}\{\boldsymbol{b}\}$, but the last (and first) pole of the modified sequence is ∞, so the last row of $\underline{K}_1$ is zero and we can use the decomposition $AV_1K_1 = V_2\underline{H}_1$ to compute the bound (3.2.6). After the second iteration, if $\xi_1 \neq \infty$, we can swap the poles ξ_1 and ∞ with the procedure outlined above to obtain the decomposition $AV_2K_2 = V_3\underline{H}_2$ associated to the poles $\{\xi_1, \infty\}$, where again the last row of $\underline{K}_2$ is equal to zero (for simplicity we still use the notation K_j instead of $\widetilde{K}_j$, and similarly for V_j and H_j). If $\xi_1 = \infty$, there is no need to swap poles and we can proceed to the next iteration.

By repeating the same procedure to swap the last two poles at each iteration, we can ensure that after j iterations we have a decomposition $AV_jK_j = V_{j+1}\underline{H}_j$, associated to the poles $\{\xi_1, \dots, \xi_{j-1}, \infty\}$ in this order, so that the last row of $\underline{K}_j$ is equal to zero and it can be used to compute the bound (3.2.6).

Remark 3.2.7. Note that V_j is a basis of $\mathcal{Q}_j(A, \boldsymbol{b})$, which is the same subspace that we would have obtained if we had run the rational Arnoldi algorithm with poles $\{\xi_1, \dots, \xi_{j-1}\}$; so the method described in this section actually computes the approximation ψ_j and the bound associated to the poles $\{\xi_1, \dots, \xi_{j-1}\}$, and not to the modified pole sequence $\{\infty, \xi_1, \dots, \xi_{j-2}\}$. The initial pole at ∞ is only added to enable the computation of the bound and it is never used in the actual approximation.

3.3 A priori and a posteriori bounds via an integral expression

The authors of [40] analyze the error of the Lanczos method for functions of Hermitian matrices, and by means of the Cauchy integral formula they derive an expression for the error that can be used to obtain both a priori and a posteriori error bounds. In this section we follow a similar approach to obtain an integral expression for the error in the approximation of $f(A)\boldsymbol{b}$ by means of a rational Krylov method, and we then use it to derive both a priori and a posteriori bounds in the case of a Hermitian matrix A, for functions that have a certain integral representation. The results that we obtain can be interpreted as a generalization to the rational Krylov case of the ones presented in [40]. The contents of this section are primarily based on [143].

Given an open set $\mathcal{S} \subset \mathbb{C}$, we consider the class of functions $f : \mathcal{S} \to \mathbb{C}$ that admit the integral expression

$$f(x) = \int_\Gamma (x - z)^{-1} \, \mathrm{d}\mu(z), \tag{3.3.1}$$

for a suitable contour $\Gamma \subset \mathbb{C}$ and measure $\mathrm{d}\mu(z)$. The representation (3.3.1) reduces to the Cauchy integral formula when f is analytic, $\mathrm{d}\mu(z) = -\dfrac{1}{2\pi i} f(z)\,\mathrm{d}z$, Γ is a simple closed curve and $\mathcal{S}$ is the

interior of Γ, and to the Cauchy-Stieltjes integral representation when $\Gamma = (-\infty, 0]$, $\mathcal{S} = \mathbb{C}\backslash\Gamma$ and $\mathrm{d}\mu(z)$ is a positive measure (see Definition 2.2.2). Rational approximations of Cauchy-Stieltjes (and more generally, Markov) functions have been investigated for instance in [8, 10].

Similarly to (3.2.1), by using the matrix version of the integral representation (3.3.1) of f we can write the error in the rational Krylov approximation of $f(A)\boldsymbol{b}$ in the form

$$f(A)\boldsymbol{b} - \boldsymbol{f}_m = \int_\Gamma \left((A - zI)^{-1}\boldsymbol{b} - V_m(A_m - zI)^{-1}V_m^H\boldsymbol{b}\right)\mathrm{d}\mu(z) = \int_\Gamma \mathtt{err}_m(z)\,\mathrm{d}\mu(z), \tag{3.3.2}$$

where $\mathtt{err}_m(z)$ denotes the error in the solution of the shifted linear system $(A - zI)\boldsymbol{x} = \boldsymbol{b}$ using the rational Krylov subspace $\mathcal{Q}_m(A, \boldsymbol{b})$, see (3.1.4).

Following a similar approach as the one used in [40] to bound the error for the Lanczos method, we are going to rewrite (3.3.2) in terms of $\mathtt{res}_m(w)$ and $\mathtt{err}_m(w)$, for a given $w \in \mathbb{C}$. For w and $z \in \mathbb{C}$, it is easy to derive from (3.1.11) the identity

$$\mathtt{res}_m(z) = \frac{q_{m-1}(z)}{q_{m-1}(w)} \cdot \frac{\chi_m(w)}{\chi_m(z)}\,\mathtt{res}_m(w) = \frac{q_{m-1}(z)}{q_{m-1}(w)}\det(h_{w,z}(A_m))\,\mathtt{res}_m(w), \tag{3.3.3}$$

where we used the notation $h_{w,z}(t) := \dfrac{t - w}{t - z}$, borrowed from [40]. In the following, we are also going to use the notation $k_z(t) := (t - z)^{-1}$.

We can use the identity (3.3.3) to replace $\mathtt{res}_m(z)$ with $\mathtt{res}_m(w)$ for any w and $z \in \mathbb{C}$. A similar relation can be obtained for $\mathtt{err}_m(z)$ and $\mathtt{err}_m(w)$, indeed we have

$$\begin{aligned}\mathtt{err}_m(z) &= (A - zI)^{-1}\mathtt{res}_m(z)\\ &= \frac{q_{m-1}(z)}{q_{m-1}(w)} \cdot \frac{\chi_m(w)}{\chi_m(z)}(A - zI)^{-1}\mathtt{res}_m(w)\\ &= \frac{q_{m-1}(z)}{q_{m-1}(w)}\det\left(h_{w,z}(A_m)\right)h_{w,z}(A)\,\mathtt{err}_m(w).\end{aligned} \tag{3.3.4}$$

By comparing (3.3.3) with (3.1.5) we also obtain the following explicit expression for the scalar coefficient in (3.3.3),

$$\frac{q_{m-1}(z)}{q_{m-1}(w)} \cdot \frac{\chi_m(w)}{\chi_m(z)} = \frac{\boldsymbol{e}_m^T K_m^{-1}(A_m - zI)^{-1}\boldsymbol{e}_1}{\boldsymbol{e}_m^T K_m^{-1}(A_m - wI)^{-1}\boldsymbol{e}_1}.$$

The identities (3.3.3) and (3.3.4) can be used to prove the following theorem, which gives us an expression for the error in the approximation of $f(A)\boldsymbol{b}$ in terms of the error or residual of shifted linear systems.

Theorem 3.3.1. Assume that f has the integral representation (3.3.1), and let $\boldsymbol{f}_m$ be the approximation (2.5.6) to $f(A)\boldsymbol{b}$ from the rational Krylov subspace $\mathcal{Q}_m(A, \boldsymbol{b}, \boldsymbol{\xi}_m)$, with denominator polynomial q_{m-1}. Let $\mathcal{D} = \{\xi_1, \ldots, \xi_{m-1}\} \cup \Lambda(A) \cup \Lambda(A_m)$ and let w be a measurable function $w : \Gamma \to \mathbb{C} \setminus \mathcal{D}$. We have

$$\begin{aligned}f(A)\boldsymbol{b} - \boldsymbol{f}_m &= \int_\Gamma \frac{q_{m-1}(z)}{q_{m-1}(w(z))}\det\left(h_{w(z),z}(A_m)\right)h_{w(z),z}(A)\,\mathtt{err}_m(w(z))\,\mathrm{d}\mu(z)\\ &= \int_\Gamma \frac{q_{m-1}(z)}{q_{m-1}(w(z))}\det\left(h_{w(z),z}(A_m)\right)k_z(A)\,\mathtt{res}_m(w(z))\,\mathrm{d}\mu(z).\end{aligned} \tag{3.3.5}$$

Proof. In (3.3.2), for each $z \in \Gamma$ replace $\mathtt{err}_m(z)$ with the expression in (3.3.4) using $w = w(z)$ to obtain the first identity in (3.3.5). The second identity is then easily obtained by recalling the definition of k_z. □

Remark 3.3.2. An identity similar to the one in Theorem 3.3.1 has been given in [40, Corollary 2.5] for the error of the approximation of $f(A)\boldsymbol{b}$ with the Lanczos method, using the Cauchy integral representation of f. Indeed, Theorem 3.3.1 reduces to [40, Corollary 2.5] by taking $q_{m-1}(z) \equiv 1$, $\mathrm{d}\mu(z) = -\dfrac{1}{2\pi i}f(z)\,\mathrm{d}z$ and a constant $w(z) \equiv w$. Apart from the factors that involve the denominator q_{m-1} of

the rational Krylov subspace, an important difference from the result in [40] is the use of a function $w(z)$ instead of a constant parameter w. This modification, which might appear minor at a quick glance, turns out to be a crucial element for dealing with the rational Krylov case. We are going to thoroughly discuss this point in Section 3.3.1.

Although it is not practically feasible to evaluate the error expression in (3.3.5) directly since it depends explicitly on the matrix A, it can be employed to derive bounds for the error that are easily computable. The following corollary is a simple consequence of Theorem 3.3.1, and it is the analogue of [40, Theorem 2.6] for rational Krylov instead of Lanczos. We denote the eigenvalues of A_m by $\lambda_i(A_m)$, for $i = 1, \dots, m$.

Corollary 3.3.3. With the same hypothesis of Theorem 3.3.1, assume further that A is Hermitian and that for some compact sets $\mathcal{S}_0, \mathcal{S}_1, \dots, \mathcal{S}_m \subset \mathbb{R} \setminus \Gamma$ we have $\Lambda(A) \subset \mathcal{S}_0$, and $\lambda_i(A_m) \in \mathcal{S}_i$ for all $i = 1, \dots, m$. Then we have the inequalities

$$\|f(A)\boldsymbol{b} - \boldsymbol{f}_m\|_2 \leq \int_\Gamma \left| \frac{q_{m-1}(z)}{q_{m-1}(w(z))} \right| \prod_{j=0}^{m} \|h_{w(z),z}\|_{\mathcal{S}_j} \cdot \|\mathtt{err}_m(w(z))\|_2 \, |\mathrm{d}\mu(z)|$$

and

$$\|f(A)\boldsymbol{b} - \boldsymbol{f}_m\|_2 \leq \int_\Gamma \left| \frac{q_{m-1}(z)}{q_{m-1}(w(z))} \right| \prod_{j=1}^{m} \|h_{w(z),z}\|_{\mathcal{S}_j} \cdot \|k_z\|_{\mathcal{S}_0} \cdot \|\mathtt{res}_m(w(z))\|_2 \, |\mathrm{d}\mu(z)|.$$

Proof. Since the eigenvalues of A are contained in $\mathcal{S}_0$, we have

$$\|h_{w(z),z}(A)\|_2 = \max_{\lambda \in \Lambda(A)} |h_{w(z),z}(\lambda)| \leq \|h_{w(z),z}\|_{\mathcal{S}_0}.$$

Similarly, we have

$$\left|\det\left(h_{w(z),z}(A_m)\right)\right| = \prod_{j=1}^{m} |h_{w(z),z}(\lambda_j(A_m))| \leq \prod_{j=1}^{m} \|h_{w(z),z}\|_{\mathcal{S}_j}.$$

By applying the above inequalities to the first identity in Theorem 3.3.1, we obtain

$$\begin{aligned}
\|f(A)\boldsymbol{b} - \boldsymbol{f}_m\|_2 &\leq \int_\Gamma \left| \frac{q_{m-1}(z)}{q_{m-1}(w(z))} \right| \cdot |\det(h_{w(z),z}(A_m))| \cdot \|h_{w(z),z}(A)\|_2 \|\mathtt{err}_m(w(z))\|_2 \, |\mathrm{d}\mu(z)| \\
&\leq \int_\Gamma \left| \frac{q_{m-1}(z)}{q_{m-1}(w(z))} \right| \cdot \|h_{w(z),z}\|_{\mathcal{S}_0} \cdot \prod_{j=1}^{m} \|h_{w(z),z}\|_{\mathcal{S}_j} \cdot \|\mathtt{err}_m(w(z))\|_2 \, |\mathrm{d}\mu(z)|.
\end{aligned}$$

The second inequality follows with the same argument, using the error expression in terms of $\mathtt{res}_m(w(z))$ from Theorem 3.3.1. □

Observe that if we use a constant function $w(z) \equiv w$ in Corollary 3.3.3 the bounds are simplified, since the terms $\|\mathtt{err}_m(w)\|_2$ and $\|\mathtt{res}_m(w)\|_2$ no longer depend on z and thus can be bounded independently from the integral term.

Corollary 3.3.3 can be interpreted as either an a priori or an a posteriori bound, depending on the choices for the sets $\mathcal{S}_j$. For example, we can easily obtain an a priori bound by taking $\mathcal{S}_j = [a, b] \supset \Lambda(A)$ for $j = 0, 1, \dots, m$. On the other hand, once A_m has been computed and its eigenvalues are known, we can obtain an a posteriori bound by taking $\mathcal{S}_0 = [a, b]$ and $\mathcal{S}_j = \{\lambda_j(A_m)\}$, for $j = 1, \dots, m$. The two following corollaries are restatements of Corollary 3.3.3 in the a priori and a posteriori setting, respectively, using the choices described above for the sets $\mathcal{S}_j$, $j = 0, \dots, m$.

Corollary 3.3.4. Under the assumptions of Theorem 3.3.1, if A is Hermitian with $\Lambda(A) \subset [a, b]$ contained in the interior of Γ, we have the a priori bounds

$$\|f(A)\boldsymbol{b} - \boldsymbol{f}_m\|_2 \leq \int_\Gamma \left| \frac{q_{m-1}(z)}{q_{m-1}(w(z))} \right| \cdot \|h_{w(z),z}\|_{[a,b]}^{m+1} \cdot \|\mathtt{err}_m(w(z))\|_2 \, |\mathrm{d}\mu(z)|$$

and

$$\|f(A)\boldsymbol{b} - \boldsymbol{f}_m\|_2 \leq \int_\Gamma \left| \frac{q_{m-1}(z)}{q_{m-1}(w(z))} \right| \cdot \|h_{w(z),z}\|_{[a,b]}^{m} \cdot \|k_z\|_{[a,b]} \cdot \|\mathtt{res}_m(w(z))\|_2 \, |\mathrm{d}\mu(z)|.$$

Corollary 3.3.5. Under the assumptions of Theorem 3.3.1, if A is Hermitian with $\Lambda(A) \subset [a,b]$ contained in the interior of Γ, we have the a posteriori bounds

$$\|f(A)\boldsymbol{b} - \boldsymbol{f}_m\|_2 \leq \int_\Gamma \left| \frac{q_{m-1}(z)}{q_{m-1}(w(z))} \cdot \frac{\chi_m(w(z))}{\chi_m(z)} \right| \cdot \|h_{w(z),z}\|_{[a,b]} \cdot \|\mathtt{err}_m(w(z))\|_2 \, |\mathrm{d}\mu(z)|$$

and

$$\|f(A)\boldsymbol{b} - \boldsymbol{f}_m\|_2 \leq \int_\Gamma \left| \frac{q_{m-1}(z)}{q_{m-1}(w(z))} \cdot \frac{\chi_m(w(z))}{\chi_m(z)} \right| \cdot \|k_z\|_{[a,b]} \cdot \|\mathtt{res}_m(w(z))\|_2 \, |\mathrm{d}\mu(z)|.$$

If additional a priori information on the spectrum of A is available, it can be incorporated in the sets $\mathcal{S}_j$ in order to get more precise bounds. For example, assume that in addition to the spectral interval $[a,b]$ we know that all except for 10 eigenvalues of A are contained in the smaller interval $[a,c]$, with $a < c < b$. Then, by the Cauchy interlacing theorem [88, Theorem 8.1.7] we can conclude that at most 10 eigenvalues of A_m are not inside the interval $[a,c]$, so we can take $\mathcal{S}_j = [a,b]$ for $j = 1,\dots,10$ and $\mathcal{S}_j = [a,c]$ for $j = 11,\dots,m$. This kind of choice for the sets $\mathcal{S}_j$ exploits more spectral information of the matrix compared to the bounds in Corollary 3.3.4, so it would result in more refined bounds that are able to better capture the convergence behavior of rational Krylov methods.

Remark 3.3.6. Although the error expressions in Theorem 3.3.1 hold for a general matrix A, we have assumed that A is Hermitian to obtain Corollary 3.3.3. While it is possible in principle to obtain bounds for non-Hermitian matrices by bounding the factors in (3.3.5) in terms of the field of values of A, such a bound would not be particularly useful in practice, since it is difficult to determine the field of values of a general matrix. Moreover, the optimization approach described in Sections 3.3.1 and 3.3.2 may fail to produce accurate bounds in the non-Hermitian case.

3.3.1 Bound discussion

The error bounds in Corollaries 3.3.4 and 3.3.5 depend on several parameters, such as the curve Γ and the function $w(z)$, and on the choice between $\|\mathtt{err}_m(w(z))\|_2$ and $\|\mathtt{res}_m(w(z))\|_2$.

Since the factors $\|h_{w,z}\|_{[a,b]}$ and $\|k_z\|_{[a,b]}$ diverge when z approaches the interval $[a,b]$, when using the Cauchy integral formula it is a good idea to take Γ as far as possible from the spectrum of A. For example, for a function such as $f(z) = \sqrt{z}$ with a growth behavior of $o(|z|)$ for $|z| \to \infty$, we can choose Γ as a large circular arc centered in the origin joined with a keyhole contour around the negative real line to avoid the branch cut of f. Since $|f(z)/z| = o(1)$ as $|z| \to \infty$, the integral on the large circular arc in (3.3.5) vanishes as the radius goes to infinity, so by taking the limit we can consider the contour $\Gamma = (-\infty, 0]$. See [40, Section 3.1] for further discussion on the choice of Γ.

Regarding the choice between the bound in terms of $\|\mathtt{err}_m(w(z))\|_2$ and the one in terms of $\|\mathtt{res}_m(w(z))\|_2$, it is clear that for an a posteriori bound it makes more sense to use the residual norm, since it can be computed quite cheaply once the Krylov basis is available, while the formulation with the error norm requires the exact solution of the linear system, which is significantly more expensive to obtain and therefore impractical.

On the other hand, for an a priori bound neither $\mathtt{res}_m(w(z))$ nor $\mathtt{err}_m(w(z))$ are available because the Krylov basis has not been constructed yet, so in a realistic scenario they have to be bounded using an a priori bound for the linear system error or residual. Which formulation gives the better bound would depend on the quality of the bound for the linear system error or residual, but numerical experiments seem to indicate that the formulation based on the error is usually more accurate.

The same can be observed for a posteriori bounds as well: the bound in Corollary 3.3.5 with the exact error is sharper than the bound with the exact residual. A possible explanation for this fact is that the inequality

$$\|h_{w,z}(A)\mathtt{err}_m(w)\|_2 \leq \|h_{w,z}\|_{[a,b]} \cdot \|\mathtt{err}_m(w)\|_2$$

is tighter than

$$\|(A - zI)^{-1}\mathtt{res}_m(w)\|_2 \leq \|k_z\|_{[a,b]} \cdot \|\mathtt{res}_m(w)\|_2,$$

especially when w can be freely chosen (note that the left-hand sides in the two inequalities above are equal). However, this observation is only meaningful for a priori bounds, because in practice one

would always use the residual formulation for an a posteriori bound, since the exact residual norm is cheap to compute when the rational Krylov basis is available. We will discuss the computation of the a posteriori bound in detail in Section 3.3.3.

A crucial factor for the effective use of the bounds in Corollaries 3.3.4 and 3.3.5 is the choice of the function $w(z)$. The simplest option (and the cheapest one to evaluate) is to use a constant function $w(z) \equiv w$, similarly to [40], since with this choice we only have to compute or bound the residual (or error) of a single shifted linear system. This turns out to be the most convenient choice for a posteriori bounds, but it often gives underwhelming results when used to obtain a priori bounds, and it may even cause the bounds to diverge (see Section 3.4.1 for an example); in this setting, using a general function $w(z)$ is much more effective.

To have a better understanding of this phenomenon, let us focus on the first bound in Corollary 3.3.4, and let us look at each factor in the integrand separately:

- the factor $|q_{m-1}(z)/q_{m-1}(w)| = \prod_{j=1}^{m-1} |(z-\xi_j)/(w-\xi_j)|$ is large when w is closer to the set of poles $\{\xi_j\}_j$ compared to z, and small when w is farther than z from the poles;
- the factor $\|h_{w,z}\|_{[a,b]}^{m+1}$ is small when w is closer than z to the spectral interval $[a,b]$, and large otherwise;
- the factor $\|\mathtt{err}_m(w)\|_2$ is large when w is close to $[a,b]$, and small when w is close to the poles or when w has a large modulus.

A consequence of the different behavior of the factors and their dependence on z is that the value of the parameter w that minimizes the integrand can be significantly different for different values of z, so it is difficult to find a single w that makes the integrand small for all $z \in \Gamma$. This is illustrated in Figures 3.2 and 3.3 with a simple example where the spectrum of the matrix A is contained in $[0.2, 4]$ and the rational Krylov subspace has four poles, for $z = -4$ and $z = -0.3$ and w that varies in the complex square $[-5, 5] \times [-5i, 5i]$. Note that the poles factors in Figures 3.2 and 3.3 only differ by a constant multiplicative factor, and that the linear system error factors are exactly the same. However, due to the different behavior of the $h_{w,z}$ factor in the two cases, the full integrands (on the bottom right in Figures 3.2 and 3.3) look significantly different.

The values of $\|h_{w,z}\|_{[a,b]}$ in Figures 3.2 and 3.3 are computed using Lemma 3.3.7, which generalizes [40, Lemma 3.1] to the case of complex w and real z. We mention that this result can also be generalized to the case of both w and z complex with a similar proof, but the statement is less elegant in that case.

Lemma 3.3.7. Let $[a,b] \subset \mathbb{R}$, $z \in \mathbb{R} \setminus [a,b]$ and $w \in \mathbb{C}$. Define

$$\lambda^\star = \frac{|w|^2 - z\,\mathrm{Re}(w)}{\mathrm{Re}(w) - z}$$

and let

$$h^\star = \begin{cases} |h_{w,z}(\lambda^\star)| = \left|\dfrac{\mathrm{Im}(w)}{w-z}\right| & \text{if } \lambda^\star \in [a,b], \\ 0 & \text{otherwise.} \end{cases}$$

We have

$$\|h_{w,z}\|_{[a,b]} = \max\left\{ \left|\frac{a-w}{a-z}\right|, \left|\frac{b-w}{b-z}\right|, h^\star \right\}.$$

Proof. The proof is similar to the proof of [40, Lemma 3.1]. For any $\lambda \in \mathbb{R}$, we have

$$H(\lambda) := |h_{w,z}(\lambda)|^2 = \left|\frac{\lambda - w}{\lambda - z}\right|^2 = \frac{(\lambda - \mathrm{Re}(w))^2 + \mathrm{Im}(w)^2}{(\lambda - z)^2},$$

and

$$\frac{\mathrm{d}H}{\mathrm{d}\lambda} = \frac{2(\lambda - \mathrm{Re}(w))(\lambda - z)^2 - 2(\lambda - z)((\lambda - \mathrm{Re}(w))^2 + \mathrm{Im}(w)^2)}{(\lambda - z)^4}.$$

It follows that $\frac{\mathrm{d}H}{\mathrm{d}\lambda} = 0$ only for $\lambda = \lambda^\star$, so the only possible maxima of $h_{w,z}(\lambda)$ in $[a,b]$ are for $\lambda = a$, $\lambda = b$ and $\lambda = \lambda^\star$, if $\lambda^\star \in [a,b]$. Finally, with some simple algebraic manipulations it can be shown that $|h_{w,z}(\lambda^\star)| = |\mathrm{Im}(w)/(w-z)|$. □

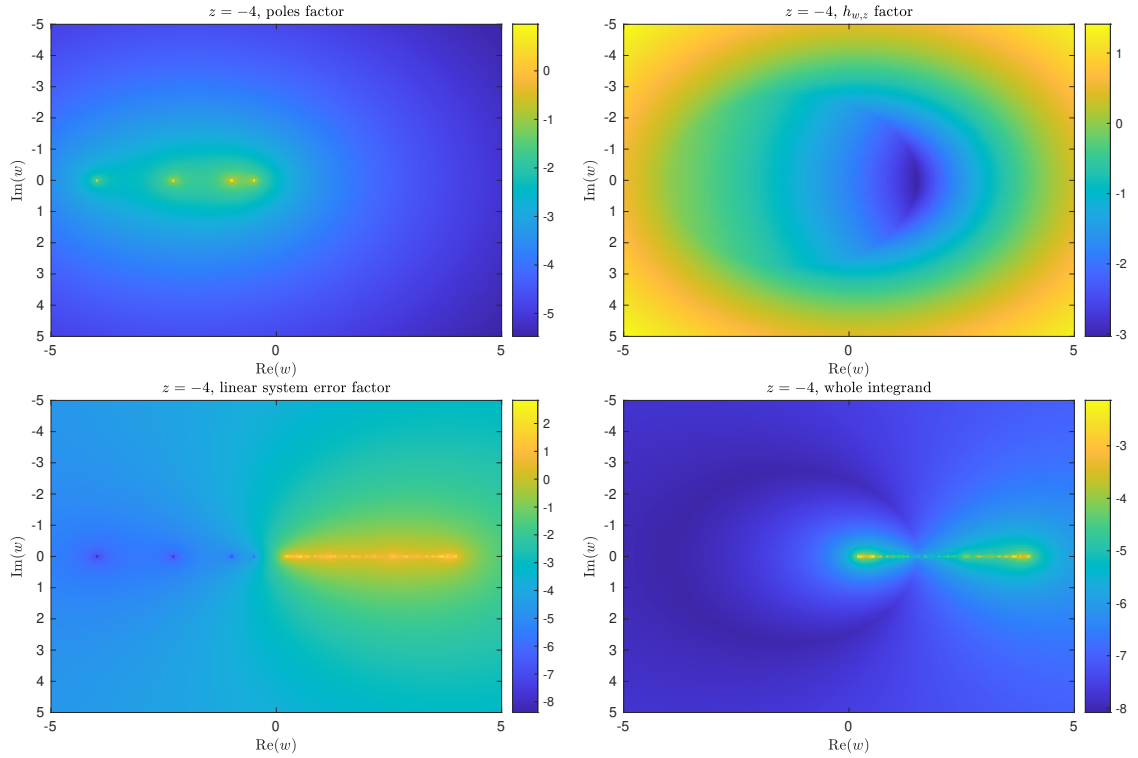


Figure 3.2 – Behavior of the different factors in the first a priori bound in Corollary 3.3.4 for $w \in [-5, 5] \times [-5i, 5i]$, for $z = -4$.

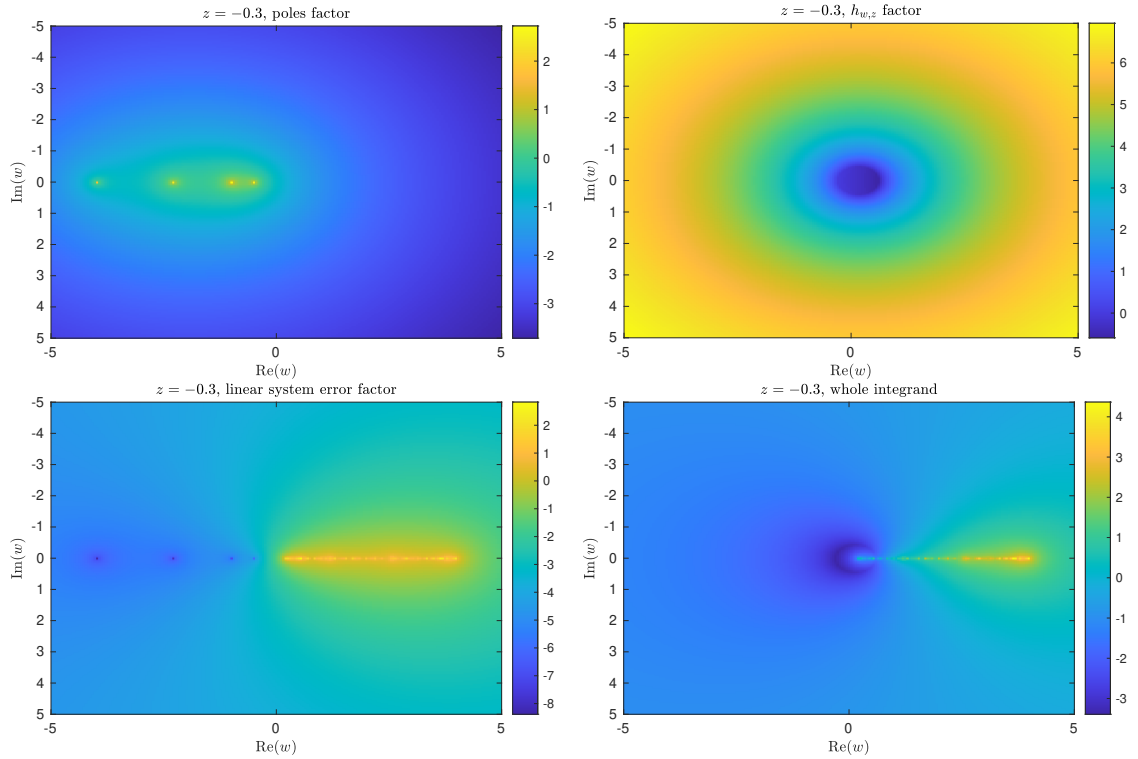


Figure 3.3 – Behavior of the different factors in the first a priori bound in Corollary 3.3.4 for $w \in [-5, 5] \times [-5i, 5i]$, for $z = -0.3$.

We can see from Figures 3.2 and 3.3 that the values of w for which the integrand is smallest are close to z. Indeed, it turns out that the bounds in Corollary 3.3.3 are minimized precisely when $w(z) = z$.

Proposition 3.3.8. Under the same assumptions of Corollary 3.3.3, the bounds in Corollary 3.3.3 are minimized for $w(z) = z$.

Proof. Let us consider the first bound in Corollary 3.3.3. For $w(z) = z$, we have $h_{z,z}(t) = 1$ for all $t \neq z$, so the bound becomes simply

$$\|f(A)\boldsymbol{b} - \boldsymbol{f}_m\|_2 \leq \int_\Gamma \|\mathtt{err}_m(z)\|_2 \, |\mathrm{d}\mu(z)|.$$

Now, consider any function $w(z)$ that satisfies the assumptions in Theorem 3.3.1 and recall (3.3.4), which implies that

$$\|\mathtt{err}_m(z)\|_2 = \left| \frac{q_{m-1}(z)}{q_{m-1}(w(z))} \right| \cdot |\det(h_{w(z),z}(A_m))| \cdot \|h_{w(z),z}(A)\, \mathtt{err}_m(w(z))\|_2.$$

Using the assumptions of Corollary 3.3.3, we can bound

$$|\det(h_{w(z),z}(A_m))| \leq \prod_{j=1}^{m} \|h_{w(z),z}\|_{\mathcal{S}_j}$$

and

$$\|h_{w(z),z}(A)\mathtt{err}_m(w(z))\|_2 \leq \|h_{w(z),z}\|_{\mathcal{S}_0} \cdot \|\mathtt{err}_m(w(z))\|_2.$$

If we plug these inequalities in the expression for $\|\mathtt{err}_m(z)\|_2$, we obtain

$$\begin{aligned}\|f(A)\boldsymbol{b} - \boldsymbol{f}_m\|_2 &\leq \int_\Gamma \|\mathtt{err}_m(z)\|_2 \, |\mathrm{d}\mu(z)| \\ &\leq \int_\Gamma \left| \frac{q_{m-1}(z)}{q_{m-1}(w(z))} \right| \cdot \prod_{j=1}^{m} \|h_{w(z),z}\|_{\mathcal{S}_j} \cdot \|h_{w(z),z}\|_{\mathcal{S}_0} \cdot \|\mathtt{err}_m(w(z))\|_2 \, |\mathrm{d}\mu(z)|,\end{aligned}$$

which coincides with the first bound in Corollary 3.3.3. Since this inequality holds for any admissible function $w(z)$, we have shown that the bound with $w(z) = z$ is smaller than the bound obtained using any other function $w(z)$, and thus the function $w(z) = z$ must be a minimizer for the right-hand side of the first bound in Corollary 3.3.3.

The same can be proved for the second bound in Corollary 3.3.3, by observing that for $w(z) = z$ the bound simplifies to

$$\|f(A)\boldsymbol{b} - \boldsymbol{f}_m\|_2 \leq \int_\Gamma \|k_z\|_{\mathcal{S}_0} \cdot \|\mathtt{res}(z)\|_2 \, |\mathrm{d}\mu(z)|$$

and then using the same strategy, recalling the identity

$$\|\mathtt{res}(z)\|_2 = \left| \frac{q_{m-1}(z)}{q_{m-1}(w(z))} \right| \cdot |\det(h_{w(z),z}(A_m))| \cdot \|\mathtt{res}(w(z))\|_2. \qquad \square$$

Although using $w(z) = z$ is not very practical because it would require the computation of the residual or error norm of a different shifted linear system for each point used in the discretization of the integral, the result of Proposition 3.3.8 provides some theoretical insight regarding the choice of the function $w(z)$. In the following sections we discuss practical ways to choose the function $w(z)$ in order to approximately minimize the bounds, both in the a priori and a posteriori setting.

3.3.2 A priori bounds

Let us consider the first a priori bound in Corollary 3.3.4. For any $z \in \Gamma$, an ideal choice for $w(z)$ would be one that minimizes the integrand in the bound, i.e.,

$$w(z) = \operatorname*{argmin}_{w \in \mathbb{C} \setminus \mathcal{D}} \left(\frac{1}{|q_{m-1}(w)|} \cdot \|h_{w,z}\|_{[a,b]}^{m+1} \cdot \|\mathtt{err}_m(w)\|_2 \right),$$

Algorithm 3.1 A priori error bound for $f(A)\boldsymbol{b}$

Input: Hermitian matrix $A \in \mathbb{C}^{n\times n}$, vector $\boldsymbol{b} \in \mathbb{C}^n$, real interval $[a,b]$ such that $\Lambda(A) \subset [a,b]$, function f with integral representation (3.3.1), poles $\{\xi_1, \dots, \xi_{m-1}\} \subset \mathbb{C}$.

Output: a priori bound $\varepsilon_m > 0$ such that $\|f(A)\boldsymbol{b} - \boldsymbol{f}_m\|_2 \le \varepsilon_m$, with $\boldsymbol{f}_m$ defined in (2.5.6) using poles $\{\xi_1, \dots \xi_{m-1}\}$.

1: Choose a discrete set $\mathcal{W} \subset \mathbb{C} \setminus [a,b] \cup \{\xi_1, \dots, \xi_{m-1}\}$ (e.g., take $\mathcal{W}$ to be a discretization of Γ).

2: For all $w \in \mathcal{W}$, compute $\varphi_m(w)$ such that $\|\mathtt{err}_m(w)\| \le \varphi_m(w)$ (e.g., using Theorem 3.3.14).

3: Choose a quadrature formula for the integral over Γ, with quadrature nodes $\mathcal{Z} \subset \Gamma$.

4: For all $z \in \mathcal{Z}$, compute $w(z) = \operatorname*{argmin}_{w\in\mathcal{W}} \left(\dfrac{1}{|q_{m-1}(w)|} \cdot \|h_{w,z}\|_{[a,b]}^{m+1} \cdot \varphi_m(w) \right)$, using [40, Lemma 3.1] or Lemma 3.3.7 to evaluate $\|h_{w,z}\|_{[a,b]}$.

5: Using the chosen quadrature formula, compute

$$\varepsilon_m \approx \int_\Gamma \left| \frac{q_{m-1}(z)}{q_{m-1}(w(z))} \right| \cdot \|h_{w(z),z}\|_{[a,b]}^{m+1} \cdot \varphi_m(w(z)) \, |\mathrm{d}\mu(z)|.$$

where we have dropped the factors that do not depend on w, and the set $\mathcal{D}$ is the one defined in Theorem 3.3.1. We have seen in Proposition 3.3.8 that the bound is minimized by taking $w(z) = z$. However, in practice we do not have access to $\|\mathtt{err}_m(w)\|_2$ a priori, so we must instead rely on an a priori bound for the linear system error of the form $\|\mathtt{err}_m(w)\|_2 \le \varphi_m(w)$, and then select

$$w(z) = \operatorname*{argmin}_{w\in\mathbb{C}\setminus\mathcal{D}} \left(\frac{1}{|q_{m-1}(w)|} \cdot \|h_{w,z}\|_{[a,b]}^{m+1} \cdot \varphi_m(w) \right). \tag{3.3.6}$$

In general, the choice $w(z) = z$ is not going to be the minimizer of (3.3.6), but we may heuristically expect that choosing $w(z)$ near z gets us close to the optimum. The optimization problem can be approximately solved numerically by replacing $\mathbb{C} \setminus \mathcal{D}$ with a finite set $\mathcal{W}$ of candidates for w, and by finding for each z the value of $w \in \mathcal{W}$ that minimizes the function on the right-hand side. The a priori bound is summed up in Algorithm 3.1. An a priori bound for the linear system error norm is given in Section 3.3.5 below.

Note that in a practical implementation of the bounds, the integral will be approximately evaluated using a quadrature formula, so the number of values of $z \in \Gamma$ for which we have to find the best $w \in \mathcal{W}$ will be finite. For each quadrature node z, the minimization over the set $\mathcal{W}$ requires the computation of $\|h_{w,z}\|_{[a,b]}$ for all $w \in \mathcal{W}$, which can be done efficiently using [40, Lemma 3.1] if w is real, or with Lemma 3.3.7 if z is real. Additionally, for each $w \in \mathcal{W}$ we have to compute $q_{m-1}(w)$ and $\varphi_m(w)$. Although these computations may significantly increase the cost of quadrature if the cardinality of $\mathcal{W}$ is large, it is important to note that their cost is independent of the matrix dimension n, and therefore it eventually becomes negligible compared to the cost of constructing the Krylov basis as the size of A increases.

A possible choice for the set $\mathcal{W}$ is a discretization of the integration contour Γ, which ensures that for each $z \in \Gamma$ there exists a $w \in \mathcal{W}$ that is quite close. Alternatively, one could decide to use a different set $\mathcal{W}_z$ for each $z \in \Gamma$, for instance by taking a small number of points in the neighborhood of z, or even a single point, such as the $w \in \mathcal{W}$ that is closest to z. However, this kind of choice heavily relies on the heuristic that the optimum of (3.3.6) is attained for w close to z, which is not always true. For example, this heuristic may fail when the upper bound $\|\mathtt{err}_m(w)\|_2 \le \varphi_m(w)$ significantly overestimates the error norm only for certain values of w.

3.3.3 A posteriori bounds

Let us consider now the a posteriori bounds. We focus on the second bound in Corollary 3.3.5, since the formulation with the linear system residual is the most likely to be useful in practice. In this case, the best choice for the function $w(z)$ would be one that minimizes the integrand in the bound, i.e.,

$$w(z) = \operatorname*{argmin}_{w\in\mathbb{C}\setminus\mathcal{D}} \frac{|\chi_m(w)|}{|q_{m-1}(w)|} \cdot \|\mathtt{res}_m(w)\|_2,$$

Algorithm 3.2 A posteriori error bound for $f(A)\boldsymbol{b}$

Input: Hermitian matrix $A \in \mathbb{C}^{n\times n}$, vector $\boldsymbol{b} \in \mathbb{C}^n$, real interval $[a,b]$ such that $\Lambda(A) \subset [a,b]$, function f with integral representation (3.3.1), V_m orthonormal basis of $\mathcal{Q}_m(A,\boldsymbol{b},\boldsymbol{\xi}_m)$ with poles $\{\xi_1,\dots,\xi_{m-1}\} \subset \mathbb{C}$, $A_m = V_m^H A V_m$.

Output: a posteriori bound $\varepsilon_m > 0$ such that $\|f(A)\boldsymbol{b} - \boldsymbol{f}_m\|_2 \le \varepsilon_m$, with $\boldsymbol{f}_m$ defined in (2.5.6) using poles $\{\xi_1,\dots\xi_{m-1}\}$.

1: Choose any $w \in \mathbb{C} \setminus [a,b] \cup \{\xi_1,\dots,\xi_{m-1}\}$.

2: Compute $\|\mathtt{res}_m(w)\|_2 = \boldsymbol{b} - (A - wI)V_m(A_m - wI)^{-1}\boldsymbol{e}_1\|\boldsymbol{b}\|_2$.

3: Choose a quadrature formula for the integral over Γ, and use it compute

$$\varepsilon_m \approx \int_\Gamma \left| \frac{q_{m-1}(z)}{q_{m-1}(w)} \cdot \frac{\chi_m(w)}{\chi_m(z)} \right| \cdot \|k_z\|_{[a,b]} \cdot \|\mathtt{res}_m(w)\|_2 \, |\mathrm{d}\mu(z)|.$$

where again we have only included the terms that depend on w. However, it follows from (3.3.3) that the quantity $\mathtt{res}_m(w) \cdot \chi_m(w)\, q_{m-1}(w)^{-1}$ is constant in w, so any value of w can be used to evaluate the bound and would give the same result obtained by taking $w(z) = z$, which minimizes the bound by Proposition 3.3.8. The most efficient choice is then to use a single w for all $z \in \Gamma$. Note that with the choice $w(z) = z$, the bound can be rewritten in the simple form

$$\|f(A)\boldsymbol{b} - \boldsymbol{f}_m\| \le \int_\Gamma \|(A - zI)^{-1}\|_2 \cdot \|\mathtt{res}_m(z)\|_2 \, |\mathrm{d}\mu(z)|,$$

which can be easily obtained directly from (3.3.2). The advantage of the formulation with $w(z) \equiv w$ is that the corresponding bound can be evaluated by computing the residual of a single linear system.

The residual-based a posteriori bound is summarized in Algorithm 3.2. Evaluating the a posteriori bound after m Krylov iterations requires the computation of the residual

$$\mathtt{res}_m(w) = \boldsymbol{b} - (A - wI)\mathtt{x}_m(w) = \boldsymbol{b} - (A - wI)V_m(A_m - wI)^{-1}\boldsymbol{e}_1\|\boldsymbol{b}\|_2.$$

Note that it is likely that an eigendecomposition of A_m has already been computed for the evaluation of $f(A_m)$, so the $m \times m$ linear system $(A_m - wI)^{-1}\boldsymbol{e}_1$ can be solved with just $O(m^2)$ cost, with no need to compute a factorization of the matrix $A_m - wI$. Hence we can compute $\mathtt{x}_m(w)$ for $O(mn)$ cost, and we can obtain $\mathtt{res}_m(w)$ with one additional matrix-vector multiplication with A. Alternatively, if we can keep in memory the matrix AV_m (whose columns are often computed anyway in the rational Arnoldi algorithm), we can entirely avoid additional matrix-vector multiplications with A and cheaply compute $\mathtt{res}_m(w)$ for a cost of $O(mn)$. In contrast, the first bound in Corollary 3.3.5 formulated in terms of linear system errors is significantly more expensive to compute, since in that case taking w constant is not equivalent to taking $w(z) = z$ and hence optimization of $w(z)$ is beneficial, and the exact solution of several shifted linear systems is required to compute the corresponding error norms.

3.3.4 Error bounds for quadratic forms

In this section we adapt the statement of Theorem 3.3.1 to the case of quadratic forms with $f(A)$ in the case of Hermitian A, and we show that it leads to improved bounds similarly to Section 3.2.1. Let us denote by $\psi_m = \boldsymbol{b}^H\boldsymbol{f}_m$ the approximation (2.5.9) to $\boldsymbol{b}^H f(A)\boldsymbol{b}$ from the rational Krylov subspace $\mathcal{Q}_m(A,\boldsymbol{b},\boldsymbol{\xi}_m)$. From (3.3.2) we immediately have the identity

$$\boldsymbol{b}^H f(A)\boldsymbol{b} - \psi_m = \int_\Gamma \boldsymbol{b}^H \mathtt{err}_m(z)\, \mathrm{d}\mu(z).$$

We can write $\boldsymbol{b} = (A - zI)\mathtt{x}_m(z) + \mathtt{res}_m(z)$, and since A is Hermitian we have

$$\begin{aligned}
\boldsymbol{b}^H \mathtt{err}_m(z) &= \mathtt{x}_m(z)^H(A - zI)\mathtt{err}_m(z) + \mathtt{res}_m(z)^H\mathtt{err}_m(z) \\
&= \mathtt{x}_m(z)^H\mathtt{res}_m(z) + \mathtt{res}_m(z)^H\mathtt{err}_m(z) \\
&= \mathtt{res}_m(z)^H\mathtt{err}_m(z),
\end{aligned}$$

where in the last equality we used the fact that $\mathtt{res}_m(z) \perp \mathcal{Q}_m(A, \boldsymbol{b})$, so $\mathtt{x}_m(z)^H \mathtt{res}_m(z) = 0$; see Lemma 3.1.1. The error for the approximation of $\boldsymbol{b}^H f(A)\boldsymbol{b}$ is therefore given by

$$\boldsymbol{b}^H f(A)\boldsymbol{b} - \psi_m = \int_\Gamma \mathtt{res}_m(z)^H \mathtt{err}_m(z)\, \mathrm{d}\mu(z). \tag{3.3.7}$$

Combining this error expression with (3.3.3) leads to the following theorem, which is the equivalent of Theorem 3.3.1 for quadratic forms.

Theorem 3.3.9. Assume that A is Hermitian and that f has the integral representation (3.3.1), and denote by ψ_m the approximation (2.5.9) to $\boldsymbol{b}^H f(A)\boldsymbol{b}$ from the rational Krylov subspace $\mathcal{Q}_m(A, \boldsymbol{b}, \boldsymbol{\xi}_m)$, with denominator polynomial q_{m-1}. Let $\mathcal{D} = \{\xi_1, \ldots, \xi_{m-1}\} \cup \Lambda(A) \cup \Lambda(A_m)$ and let w be a measurable function $w : \Gamma \to \mathbb{C} \setminus \mathcal{D}$. We have

$$\begin{aligned}
\boldsymbol{b}^H f(A)\boldsymbol{b} - \psi_m &= \int_\Gamma \left(\frac{q_{m-1}(z)}{q_{m-1}(w(z))} \det(h_{w(z),z}(A_m)) \right)^2 \mathtt{res}_m(w(z))^H k_z(A) \mathtt{res}_m(w(z))\, \mathrm{d}\mu(z) \\
&= \int_\Gamma \left(\frac{q_{m-1}(z)}{q_{m-1}(w(z))} \det(h_{w(z),z}(A_m)) \right)^2 \mathtt{res}_m(w(z))^H h_{w(z),z}(A) \mathtt{err}_m(w(z))\, \mathrm{d}\mu(z).
\end{aligned}$$

Proof. In (3.3.7), use $\mathtt{err}_m(z) = (A - zI)^{-1}\mathtt{res}_m(z)$ and use (3.3.3) to replace $\mathtt{res}_m(z)$ with $\mathtt{res}_m(w)$. □

The following corollary can be derived from the two error expressions in Theorem 3.3.9 with a proof similar to the one of Corollary 3.3.3.

Corollary 3.3.10. With the same hypothesis as in Theorem 3.3.9, assume that for some compact sets $\mathcal{S}_0, \mathcal{S}_1, \ldots, \mathcal{S}_m \subset \mathbb{R} \setminus \Gamma$ we have $\Lambda(A) \subset \mathcal{S}_0$, and $\lambda_i(A_m) \in \mathcal{S}_i$ for all $i = 1, \ldots, m$. Then we have

$$\|\boldsymbol{b}^H f(A)\boldsymbol{b} - \psi_m\|_2 \leq \int_\Gamma \left| \frac{q_{m-1}(z)}{q_{m-1}(w(z))} \right|^2 \prod_{j=1}^m \|h_{w(z),z}\|_{\mathcal{S}_j}^2 \cdot \|k_z\|_{\mathcal{S}_0} \cdot \|\mathtt{res}_m(w(z))\|_2^2\, |\mathrm{d}\mu(z)|$$

and similarly

$$\|\boldsymbol{b}^H f(A)\boldsymbol{b} - \psi_m\|_2 \leq \int_\Gamma G_m(z)\, |\mathrm{d}\mu(z)|,$$

where

$$G_m(z) = \left| \frac{q_{m-1}(z)}{q_{m-1}(w(z))} \right|^2 \prod_{j=1}^m \|h_{w(z),z}\|_{\mathcal{S}_j}^2 \cdot \|h_{w(z),z}\|_{\mathcal{S}_0} \cdot \|\mathtt{res}_m(w(z))\|_2 \cdot \|\mathtt{err}_m(w(z))\|_2.$$

Note that the first bound in Corollary 3.3.10 has the same structure as the second one in Corollary 3.3.3, with the difference that several of the factors are squared. This is in line with the fact that the error $\|\boldsymbol{b}^H f(A)\boldsymbol{b} - \psi_m\|_2$ is related to the error in the approximation of f with rational functions of type $(2m-1, 2m-2)$, instead of type $(m-1, m-1)$, so we can expect to have roughly twice the convergence rate compared to the error $\|f(A)\boldsymbol{b} - \boldsymbol{f}_m\|_2$ (see Proposition 2.5.12). A similar result has been obtained in [40, Section 6] for the Lanczos method.

Corollary 3.3.10 can be specialized by choosing different sets $\mathcal{S}_j$ to obtain a priori and a posteriori bounds. This is formalized in the following two corollaries.

Corollary 3.3.11. Under the assumptions of Theorem 3.3.9, we have the a priori bounds

$$\|\boldsymbol{b}^H f(A)\boldsymbol{b} - \psi_m\|_2 \leq \int_\Gamma \left| \frac{q_{m-1}(z)}{q_{m-1}(w(z))} \right|^2 \|h_{w(z),z}\|_{[a,b]}^{2m} \cdot \|k_z\|_{[a,b]} \cdot \|\mathtt{res}_m(w(z))\|_2^2\, |\mathrm{d}\mu(z)|$$

and

$$\|\boldsymbol{b}^H f(A)\boldsymbol{b} - \psi_m\|_2 \leq \int_\Gamma H_m(z)\, |\mathrm{d}\mu(z)|,$$

where

$$H_m(z) = \left| \frac{q_{m-1}(z)}{q_{m-1}(w(z))} \right|^2 \prod_{j=1}^m \|h_{w(z),z}\|^{2m+1} \cdot \|\mathtt{res}_m(w(z))\|_2 \cdot \|\mathtt{err}_m(w(z))\|_2.$$

Corollary 3.3.12. Under the assumptions of Theorem 3.3.9, we have the a posteriori bound

$$\|\boldsymbol{b}^H f(A)\boldsymbol{b} - \psi_m\|_2 \leq \int_\Gamma \left| \frac{q_{m-1}(z)}{q_{m-1}(w(z))} \cdot \frac{\chi_m(w(z))}{\chi_m(z)} \right|^2 \cdot \|k_z\|_{[a,b]} \cdot \|\mathtt{res}_m(w(z))\|_2^2 \, |\mathrm{d}\mu(z)|.$$

Remark 3.3.13. With the same arguments used in the proof of Proposition 3.3.8, it can be shown that the bounds in Corollary 3.3.10 are minimized by taking $w(z) = z$, and with this choice the bounds have much simpler expressions that can be derived directly from (3.3.7). From these expressions, it is easy to see that for $w(z) = z$ the second bound in Corollary 3.3.11 is always sharper than the first one, because of the inequality $\|\mathtt{err}_m(z)\|_2 \leq \|k_z\|_{[a,b]} \|\mathtt{res}_m(z)\|_2$. In the experiments of Section 3.4 we observe this also when $w(z)$ is an approximate minimizer of the integrand, assuming that the exact linear system error and residual norms are known. On the other hand, when using a priori bounds on the linear system error and residual norms, which bound is better depends on the linear system bounds that we use. In particular, if we have an a priori bound on the linear system residual and we use it to obtain a bound on the error via (3.3.8), then the first bound in Corollary 3.3.11 is always sharper than the second one, because of the inequality $\|k_z\|_{[a,b]} \leq \|h_{w,z}\|_{[a,b]} \|k_w\|_{[a,b]}$.

Let us briefly discuss how to select the function $w(z)$. In the case of a priori bounds, similarly to the case of $f(A)\boldsymbol{b}$, we should select $w(z)$ as a minimizer of the integrand evaluated at z, where in practice $\|\mathtt{res}_m(w)\|_2$ is replaced by an a priori bound of the form $\widetilde{\varphi}_m(w) \geq \|\mathtt{res}_m(w)\|_2$, and $\|\mathtt{err}_m(w)\|_2$ is replaced by $\varphi_m(w) \geq \|\mathtt{err}_m(w)\|_2$. In the end, we obtain

$$w(z) = \operatorname*{argmin}_{w \in \mathbb{C} \setminus \mathcal{D}} \left(\frac{1}{|q_{m-1}(w)|^2} \cdot \|h_{w,z}\|_{[a,b]}^{2m} \cdot \widetilde{\varphi}_m(w)^2 \right)$$

for the first bound in Corollary 3.3.11, and

$$w(z) = \operatorname*{argmin}_{w \in \mathbb{C} \setminus \mathcal{D}} \left(\frac{1}{|q_{m-1}(w)|^2} \cdot \|h_{w,z}\|_{[a,b]}^{2m+1} \cdot \varphi_m(w) \widetilde{\varphi}_m(w) \right)$$

for the second bound. In practice, the minimization problem can be solved approximately by taking w in a finite set $\mathcal{W}$, which can be chosen for example as a discretization of Γ, similarly to the case of bounds for $f(A)\boldsymbol{b}$.

When considering a posteriori bounds the situation is simpler, since by (3.3.3) for any $w \in \mathbb{C} \setminus \mathcal{D}$ we have

$$\left(\frac{q_{m-1}(z)}{q_{m-1}(w)} \cdot \frac{\chi_m(w)}{\chi_m(z)} \right) \mathtt{res}_m(w) = \mathtt{res}_m(z),$$

so we can choose $w(z) \equiv w$ for any fixed w and obtain the same bound.

3.3.5 An a priori bound for the linear system residual

In this section we present an a priori bound on the residual of a shifted linear system, which can be used to bound $\|\mathtt{res}_m(w(z))\|_2$ and $\|\mathtt{err}_m(w(z))\|_2$ in Corollaries 3.3.4 and 3.3.11. Assume that A is Hermitian and consider the shifted linear system

$$(A - wI)\boldsymbol{x} = \boldsymbol{b}, \qquad w \in \mathbb{R}.$$

This linear system coincides with a Sylvester equation in which one of the coefficient matrices is 1×1,

$$AX - XB = \boldsymbol{b}\boldsymbol{c}^H,$$

where $X = \boldsymbol{x} \in \mathbb{C}^{n \times 1}$, $B = w \in \mathbb{R}^{1 \times 1}$ and $\boldsymbol{c} = 1 \in \mathbb{R}$. As a consequence, the residual norm $\|\mathtt{res}_m(w)\|_2$ of the shifted linear system can be bounded by using the bounds developed in [7] for the residual of Sylvester equations solved using the rational Krylov subspaces $\mathcal{Q}_m(A, \boldsymbol{b}, \boldsymbol{\xi}_m)$ and $\mathcal{Q}_1(B^H, \boldsymbol{c})$. Since B and $\boldsymbol{c}$ are scalars, a Krylov subspace of dimension 1 generated with B^H and $\boldsymbol{c}$ coincides with the whole space, and the approximate solution of the Sylvester equation obtained using the method from [7] coincides with the rational Krylov approximation to the solution of a shifted linear system described in Section 3.1.2. The theorem that follows is obtained by specializing [7, Theorem 2.3] to the case of a shifted linear system with a Hermitian matrix A. The notation in the statement of the theorem is changed from [7] in order to match the notation used in the rest of this chapter.

Theorem 3.3.14 ([7, Theorem 2.3])**.** Assume that A is Hermitian with spectrum contained in the interval $[\lambda_{\min}, \lambda_{\max}]$, and let the set of poles $\{\xi_1, \dots, \xi_{m-1}\}$ be closed under complex conjugation. For any $w \in \mathbb{R} \setminus [\lambda_{\min}, \lambda_{\max}] \cup \{\xi_1, \dots, \xi_{m-1}\}$, we have

$$\|\mathtt{res}_m(w)\|_2 \leq \|\boldsymbol{b}\|_2 \, (4 + c) \, \gamma_m,$$

where

$$c = 2\sqrt{2}\sqrt{\kappa_2(A - wI)}$$

and

$$\gamma_m = \left|\frac{\varphi(w) - 1}{\varphi(w) + 1}\right| \cdot \prod_{j=1}^{m-1} \left|\frac{\varphi(w)/\varphi(\xi_j) - 1}{\varphi(w)/\varphi(\bar{\xi_j}) + 1}\right|, \qquad \varphi(z) = \sqrt{\frac{z - \lambda_{\max}}{z - \lambda_{\min}}}\,.$$

Proof. We show that this theorem follows by applying the second part of [7, Theorem 2.3] to the linear system $(A - wI)\boldsymbol{x} = \boldsymbol{b}$, interpreted as a Sylvester equation. In this proof we use some notation from [7].

Note that in our setting we have $B = w \in \mathbb{R}$, so the size of the Krylov subspace associated to B is $n = 1$, and the corresponding pole is $z_{B,1} = \infty$. On the other hand, the poles $\{z_{A,1}, \dots, z_{A,m}\}$ used for the rational Krylov subspace associated to A are given by $\{\infty, \xi_1, \dots, \xi_{m-1}\}$, where the pole at infinity appears because of the slightly different definition of a rational Krylov subspace given in [7]: namely, the vector $\boldsymbol{b}$ is included in the subspace only if one of the poles is ∞; the definition used in [7] coincides with Definition 5.2.1, which we are going to use in Chapter 5.

The numerical range of B is $\mathbb{W}(B) = \{w\}$, so we have $u_{B,1} \equiv 0$ (see [7, eq. (2.10)]) and

$$\gamma_{B,A} := \max_{z \in [\lambda_{\min}, \lambda_{\max}]} u_{B,1}(z) = 0.$$

We have

$$c_3 = 2\sqrt{2}\sqrt{\frac{\max\{|\lambda_{\min} - w|, |\lambda_{\max} - w|\}}{\min\{|\lambda_{\min} - w|, |\lambda_{\max} - w|\}}} = 2\sqrt{2}\sqrt{\kappa_2(A - wI)} =: c,$$

and

$$\gamma_{A,B} = u_{A,m}(w) = \left|\frac{\varphi(w) - 1}{\varphi(w) + 1}\right| \cdot \prod_{j=1}^{m-1} \left|\frac{\varphi(w)/\varphi(\xi_j) - 1}{\varphi(w)/\varphi(\bar{\xi_j}) + 1}\right| =: \gamma_m, \qquad \varphi(z) := \sqrt{\frac{z - \lambda_{\max}}{z - \lambda_{\min}}}\,,$$

where the first factor in γ_m corresponds to the pole at infinity (we refer to the statement of [7, Theorem 2.3] for the original definitions of c_3 and $\gamma_{A,B}$).

Denoting by $X^G_{m,n}$ the approximate solution to the Sylvester equation as defined in [7, eq. (1.6)], observe that the Sylvester equation residual $\|AX^G_{m,n} - X^G_{m,n}B - \boldsymbol{b}\boldsymbol{c}^H\|_F$ from [7] corresponds to the shifted linear system residual $\|\mathtt{res}_m(w)\|_2$ in our setting. Hence, by [7, Theorem 2.3] we can conclude that

$$\begin{aligned}\|\mathtt{res}_m(w)\|_2 &\leq \|\boldsymbol{b}\|_2 \, (4 \, \max\{\gamma_{A,B} + \gamma_{B,A}\} + c_3 \, (\gamma_{A,B} + \gamma_{B,A})) \\ &= \|\boldsymbol{b}\|_2 \, (4 + c_3) \, \gamma_m.\end{aligned}$$

□

Remark 3.3.15. The original statement of [7, Theorem 2.3] holds also for non-Hermitian matrices, by using the numerical ranges of A and B and the associated Green functions. The main assumption on the matrices A and B is that $\mathbb{W}(A) \cap \mathbb{W}(B) = \emptyset$, which in our setting corresponds to $w \notin \mathbb{W}(A)$. Here we prefer to consider only the Hermitian case, since it is difficult to compute the numerical range of a general matrix in practice.

Remark 3.3.16. The bound on the residual norm $\|\mathtt{res}_m(w)\|_2$ also provides a bound for the error norm $\|\mathtt{err}_m(w)\|_2$ via the inequality

$$\begin{aligned}\|\mathtt{err}_m(w)\|_2 &\leq \|(A - wI)^{-1}\|_2 \cdot \|\mathtt{res}_m(w)\|_2 \\ &\leq \max\{|\lambda_{\min} - w|^{-1}, |\lambda_{\max} - w|^{-1}\} \cdot \|\mathtt{res}_m(w)\|_2.\end{aligned} \tag{3.3.8}$$

The right-hand side of the inequality (3.3.8) can be used in (3.3.6) as the function $\varphi_m(w)$.

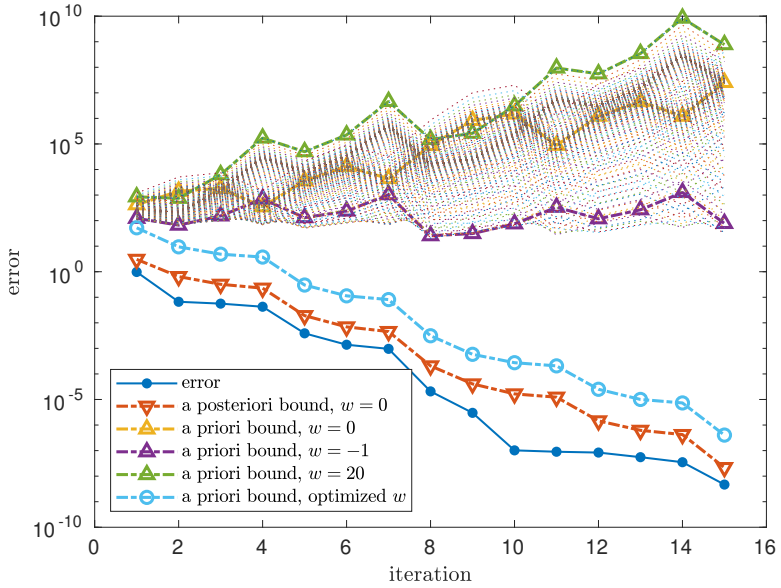


Figure 3.4 – A priori and a posteriori error bounds for $A^{-1/2}\boldsymbol{b}$ for different choices of w, where A is a real symmetric 1000×1000 matrix with logspaced eigenvalues in $[10^{-1}, 10^{1}]$.

3.4 Numerical experiments

In this section we include some experiments that demonstrate the accuracy of the error bounds derived in this chapter for the approximation of $f(A)\boldsymbol{b}$ and $\boldsymbol{b}^H f(A)\boldsymbol{b}$ when the function f is either Cauchy-Stieltjes or analytic in a neighborhood of the spectrum of A. We use the functions $f(z) = z^{-1/2}$ and $f(z) = \log(1+z)/z$ with the Cauchy-Stieltjes formulations given below Definition 2.2.2, and the function $f(z) = z^{1/2}$ with the Cauchy integral formula on the contour $\Gamma = (-\infty, 0]$, obtained as a limit according to the discussion at the start of Section 3.3.1. In all experiments, the poles used to construct the rational Krylov subspace are the asymptotically optimal poles for Cauchy-Stieltjes functions described in Section 2.5.3. In our implementation, we assume that the smallest and largest eigenvalues of A are known exactly. Note that an interval $[a, b] \supset \Lambda(A)$ is required both to compute the poles and to evaluate the a priori and a posteriori bounds. If no spectral information on A is available, it is still possible to compute estimates of the extremal eigenvalues of A by using the matrix A_m computed while running the rational Krylov method, and use them to estimate the a posteriori bounds. The integrals required for the evaluation of the bounds are computed numerically using the MATLAB `integral` function.

3.4.1 Comparison between bounds with and without optimization

We first conduct a simple experiment to show that the optimization of the parameter w discussed in Section 3.3.2 is fundamental for the applicability of a priori bounds. We consider the computation of $A^{-1/2}\boldsymbol{b}$, for a 1000×1000 real symmetric matrix A with logspaced eigenvalues in the interval $[10^{-1}, 10^{1}]$, and we compare the error with the residual-based a posteriori bound from Corollary 3.3.5 with $w = 0$ (`a posteriori bound,` $w = 0$) , and the error-based a priori bounds from Corollary 3.3.4 for some fixed values of w (`a priori bound,` $w = 0$, -1, 20) and with the optimization described in Algorithm 3.1, where $\mathcal{W}$ contains 50 logspaced points in $[-10^{4}, -10^{-4}]$ (`a priori bound, optimized` w). The linear system error norms in the a priori bounds are bounded using (3.3.8) and Theorem 3.3.14. The results are depicted in Figure 3.4. The dotted lines are a priori bounds obtained using 500 different values of w in $\mathbb{R} \setminus [10^{-1}, 10^{1}]$. It turns out that in this case none of the a priori bounds without optimization converge, and they give no useful information, with several of them quickly diverging. On the other hand, the bound obtained by optimizing w for each $z \in \Gamma$ used to evaluate the integral correctly captures the convergence behavior, although it is off by a couple of orders of magnitude. The a posteriori bound requires no optimization, confirming the discussion in Section 3.3.3.

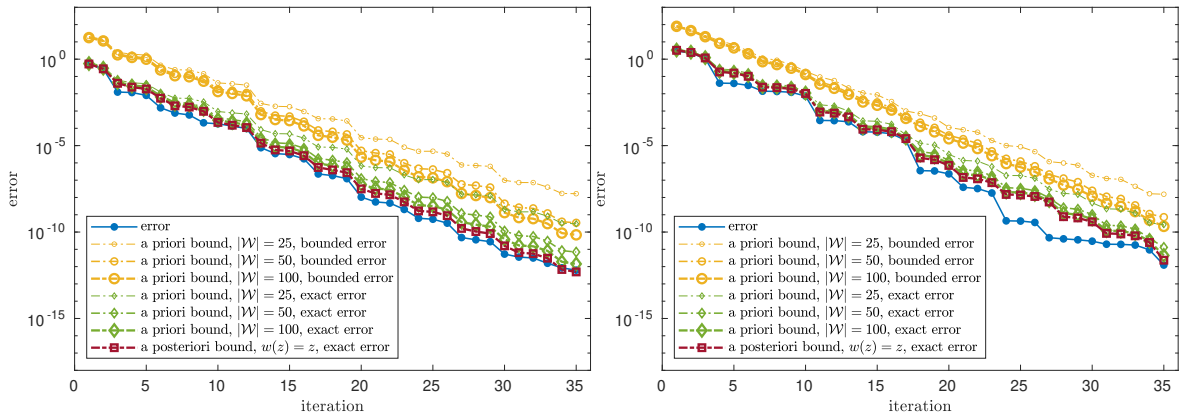


Figure 3.5 – Comparison of optimized a priori error bounds with different sets $\mathcal{W}$ for $f(A)\boldsymbol{b}$, where A is a 1000×1000 real symmetric matrix with logspaced eigenvalues in $[10^{-2}, 10^{2}]$. Left: $f(z) = \log(1+z)/z$. Right: $f(z) = \sqrt{z}$.

In the next experiment we compare the impact of the cardinality of the set $\mathcal{W}$ on the quality of the optimized bounds. We consider the computation of $f(A)\boldsymbol{b}$, where $f(z) = \log(1+z)/z$ or $f(z) = \sqrt{z}$, A is a 1000×1000 real symmetric matrix with logspaced eigenvalues in $[10^{-2}, 10^{2}]$ and $\boldsymbol{b}$ is a random vector. We compare the a priori bound from Algorithm 3.1 with different cardinalities of the set $\mathcal{W}$, both with $\varphi_m(w)$ given by the linear system error bound from Theorem 3.3.14 (`a priori bound`, $|W| = 25, 50, 100$, `bounded error`) and with the exact linear system errors (`a priori bound`, $|W| = 25, 50, 100$, `exact error`). In all cases, the set $\mathcal{W}$ is chosen as a discretization of $[-10^{5}, -10^{-5}]$ with logspaced points. We also make a comparison with the first a posteriori bound form Corollary 3.3.5 using $w(z) = z$ and the exact linear system error norms (`a posteriori bound`, $w(z) = z$, `exact error`), which by Proposition 3.3.8 is the best bound that can be obtained using the approach that we consider in this work, but it requires to compute or bound the error norm of a linear system for each point used in the discretization of the integral. The results are shown in Figure 3.5. For both functions, we see that the a priori bound with $|\mathcal{W}| = 100$ and exact error norms is very close to the a posteriori bound with $w(z) = z$. Both bounds are also quite close to the exact error. The bounds with $|\mathcal{W}| = 50$ are also only slightly worse than the ones with $|\mathcal{W}| = 100$.

3.4.2 Bounds for matrix-vector products

We compare the following bounds for the error in the computation of $f(A)\boldsymbol{b}$:

- the residual-based a posteriori bound from Algorithm 3.2 with $w = 0$ (`a posteriori bound, fixed` w`, exact residual`),
- the error-based a priori bound from Algorithm 3.1 with optimized w and linear system error bounded with (3.3.8) and Theorem 3.3.14 (`a priori bound, optimized` w`, bounded error`),
- the error-based a posteriori bound from Corollary 3.3.5 with optimized w and exact linear system error (`a posteriori bound, optimized` w`, exact error`),
- the error-based a priori bound from Algorithm 3.1 with optimized w and exact linear system error (`a priori bound, optimized` w`, exact error`),
- the a priori bound from [114, Corollary 4] for Cauchy-Stieltjes functions (`a priori bound slope, [114, Corollary 4]`),
- a simple error estimate based on the difference of consecutive approximations (`consecutive differences`), i.e., we approximate $\|f(A)\boldsymbol{b} - \boldsymbol{f}_m\|_2 \approx \|\boldsymbol{f}_m - \boldsymbol{f}_{m+1}\|_2$.

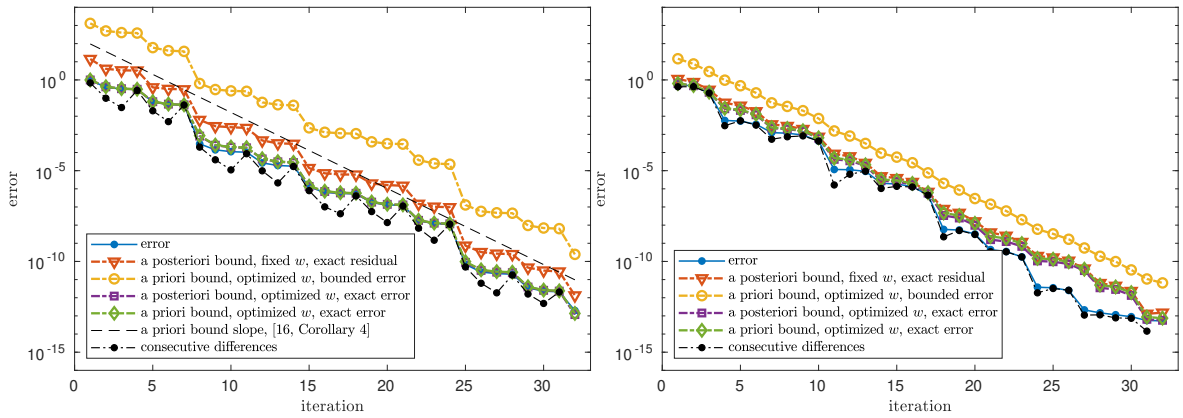


Figure 3.6 – A priori and a posteriori error bounds for $f(A)\boldsymbol{b}$, where A is the 2146×2146 real symmetric matrix `Nasa/nasa2146`. Left: $f(z) = \log(1+z)/z$. Right: $f(z) = \sqrt{z}$.

Recall that the poles used in [114, Corollary 4] are not nested, so they are not convenient for the construction of a rational Krylov subspace. Instead, we use a nested sequence of poles based on an equidistributed sequence, as we mentioned in Section 2.5.3 (see also [114, Section 3.5]); this pole sequence is guaranteed to have the same asymptotic rate of convergence as predicted by [114, Corollary 4], but the actual bound on the error may not be satisfied. The error estimate based on consecutive differences requires running the method for an additional iteration to be computed, and although it is usually quite reliable it may underestimate the true error when convergence is slow or stagnating; this behavior might be undesirable in situations in which a guarantee on the error bound is required. Note that the third and especially the fourth bound cannot be used in a practical setting, but they can provide interesting theoretical insight. Recall that the residual-based a posteriori bound is independent of the choice of w, so we can simply take $w = 0$ (see Section 3.3.3), but the error-based a posteriori bound does not have this property and hence benefits from the optimization of w, similarly to the a priori bounds. For all optimized bounds, we use as $\mathcal{W}$ a discretization of $[-10^3\lambda_{\max}(A), -10^{-3}\lambda_{\min}(A)]$ with 100 logspaced points.

The results for the two functions $f(z) = \log(1+z)/z$ and $f(z) = \sqrt{z}$ are shown in Figure 3.6, using as A the symmetric positive definite matrix `Nasa/nasa2146` of size 2146×2146 from the SuiteSparse Matrix Collection [51] and as $\boldsymbol{b}$ a random vector. We notice that the a priori and a posteriori bounds that use the exact error are slightly better than the residual-based a posteriori bound, and they almost overlap with each other. By comparing the a priori bounds that use the exact linear system error with the ones that use the bound in Theorem 3.3.14, we can conclude that most of the overestimation in the a priori error bounds for $f(A)\boldsymbol{b}$ comes from the overestimation of the linear system error when combining (3.3.8) with Theorem 3.3.14. Indeed, if we had access to the exact linear system errors in advance, we would be able to obtain a priori the fourth bound in Figure 3.6, which captures the convergence behavior very well. Although this scenario is clearly not realistic, an interesting consequence of this observation is that any improvement to a priori bounds for the linear system error would automatically lead to an equal improvement in the a priori bounds for $f(A)\boldsymbol{b}$ of Corollary 3.3.4, with the limit case of an exact linear system error yielding a bound for $f(A)\boldsymbol{b}$ that is quite close to the true error. The a priori bound [114, Corollary 4] correctly predicts the convergence rate of the rational Krylov approximation for $f(z) = \log(1+z)/z$ and it is more accurate than the a priori bound with bounded linear system error from Algorithm 3.1, although as we discussed above it might not be reliable as an upper bound because we are using the nested pole sequence from [114, Section 3.5]. Note that this bound does not hold for the function $f(z) = \sqrt{z}$ since it is not Cauchy-Stieltjes, but it is still able to predict the convergence rate quite well, due to the fact that $f(z) = z \cdot z^{-1/2}$ and $g(z) = z^{-1/2}$ is Cauchy-Stieltjes. The error estimates based on consecutive differences are usually quite accurate, but they sometimes underestimate the error; this is especially noticeable for the function $f(z) = \log(1+z)/z$.

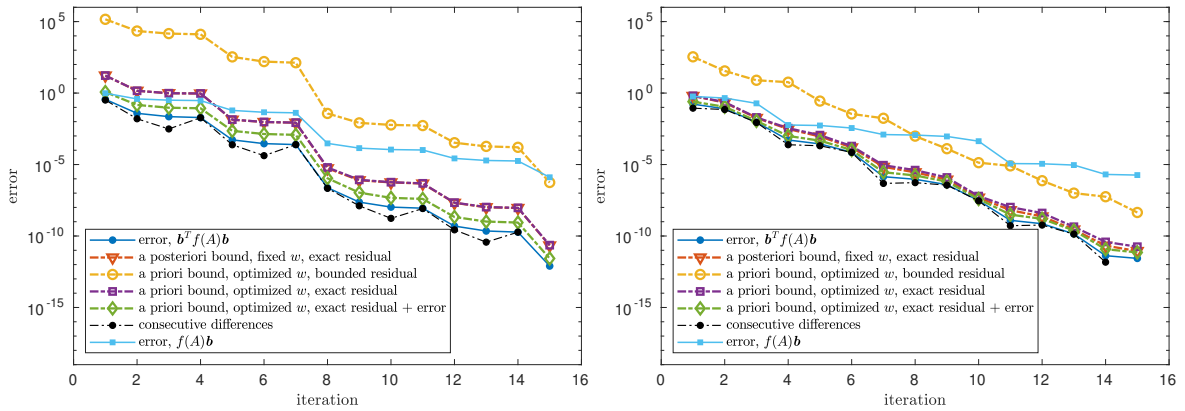


Figure 3.7 – A priori and a posteriori error bounds for $\boldsymbol{b}^H f(A)\boldsymbol{b}$, where A is the 2146×2146 real symmetric matrix `Nasa/nasa2146`. Left: $f(z) = \log(1+z)/z$. Right: $f(z) = \sqrt{z}$.

3.4.3 Bounds for quadratic forms

We compare the following bounds for the error in the computation of the quadratic form $\boldsymbol{b}^H f(A)\boldsymbol{b}$:

- the a posteriori bound from Corollary 3.3.12 with $w = 0$, using the exact linear system residual (`a posteriori bound, fixed` w`, exact residual`),
- the first a priori bound from Corollary 3.3.11 with optimized w and linear system residual bounded using Theorem 3.3.14 (`a priori bound, optimized` w`, bounded residual`),
- the first a priori bound from Corollary 3.3.11 with optimized w and exact linear system residual (`a priori bound, optimized` w`, exact residual`),
- the second a priori bound from Corollary 3.3.11 with optimized w and exact linear system error and residual (`a priori bound, optimized` w`, exact residual + error`),
- a simple error estimate based on the difference of consecutive approximations (`consecutive differences`), i.e., we approximate $|\boldsymbol{b}^H f(A)\boldsymbol{b} - \psi_m| \approx |\psi_m - \psi_{m+1}|$.

As in the previous experiment, for all optimized bounds we use as the set $\mathcal{W}$ a discretization of $[-10^3\lambda_{\max}(A), -10^{-3}\lambda_{\min}(A)]$ with 100 logspaced points. Recall that the second bound in Corollary 3.3.11 with $\|\mathtt{res}_m(w)\|_2$ bounded with Theorem 3.3.14 and $\|\mathtt{err}_m(w)\|_2$ bounded using (3.3.8) is always less accurate than the first one (see Remark 3.3.13), so we omit it from our experiments.

The results for the two functions $f(z) = \log(1+z)/z$ and $f(z) = \sqrt{z}$ are shown in Figure 3.7, where we also include the error for $f(A)\boldsymbol{b}$ for comparison. We can notice that the convergence for the quadratic form $\boldsymbol{b}^H f(A)\boldsymbol{b}$ is roughly twice as fast as the convergence for the matrix-vector product with $f(A)$. Similarly to the case of the bounds for $f(A)\boldsymbol{b}$, the a priori bound with exact linear system residual is significantly better than the bound that uses Theorem 3.3.14. Moreover, the a priori bound that uses both the exact residual and the exact error is even more accurate; this is in agreement with the observations in Remark 3.3.13, where we show that for the best possible choice of w, i.e. $w(z) = z$, the bound in Corollary 3.3.11 that uses both the error and the residual is more accurate than the bound that only uses $\|\mathtt{res}_m(z)\|_2$. The error estimate based on consecutive differences is very accurate for $f(z) = \sqrt{z}$, but it occasionally underestimates the error for $f(z) = \log(1+z)/z$.

3.4.4 Comparison with Lanczos setting

In this section we replicate the experiment in [40, Figure 4b] to show that the optimization of w can also lead to improvements for the error bounds in the polynomial Krylov (Lanczos) setting.

We consider the computation of $A^{1/2}\boldsymbol{b}$, where A is a 1000×1000 real symmetric matrix with uniformly spaced eigenvalues in $[10^{-2}, 10^2]$, using a polynomial Krylov method. The a priori and a posteriori bounds used in [40] can be obtained as a special case of the error-based bounds in Corollaries 3.3.4

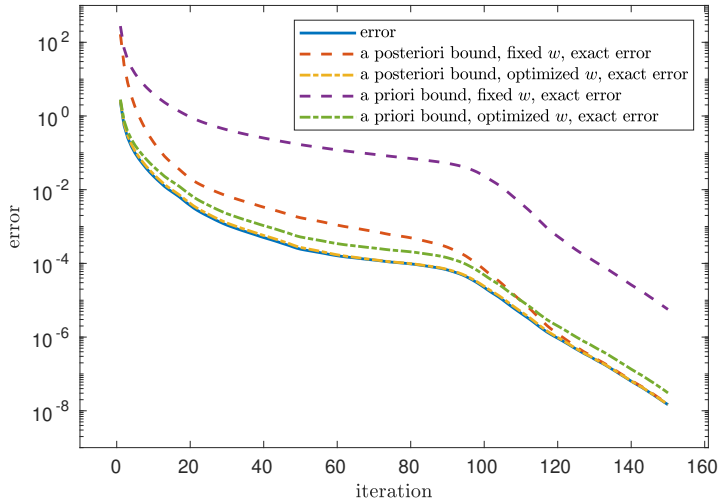


Figure 3.8 – Comparison of error bounds for $A^{1/2}\boldsymbol{b}$ in the Lanczos case, where A is a 1000×1000 real symmetric matrix with linearly spaced eigenvalues in $[10^{-2}, 10^{2}]$, with and without optimization of w.

and 3.3.5, by taking $q_{m-1}(z) \equiv 1$ and fixing $w = 0$. They are respectively denoted by `a priori bound, fixed` w`, exact error` and `a posteriori bound, fixed` w`, exact error` in Figure 3.8. We compare these bounds with the ones obtained by optimizing w as described in Section 3.3.2, which we label as `a priori bound, optimized` w`, exact error` and `a posteriori bound, optimized` w`, exact error`. In particular, the optimized a priori bound coincides with the one given in Algorithm 3.1 when all the poles $\xi_j = \infty$. Note that since we are using $\|\mathtt{err}_m(w)\|_2$ instead of $\|\mathtt{res}_m(w)\|_2$ in the a posteriori bound, the discussion in Section 3.3.3 does not apply and the bound can be improved by optimizing the choice of w. Similarly to [40], we assume that the error norms $\|\mathtt{err}_m(w)\|_2$ are known exactly. This assumption allows us to theoretically investigate the error bounds by eliminating any dependence on possibly inaccurate bounds on the linear system error norms, although it is not realistic in a practical scenario.

The results of the comparison are shown in Figure 3.8, where the optimized bounds are computed using as set $\mathcal{W}$ a discretization of $[-10^5, -10^{-5}]$ with 25 logspaced points. We can see that optimizing w gives significant improvements, especially for the a priori bound; the optimized a posteriori bound is practically overlapping with the error curve. If the number of points in $\mathcal{W}$ is increased to 100, the accuracy of the optimized a priori bound increases, and it becomes essentially indistinguishable from the optimized a posteriori bound. It should be mentioned that the plots in [40, Figure 4] consider the A-norm of the error, while here we consider the 2-norm, so the error curves shown in Figure 3.8 are slightly different from the ones in [40].

Chapter 4

Computation of generalized matrix functions with rational Krylov methods

Similarly to the case of standard matrix functions, in many applications in which generalized matrix functions (GMFs, see Definition 2.4.1) are involved, it is often required to compute the action of a GMF on a vector [4, 5], and hence it is preferable to use a method that avoids the computation of the whole GMF by means of an SVD. This problem was recently investigated in [5, 6], using methods based on the Golub-Kahan bidiagonalization in [5] and on Chebyshev polynomial interpolation in [6].

In this chapter we propose a generalization of the method proposed in [5], using the interpretation of the Golub-Kahan bidiagonalization in terms of Krylov subspaces. Indeed, performing k steps of the Golub-Kahan bidiagonalization of a matrix A with starting vector $\boldsymbol{b}$ is equivalent to the simultaneous computation of orthonormal bases of the polynomial Krylov subspaces $\mathcal{K}_k(A^HA, \boldsymbol{b})$ and $\mathcal{K}_k(AA^H, A\boldsymbol{b})$. By replacing the polynomial Krylov subspaces with the corresponding rational Krylov subspaces, we obtain a rational Krylov method for the computation of the action of a GMF on a vector, which includes the method proposed in [5] as a special case. As one could expect by analogy with the case of standard matrix functions, in the case of non-analytic functions and functions of low regularity these rational Krylov methods converge faster than those based on the Golub-Kahan bidiagonalization. However, their increased effectiveness comes at the cost of having to solve a linear system at each iteration, while the methods discussed in [5, 6] only require matrix-vector products.

The Golub-Kahan bidiagonalization of a matrix can be computed with a short-term recurrence, which relies on the fact that the projected matrix is bidiagonal. This structure is unfortunately not preserved in the rational case that we consider here. However, we are still able to construct a short-term recurrence to update the rational Krylov bases and the projected matrix in Section 4.2.3, using the fact that the projected matrix is a quasiseparable matrix [69, 148].

In Section 4.3 we also prove some error bounds that link the error of the method from [5] based on the Golub-Kahan bidiagonalization and the rational methods introduced in this chapter with, respectively, the error of uniform polynomial and rational approximation of the function f on an iterval that contains the singular values of A. These bounds are a direct generalization of the bounds for standard matrix functions, and they can be proved with the same techniques. Although the connection of GMFs to standard matrix functions is well known, to the best of our knowledge these error bounds for the approximation of GMFs have never appeared in previous literature. The contents of this chapter have been published in [37].

4.1 Properties of generalized matrix functions

We start by introducing some properties of generalized matrix functions that will be required in the sections that follow. A discussion of additional properties of generalized matrix functions can be found

in [5]. Let $A \in \mathbb{C}^{m\times n}$ be a possibly rectangular matrix, and let $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_r$ be its nonzero singular values, where $r = \operatorname{rank}(A)$. Recall that $f^\diamond(A) = Uf^\diamond(\Sigma)V^H$, where $U\Sigma V^H$ is an SVD of A (see Definition 2.4.1).

Remark 4.1.1 ([6, Theorem 2.1])**.** If p is a polynomial that interpolates f in $\sigma_1, \ldots, \sigma_r$, it is easy to see that we have $f^\diamond(A) = p^\diamond(A)$. Moreover, since $\sigma_i > 0$ for $i = 1, \ldots, r$, we can always take p as an odd polynomial, i.e., $p(z) = q(z^2)z$ for some polynomial q.

The following lemma gives some conditions under which a GMF coincides with a standard matrix function.

Lemma 4.1.2 ([5, Remark 5])**.** Let $S \in \mathbb{C}^{n\times n}$ be a Hermitian matrix and let f be defined on the singular values of S. If S is positive definite, or S is positive semidefinite and $f(0) = 0$, we have

$$f^\diamond(S) = f(S).$$

Proof. Since S is Hermitian positive semidefinite there exist $U \in \mathbb{C}^{n\times n}$ unitary and Σ diagonal with non-negative diagonal entries such that $S = U\Sigma U^H$. Assuming that the diagonal entries of Σ are ordered in a non-decreasing way, such a decomposition is both an SVD and an eigenvalue decomposition, and hence we have

$$f^\diamond(S) = Uf^\diamond(\Sigma)U^H \qquad \text{and} \qquad f(S) = Uf(\Sigma)U^H.$$

If we assume that either $f(0) = 0$ or S is positive definite, we have $f(\Sigma) = f^\diamond(\Sigma)$ and so it follows that $f(S) = f^\diamond(S)$. □

The results that follow give us expressions of $f^\diamond(A)$ in terms of A and A^H when f is either a polynomial or a rational function, which will be useful to derive Krylov subspace methods for the approximation of $f^\diamond(A)\boldsymbol{b}$.

Proposition 4.1.3. Let p be an odd polynomial, i.e., such that we can write $p(z) = q(z^2)z$ for some polynomial q. Then, for any matrix $A \in \mathbb{C}^{m\times n}$ we have

$$p^\diamond(A) = q(AA^H)A = Aq(A^HA).$$

Proof. This proof can be found in [6, Section 2.2]. Let $A = U\Sigma V^H$ be an SVD of A, and notice that $AA^H = U\Sigma^2U^H$ is an eigendecomposition of AA^H. From the definition of a matrix function we have

$$q(AA^H)A = Uq(\Sigma^2)U^HU\Sigma V^H = Uq(\Sigma^2)\Sigma V^H = Up(\Sigma)V^H.$$

Since $p(0) = 0$, we have $Up(\Sigma)V^H = p^\diamond(A)$ and we conclude that $p^\diamond(A) = q(AA^H)A$. The equivalence with A^HA can be proved in the same way. □

Corollary 4.1.4. Let r be a rational function with an odd numerator and an even denominator, i.e., $r(z) = q(z^2)^{-1}p(z^2)z$ for some polynomials p and q, such that the singular values of A and 0 are not roots of q. Then we have

$$r^\diamond(A) = q(AA^H)^{-1}p(AA^H)A = Aq(A^HA)^{-1}p(A^HA).$$

Proof. The proof is essentially the same as that of Proposition 4.1.3. Let $A = U\Sigma V^H$ be an SVD of A. Then we have $AA^H = U\Sigma^2U^H$, and more generally $s(AA^H) = Us(\Sigma^2)U^H$ for any polynomial s. Since 0 is not a root of q, we have $r(0) = 0$ and hence by Lemma 4.1.2 we have

$$r^\diamond(A) = Ur^\diamond(\Sigma)V^H = Ur(\Sigma)V^H = Uq(\Sigma^2)^{-1}p(\Sigma^2)\Sigma V^H.$$

Note that $q(\Sigma^2)^{-1}$ is well defined because the singular values of A are not roots of q. Now, multiplying by $U^HU = I$ we obtain

$$r^\diamond(A) = Uq(\Sigma^2)^{-1}U^HUp(\Sigma^2)U^HU\Sigma V^H = q(AA^H)^{-1}p(AA^H)A.$$

The second identity can be obtained in a similar way. □

The following proposition gives a formulation of the above results for general functions. See also [5, Theorem 10] for an alternative formulation of the same identity.

Proposition 4.1.5. Let $A \in \mathbb{C}^{m\times n}$ and let f be a function defined on the nonzero singular values of A. Defining $g(z) = f(\sqrt{z})/\sqrt{z}$ for $z \neq 0$, we have

$$f^\diamond(A) = g^\diamond(AA^H)A = Ag^\diamond(A^HA).$$

Moreover, if $\lim_{z\to 0} f(z)/z = 0$, we can define $g(0) = 0$. Then $g(AA^H)$ and $g(A^HA)$ are well defined, and we have

$$f^\diamond(A) = g(AA^H)A = Ag(A^HA).$$

Proof. Let $p(z) = q(z^2)z$ be an odd polynomial that interpolates f in the nonzero singular values of A. From Proposition 4.1.3 and Remark 4.1.1 we have

$$f^\diamond(A) = p^\diamond(A) = q(AA^H)A = Aq(A^HA).$$

Since p interpolates f in the nonzero singular values of A, we have that q interpolates g in the squares of the nonzero singular values of A, which are the nonzero singular values of AA^H and A^HA. Hence $q^\diamond(AA^H) = g^\diamond(AA^H)$ and $q^\diamond(A^HA) = g^\diamond(A^HA)$. If $g(0) = 0$, by Lemma 4.1.2 we also have $g^\diamond(A^HA) = g(A^HA)$ and $g^\diamond(AA^H) = g(AA^H)$. □

Remark 4.1.6. If AA^H is positive definite, by Lemma 4.1.2 we have that $f^\diamond(A) = g(AA^H)A$ without the assumption $\lim_{z\to 0} f(z)/z = 0$, and similarly if A^HA is positive definite we have $f^\diamond(A) = Ag(A^HA)$. Note that we could also artificially define $g(0) = 0$ without any assumptions on f, since the definition of a GMF only depends on the nonzero singular values of the matrix. However, this would cause g to be discontinuous at 0 and it would not be very useful in practice.

The following proposition links $f^\diamond(A)$ with $f^\diamond(A^H)$, which will be useful when A is rectangular. The more general statement in Proposition 4.1.7 can be seen as a generalization of [5, Proposition 7(iv)] and [94, Theorem 4(d)].

Proposition 4.1.7. Let $A \in \mathbb{C}^{m\times n}$ and let f be a function defined on the nonzero singular values of A. Then

$$f^\diamond(A) = (A^\dagger)^H f^\diamond(A^H)A.$$

More generally, assume that $f(z) = g(z)h(z)k(z)$, where g, h, k are functions defined on the nonzero singular values of A. Then

$$f^\diamond(A) = g^\diamond(A)h^\diamond(A^H)k^\diamond(A).$$

Proof. We directly prove the generalized version, since the first statement simply follows by taking $g(z) = z^{-1}$, $h(z) = f(z)$ and $k(z) = z$. Let A have the singular value decomposition $A = U\Sigma V^H$. We have

$$\begin{aligned} g^\diamond(A)h^\diamond(A^H)k^\diamond(A) &= Ug^\diamond(\Sigma)V^HVh^\diamond(\Sigma^H)U^HUk^\diamond(\Sigma)V^H \\ &= Ug^\diamond(\Sigma)h^\diamond(\Sigma)^Hk^\diamond(\Sigma)V^H \\ &= Uf^\diamond(\Sigma)V^H = f^\diamond(A), \end{aligned}$$

where we used that $g^\diamond(\Sigma)h^\diamond(\Sigma)^Hk^\diamond(\Sigma) = f^\diamond(\Sigma)$, which can be verified directly. □

Remark 4.1.8. The same proof of Proposition 4.1.7 can be used to show that, if $f(z)g(z) = h(z)k(z)$, then

$$f^\diamond(A^H)g^\diamond(A) = h^\diamond(A^H)k^\diamond(A).$$

In particular we have $A^Hf^\diamond(A) = f^\diamond(A^H)A$.

4.2 Krylov subspace methods for GMFs

The computation of $f^\diamond(A)\boldsymbol{b}$ by using an SVD of A can be unfeasible if the size of A is large. A possible way to approximate the product of a generalized matrix function with a vector is to use two rectangular matrices with orthonormal columns to project the matrix A onto a smaller space and then compute the generalized matrix function of this projected matrix: let $k \ll n$ and let $U_k, V_k \in \mathbb{C}^{n\times k}$ with orthonormal columns and $B_k = U_k^H A V_k \in \mathbb{C}^{k\times k}$, then we can approximate

$$f^\diamond(A)\boldsymbol{b} \approx U_k f^\diamond(B_k) V_k^H \boldsymbol{b}, \tag{4.2.1}$$

where the matrix $f^\diamond(B_k)$ can be computed with a singular value decomposition.

In Section 4.2.1 we describe how to compute such a projected matrix using the Golub-Kahan bidiagonalization, which is equivalent to using a polynomial Krylov method on the matrices $A^H A$ and AA^H. This strategy corresponds to the "third approach" discussed in [5, Section 5.4]; the numerical results of [5] indicate that (4.2.1) is often a more effective approximation compared with the other approaches that they propose, which are based on Gauss and Gauss-Radau quadrature formulas. In Section 4.2.2 we generalize this approach to the rational Krylov case and we show in Section 4.2.3 that a short-term recurrence like the one of the Golub-Kahan bidiagonalization can be obtained also in the rational case.

4.2.1 Golub-Kahan bidiagonalization

The Golub-Kahan bidiagonalization is a method for the computation of a truncated SVD that was introduced for the first time in 1965 [86].

Theorem 4.2.1. Let $A \in \mathbb{C}^{m\times n}$, with $m > n$. There exist unitary matrices $P \in \mathbb{C}^{m\times m}, Q \in \mathbb{C}^{n\times n}$ such that

$$P^H A Q = B = \begin{bmatrix} \alpha_1 & \beta_1 & \dots & \dots & 0 \\ 0 & \alpha_2 & \beta_2 & \dots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & 0 & \alpha_{n-1} & \beta_{n-1} \\ 0 & \dots & \dots & 0 & \alpha_n \\ 0 & \dots & \dots & \dots & 0 \\ \vdots & & & & \vdots \\ 0 & \dots & \dots & \dots & 0 \end{bmatrix}.$$

Moreover the first column of Q can be chosen almost[1] arbitrarily.

The proof of Theorem 4.2.1 is constructive and it is usually called Householder bidiagonalization process. It can be found in [88, Section 5.4].

In the case of large matrices the full bidiagonalization is too expensive. The goal of the Golub-Kahan bidiagonalization is to extract good approximations of singular values and singular vectors before the full bidiagonalization is completed. Denote by $\boldsymbol{p}_j$ and $\boldsymbol{q}_j$ the columns of P and Q, respectively. By stopping the bidiagonalization process after k steps, we obtain the matrices $P_k = [\boldsymbol{p}_1 \dots \boldsymbol{p}_k]$, $Q_k = [\boldsymbol{q}_1 \dots \boldsymbol{q}_k]$ and

$$B_k = \begin{bmatrix} \alpha_1 & \beta_1 & 0 & \dots & 0 \\ 0 & \alpha_2 & \beta_2 & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & 0 \\ \vdots & & 0 & \alpha_{k-1} & \beta_{k-1} \\ 0 & \dots & \dots & 0 & \alpha_k \end{bmatrix},$$

[1] In order to avoid breakdowns, the first column of Q must be chosen as a unit vector $\boldsymbol{q}_1$, such that $\mathcal{K}_n(A^H A, \boldsymbol{q}_1)$ has dimension n.

such that

$$\begin{aligned} AQ_k &= P_k B_k, \\ A^H P_k &= Q_k B_k^H + \boldsymbol{s}_k \boldsymbol{e}_k^T, \end{aligned}$$

where $\boldsymbol{s}_k = \bar{\beta}_k \boldsymbol{q}_{k+1}$. In particular we have $P_k^H A Q_k = B_k$. The columns of P_k, Q_k and the entries of B_k can be computed using the short-term recurrences

$$\begin{aligned} A\boldsymbol{q}_j &= \alpha_j \boldsymbol{p}_j + \beta_{j-1}\boldsymbol{p}_{j-1}, \qquad j \geq 2, \\ A^H \boldsymbol{p}_j &= \bar{\alpha}_j \boldsymbol{q}_j + \bar{\beta}_j \boldsymbol{q}_{j+1}, \qquad j \geq 1. \end{aligned} \tag{4.2.2}$$

Note that one can choose α_k and β_k to be real and positive, see [88, Section 10.4.1]. It can be shown that

$$\begin{aligned} \operatorname{span}\{\boldsymbol{q}_1, \ldots, \boldsymbol{q}_k\} &= \mathcal{K}_k(A^H A, \boldsymbol{q}_1), \\ \operatorname{span}\{\boldsymbol{p}_1, \ldots, \boldsymbol{p}_k\} &= \mathcal{K}_k(AA^H, A\boldsymbol{q}_1), \end{aligned}$$

thus the convergence of the Golub-Kahan bidiagonalization follows from the convergence of the Lanczos method applied on $A^H A$ and AA^H. For a given k, we can approximate A with $P_k B_k Q_k^H$, and hence we can compute an approximation of the SVD of A by computing the SVD of B_k. For further information on the Golub-Kahan bidiagonalization we refer to [88, Chapter 10].

The vector $f^\diamond(A)\boldsymbol{b}$ can be approximated by the expression

$$\boldsymbol{f}_k^\diamond := P_k f^\diamond(B_k) Q_k^H \boldsymbol{b} = \|\boldsymbol{b}\|_2 P_k f^\diamond(B_k)\boldsymbol{e}_1 \ \in \ \mathcal{K}_k(AA^H, A\boldsymbol{b}). \tag{4.2.3}$$

We refer to this approximation as a polynomial Krylov method for GMFs.

4.2.2 Rational Krylov methods for GMFs

We have seen that the Golub-Kahan bidiagonalization computes orthonormal bases for the polynomial Krylov subspaces $\mathcal{K}_k(A^H A, \boldsymbol{b})$ and $\mathcal{K}_k(AA^H, A\boldsymbol{b})$. By analogy with that approach, in this section we propose to compute an approximation to $f^\diamond(A)\boldsymbol{b}$ using the rational Krylov subspaces

$$\mathcal{Q}_k(A^H A, \boldsymbol{b}, \boldsymbol{\xi}_k) \qquad \text{and} \qquad \mathcal{Q}_k(AA^H, A\boldsymbol{b}, \boldsymbol{\xi}_k),$$

for a given sequence of poles $\{\xi_j\}_{j\geq 1}$.

Assume that we have constructed two matrices with orthonormal columns P_k and Q_k, such that $\operatorname{range}(P_k) = \mathcal{Q}_k(AA^H, A\boldsymbol{b})$ and $\operatorname{range}(Q_k) = \mathcal{Q}_k(A^H A, \boldsymbol{b})$. Then, defining $B_k = P_k^H A Q_k$, by analogy with the polynomial Krylov approach we can introduce the vector

$$\boldsymbol{f}_k^\diamond = P_k f^\diamond(B_k) Q_k^H \boldsymbol{b} = \|\boldsymbol{b}\|_2 P_k f^\diamond(B_k)\boldsymbol{e}_1, \tag{4.2.4}$$

which is an approximation to $f^\diamond(A)\boldsymbol{b}$ from the rational Krylov subspace $\mathcal{Q}_k(AA^H, A\boldsymbol{b})$.

First of all, notice that it is sufficient to compute only one of the two rational Krylov subspaces: indeed, since we have $A\mathcal{Q}_k(A^H A, \boldsymbol{b}) = \mathcal{Q}_k(AA^H, A\boldsymbol{b})$, we can compute an orthonormal basis of the subspace $\mathcal{Q}_k(AA^H, A\boldsymbol{b})$ by simply orthonormalizing the columns of AQ_k. This is equivalent to computing a QR decomposition $AQ_k = W_k R_k$, where W_k has orthonormal columns and R_k is upper triangular, so we can set $P_k = W_k$. Moreover, we have

$$B_k = P_k^H A Q_k = W_k^H W_k R_k = R_k,$$

i.e., with the QR decomposition we also recover the matrix B_k without the need to project A explicitly. The basis Q_k of the subspace $\mathcal{Q}_k(A^H A, \boldsymbol{b})$ can be computed by applying the rational Arnoldi algorithm (Algorithm 2.2) to the matrix $A^H A$ with initial vector $\boldsymbol{b}$. The procedure to compute the approximation (4.2.4) is summarized in Algorithm 4.1.

Algorithm 4.1 Rational Krylov approximation of $f^\diamond(A)\boldsymbol{b}$

Input: $A \in \mathbb{R}^{m\times n}, \boldsymbol{b} \in \mathbb{R}^n$, function f, poles $\{\xi_1, \dots, \xi_{k-1}\}$.
Output: $\boldsymbol{f}_k^\diamond \in \mathcal{Q}_k(AA^H, A\boldsymbol{b}, \boldsymbol{\xi}_k)$ such that $\boldsymbol{f}_k^\diamond \approx f^\diamond(A)\boldsymbol{b}$, given by (4.2.4).
1: Compute an orthonormal basis Q_k of $\mathcal{Q}_k(A^HA, \boldsymbol{b}, \boldsymbol{\xi}_k)$ with the rational Arnoldi algorithm.
2: Compute the QR decomposition $P_kB_k = AQ_k$.
3: Compute $f^\diamond(B_k)$ via an SVD of B_k.
4: Compute $\boldsymbol{f}_k^\diamond = \|\boldsymbol{b}\|_2 P_k f^\diamond(B_k)\boldsymbol{e}_1$.

4.2.3 Short-term recurrence for rational Golub-Kahan algorithm

In the Golub-Kahan bidiagonalization (without reorthogonalization), we can compute the last column of P_k, Q_k and B_k just by knowing the two previous columns of P_k and Q_k, by means of the expressions (4.2.2). This short-term recurrence is possible because of the bidiagonal structure of the matrix B_k, that is unfortunately not preserved when we perform a rational Krylov method.

In this section we show that if a rational Krylov method is used then B_k is a quasiseparable matrix (see Definition 4.2.3). This structure extends the bidiagonal form that is obtained during the Golub-Kahan bidiagonalization. Using this structure, we build a short-term recurrence that allows us to update the matrices P_k and B_k while avoiding the full orthogonalization related to the computation of the QR factorization of the matrix AQ_k.

We note that rank structures of rational Krylov matrices have been extensively investigated in the literature, see for instance [34, 69, 128, 148], and they have been exploited in [110, 111]. Structured projected matrices that lead to short-term recurrences have also appeared in the context of biorthogonal rational Krylov methods [149]. In the following, we use a MATLAB-like notation $A(i:j, h:k)$ to denote the submatrix of A associated with row indices from i to j, and column indices from h to k.

Definition 4.2.2 ([152, Definition 1.21])**.** A matrix $S \in \mathbb{C}^{n\times n}$ is called a semiseparable matrix if all the submatrices extracted from the lower and upper triangular part of the matrix have rank at most one, that is

$$\operatorname{rank} S(i:n, 1:i) \le 1 \qquad \text{and} \qquad \operatorname{rank} S(1:i, i:n) \le 1,$$

for every $i = 1, \dots, n$. Moreover, S is called a generator representable semiseparable matrix [152, Definition 1.23] if the lower and upper triangular parts of the matrix are derived from a rank 1 matrix, that is

$$\operatorname{tril}(S) = \operatorname{tril}(\boldsymbol{u}\boldsymbol{v}^H) \qquad \text{and} \qquad \operatorname{triu}(S) = \operatorname{triu}(\boldsymbol{p}\boldsymbol{q}^H),$$

for some $\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{p}, \boldsymbol{q} \in \mathbb{C}^n$, where $\operatorname{tril}(A)$ and $\operatorname{triu}(A)$ extract respectively the lower and upper triangular part of a matrix A.

Definition 4.2.3 ([152, Definition 1.24])**.** A matrix $S \in \mathbb{C}^{n\times n}$ is called a quasiseparable matrix if all the submatrices extracted from the strictly lower and strictly upper triangular part of the matrix are of rank at most 1, that is

$$\operatorname{rank} S(i+1:n, 1:i) \le 1 \quad \text{and} \quad \operatorname{rank} S(1:i, i+1:n) \le 1,$$

for every $i = 1, \dots, n$.

The following theorem from [152, Section 1.5.2] gives us a complete characterization of invertible semiseparable matrices.

Theorem 4.2.4. The inverse of an invertible tridiagonal matrix is a semiseparable matrix, and vice versa. Moreover, the inverse of an invertible irreducible tridiagonal matrix is a generator representable semiseparable matrix and vice versa.

Proof. See the proofs of [152, Theorem 1.38] and [152, Corollary 1.39]. □

Recall that when A is Hermitian and the poles $\boldsymbol{\xi}_{k+1}$ are real, we can compute an orthonormal basis Q_{k+1} of $\mathcal{Q}_{k+1}(A, \boldsymbol{b}, \boldsymbol{\xi}_{k+1})$ that satisfies a rational Arnoldi relation with $\underline{H}_k$ and $\underline{K}_k$ tridiagonal using the rational Lanczos algorithm (see Section 2.5.2). Specifically, we have

$$AQ_{k+1}\underline{K}_k = Q_{k+1}\underline{H}_k, \qquad \text{with} \qquad \underline{K}_k = \begin{bmatrix} I_k \\ \mathbf{0}_k^T \end{bmatrix} + D_k\underline{H}_k, \tag{4.2.5}$$

where $D_k = \operatorname{diag}(0, \xi_1^{-1}, \ldots, \xi_k^{-1})$ and $\underline{H}_k$ is a $(k+1) \times k$ full rank tridiagonal irreducible Hermitian matrix, with the convention that $1/\infty = 0$.

Starting form (4.2.5), we are going to prove that the projection of A onto the Krylov subspace (i.e., the matrix $Q_{k+1}^H A Q_{k+1}$) is the sum of a diagonal matrix and a semiseparable matrix. A similar result has been proved in [69, Theorem 1]. This fact will be exploited to obtain a short-term recurrence for the computation of B_k and P_k.

Theorem 4.2.5. Let $A \in \mathbb{C}^{n\times n}$ be a Hermitian matrix and let $Q_{k+1} \in \mathbb{C}^{n\times(k+1)}$ be the orthonormal basis of $\mathcal{Q}_{k+1}(A, \boldsymbol{b}, \boldsymbol{\xi}_{k+1})$ generated by the rational Lanczos algorithm, with real poles $\{\xi_1, \ldots, \xi_k\}$ different from 0 and ∞. Then we have

$$J_{k+1} := Q_{k+1}^H A Q_{k+1} = S + \widetilde{D}_k,$$

where $\widetilde{D}_k = \operatorname{diag}(0, \xi_1, \ldots, \xi_k)$ and S is a Hermitian generator representable semiseparable matrix.

Proof. Let us assume that the rational Lanczos algorithm is run for one additional iteration using the pole $\xi_{k+1} = \infty$. Then, from (4.2.5) we obtain the relation

$$AQ_{k+2}\underline{K}_{k+1} = Q_{k+2}\underline{H}_{k+1}, \tag{4.2.6}$$

where $\underline{K}_{k+1}$ and $\underline{H}_{k+1}$ are tridiagonal and the last row of $\underline{K}_{k+1}$ is zero. Hence, multiplying (4.2.6) on the left by Q_{k+1}^H we get

$$J_{k+1} = H_{k+1}K_{k+1}^{-1}.$$

We can assume that the first row of J_{k+1} has at least one nonzero off-diagonal entry. Indeed, this is true if k is such that Q_{k+1} is an invariant subspace of A, i.e., if $AQ_{k+1} = Q_{k+1}J_{k+1}$, since the first column of Q_{k+1} is not an eigenvector of A. So the assumption is satisfied provided that we take k large enough, and if we prove the statement in this case, then it also follows for smaller k because leading principal blocks of a generator representable semiseparable matrix still have the same property.

Let us define $\widehat{D}_k = \operatorname{diag}(\gamma, \xi_1, \ldots, \xi_k)$, for $\gamma \in \mathbb{C}$. Since the first row of J_{k+1} has at least one nonzero off-diagonal entry, it can be noticed that $J_{k+1} - \widetilde{D}_k$ is a Hermitian generator representable semiseparable matrix if and only if $J_{k+1} - \widehat{D}_k$ is so. Moreover, taking $\gamma \neq h_{1,1}$ and $\gamma \neq 0$ we have that $J_{k+1} - \widehat{D}_k$ is invertible and its inverse can be computed as follows:

$$\begin{aligned}\left(H_{k+1}K_{k+1}^{-1} - \widehat{D}_k\right)^{-1} &= \left(-\widehat{D}_k(K_{k+1} - \widehat{D}_k^{-1}H_{k+1})K_{k+1}^{-1}\right)^{-1} = \\ &= -K_{k+1}(K_{k+1} - \widehat{D}_k^{-1}H_{k+1})^{-1}\widehat{D}_k^{-1}.\end{aligned}$$

Since $K_{k+1} = I_{k+1} + D_k H_{k+1}$ where $D_k = \operatorname{diag}(0, \xi_1^{-1}, \ldots, \xi_k^{-1})$, we have

$$\begin{aligned}\left(H_{k+1}K_{k+1}^{-1} - \widehat{D}_k\right)^{-1} &= -K_{k+1}(I_{k+1} + (D_k - \widehat{D}_k^{-1})H_{k+1})^{-1}\widehat{D}_k^{-1} \\ &= -K_{k+1}(I_{k+1} - \frac{1}{\gamma}\boldsymbol{e}_1\boldsymbol{e}_1^T H_{k+1})^{-1}\widehat{D}_k^{-1} \\ &= -K_{k+1}(I_{k+1} + \frac{1}{\gamma - h_{1,1}}\boldsymbol{e}_1\boldsymbol{e}_1^T H_{k+1})\widehat{D}_k^{-1} = \\ &= -(K_{k+1} + \frac{1}{\gamma - h_{1,1}}K_{k+1}(\boldsymbol{e}_1\boldsymbol{e}_1^T)H_{k+1})\widehat{D}_k^{-1}.\end{aligned}$$

The third equality follows from the Sherman-Morrison formula (2.1.1), using the fact that $\gamma \neq h_{1,1}$. This also shows that the matrix $J_{k+1} - \widehat{D}_k$ is indeed invertible. From the expression that we obtained,

we deduce that the inverse of $J_{k+1} - \widehat{D}_k$ is an irreducible tridiagonal matrix since K_{k+1} and H_{k+1} have such a structure and $\gamma \neq 0$. Hence, using Theorem 4.2.4 we conclude that $J_{k+1} - \widehat{D}_k$ is a generator representable semiseparable matrix, and therefore also $J_{k+1} - \widetilde{D}_k$ is so. □

Corollary 4.2.7 below generalizes the statement of Theorem 4.2.5 to the case with poles at ∞. To simplify its proof we introduce the following lemma.

Lemma 4.2.6. Let A be a Hermitian matrix. Fix an integer $i < k$, and let $\{\xi_i^{(j)}\}_{j\in\mathbb{N}}$ be a sequence of real numbers outside of the convex hull of $\Lambda(A)$ that tends to ∞. Let $Q_{k+1}^{(j)}$ be the orthonormal basis computed by the rational Arnoldi algorithm using poles $\{\xi_1, \ldots, \xi_{i-1}, \xi_i^{(j)}, \xi_{i+1}, \ldots, \xi_k\}$ and let $J_{k+1}^{(j)} = (Q_{k+1}^{(j)})^H A Q_{k+1}^{(j)}$ be the associated projected matrix. We have that

$$\lim_{j\to\infty} Q_{k+1}^{(j)} = Q_{k+1} \qquad \text{and} \qquad \lim_{j\to\infty} J_{k+1}^{(j)} = J_{k+1},$$

where Q_{k+1} is the orthonormal basis computed by the rational Arnoldi algorithm with the vector of poles $\{\xi_1, \ldots, \xi_{i-1}, \infty, \xi_{i+1}, \ldots, \xi_k\}$ and $J_{k+1} = Q_{k+1}^H A Q_{k+1}$.

Proof. Denote by $\boldsymbol{q}_\ell$ and $\boldsymbol{q}_\ell^{(j)}$, for $1 \leq \ell \leq k+1$, the columns of Q_{k+1} and $Q_{k+1}^{(j)}$, respectively. Since the first i columns of the matrices $Q_{k+1}^{(j)}$ and Q_{k+1} are the same, we denote them by $\boldsymbol{q}_1, \ldots, \boldsymbol{q}_i$. Let us prove that $\boldsymbol{q}_{i+1}^{(j)}$ converges to $\boldsymbol{q}_{i+1}$ as $j \to \infty$. To compute $\boldsymbol{q}_{i+1}^{(j)}$ by the rational Arnoldi algorithm, we first compute $\boldsymbol{w}_i^{(j)} = (I - A/\xi_i^{(j)})^{-1} A\boldsymbol{q}_i$. Note that this is a continuous operation when $\xi_i^{(j)}$ tends to infinity, indeed we have

$$\lim_{j\to\infty} (I - A/\xi_i^{(j)})^{-1} A\boldsymbol{q}_i = A\boldsymbol{q}_i.$$

Since the orthonormalization of $\boldsymbol{q}_{i+1}^{(j)}$ against $\boldsymbol{q}_1, \ldots, \boldsymbol{q}_i$ is also a continuous operation, we have that

$$\lim_{j\to\infty} \boldsymbol{q}_{i+1}^{(j)} = \boldsymbol{q}_{i+1}.$$

We can now prove by induction that $\lim_{j\to\infty} \boldsymbol{q}_{\ell+1}^{(j)} = \boldsymbol{q}_{\ell+1}$ for all $i+1 \leq \ell \leq k$. By writing the orthogonalization step explicitly, we have

$$\boldsymbol{q}_{\ell+1}^{(j)} = (I - Q_\ell^{(j)}(Q_\ell^{(j)})^H)\boldsymbol{w}_\ell^{(j)}, \qquad \text{where} \qquad \boldsymbol{w}_\ell^{(j)} = (I - A/\xi_\ell)^{-1} A\boldsymbol{q}_\ell^{(j)},$$

and likewise

$$\boldsymbol{q}_{\ell+1} = (I - Q_\ell Q_\ell^H)\boldsymbol{w}_\ell, \qquad \text{where} \qquad \boldsymbol{w}_\ell = (I - A/\xi_\ell)^{-1} A\boldsymbol{q}_\ell.$$

By the inductive hypothesis $\lim_{j\to\infty} Q_\ell^{(j)} = Q_\ell$, so we also have $\lim_{j\to\infty} \boldsymbol{w}_\ell^{(j)} = \boldsymbol{w}_\ell$, and therefore

$$\lim_{j\to\infty} \boldsymbol{q}_{\ell+1}^{(j)} = \boldsymbol{q}_{\ell+1}.$$

The statement for J_{k+1} follows immediately from the definitions of $J_{k+1}^{(j)}$ and J_{k+1}. □

Corollary 4.2.7. Under the same assumptions of Theorem 4.2.5, but allowing some poles to be equal to infinity, the projected matrix $J_{k+1} = Q_{k+1}^H A Q_{k+1}$ is a quasiseparable matrix.

Proof. Let us prove the statement by induction on the number of poles at infinity. If there are no poles equal to infinity the thesis follows from Theorem 4.2.5. Assume now that the statement holds in the case of s poles equal to infinity and assume that we are using $s+1$ infinite poles. Let $\xi_i = \infty$ be the last infinite pole and let $\{\xi_i^{(j)}\}_{j\in\mathbb{N}}$ be a sequence of real numbers outside the convex hull of $\Lambda(A)$ that converges to ξ_i, that is

$$\lim_{j\to\infty} \xi_i^{(j)} = \infty.$$

Let $J_{k+1}^{(j)}$ be the projected matrix obtained by using poles equal to

$$\{\xi_1, \ldots, \xi_{i-1}, \xi_i^{(j)}, \xi_{i+1}, \ldots, \xi_k\}.$$

By Lemma 4.2.6, we have that $J_{k+1}^{(j)}$ converges to the projected matrix J_{k+1} obtained with the poles $\{\xi_1, \ldots, \xi_i, \ldots, \xi_k\}$, that is

$$\lim_{j\to\infty} J_{k+1}^{(j)} = J_{k+1}.$$

From the inductive hypothesis we have that the matrices $J_{k+1}^{(j)}$ are quasiseparable for all j. Since Hermitian quasiseparable matrices are a closed set [152, Theorem 1.30], we have the thesis. □

Notice that, if the matrix A is Hermitian positive semidefinite, the projected matrix $J_k = Q_k^H A Q_k$ has to be positive definite. Indeed, if there exists a vector $\boldsymbol{x} \neq \mathbf{0}$ such that $J_k\boldsymbol{x} = \mathbf{0}$, we have that $AQ_k\boldsymbol{x} = \mathbf{0}$. In particular, since $Q_k\boldsymbol{x} \in q_{k-1}(A)^{-1}\mathcal{K}_k(A, \boldsymbol{b})$, where q_{k-1} is the denominator of the rational Krylov subspace, there exist $\alpha_0, \ldots, \alpha_j \in \mathbb{C}$, for some $j \leq k-1$ and with $\alpha_j \neq 0$, such that

$$Q_k\boldsymbol{x} = q_{k-1}(A)^{-1} \sum_{i=0}^{j} \alpha_i A^i \boldsymbol{b},$$

and so

$$\mathbf{0} = AQ_k\boldsymbol{x} = q_{k-1}(A)^{-1} \sum_{i=0}^{j} \alpha_i A^{i+1} \boldsymbol{b}.$$

This implies that $A^{j+1}\boldsymbol{b} \in \mathcal{K}_j(A, \boldsymbol{b})$, i.e., that $\mathcal{K}_j(A, \boldsymbol{b})$ is an A-invariant subspace. However, this is impossible since we are assuming that the invariance index of the Krylov subspace is larger than $k+1$. In particular, the fact that J_k is positive definite when the matrix A is Hermitian positive semidefinite implies the existence of the Cholesky factorization of J_k.

Going back to the notation used in Section 4.2.2, let Q_k and P_k denote the orthonormal bases of the rational Krylov subspaces $\mathcal{Q}_k(A^HA, \boldsymbol{b}, \boldsymbol{\xi}_k)$ and $\mathcal{Q}_k(AA^H, A\boldsymbol{b}, \boldsymbol{\xi}_k)$, respectively. The projected matrix $B_k = P_k^H A Q_k$ that appears in (4.2.4) is exactly the transpose of the Cholesky factor of the matrix $J_k = Q_k^H(A^HA)Q_k$. Indeed B_k is upper triangular and

$$J_k = (AQ_k)^H AQ_k = (P_kB_k)^H(P_kB_k) = B_k^H B_k,$$

where we have used the fact that $P_kP_k^H AQ_k = AQ_k$, which follows from the fact that $\text{range}(AQ_k) = \mathcal{Q}_k(AA^H, A\boldsymbol{b}, \boldsymbol{\xi}_k)$. The Cholesky factor of a quasiseparable matrix is quasiseparable, see [152, Theorem 2.10] and [152, Proposition 4.16], so we can conclude that B_k is also quasiseparable.

Exploiting the quasiseparable structure of the matrix B_k, we can compute the matrices B_k and P_k by only performing a few scalar products. Indeed, if we let

$$B_k = \begin{bmatrix} d_1 & \beta_1 & \gamma_1 & * & \cdots & * \\ & d_2 & \beta_2 & \gamma_2 & \ddots & \vdots \\ & & \ddots & \ddots & \ddots & * \\ & & & d_{k-2} & \beta_{k-2} & \gamma_{k-2} \\ & & & & d_{k-1} & \beta_{k-1} \\ & & & & & d_k \end{bmatrix},$$

and we define $\boldsymbol{x}_k = [P_{k-1}\,\mathbf{0}]B_k\boldsymbol{e}_k$, we have

$$A\boldsymbol{q}_k = AQ_k\boldsymbol{e}_k = P_kB_k\boldsymbol{e}_k = d_k\boldsymbol{p}_k + \boldsymbol{x}_k.$$

Using the fact that the submatrix of B_k that involves the last two columns and all except for its last two rows has rank at most 1, we can compute $\boldsymbol{x}_k$ with the recursive relation

$$\boldsymbol{x}_k = \frac{\gamma_{k-2}}{\beta_{k-2}}\boldsymbol{x}_{k-1} + \beta_{k-1}\boldsymbol{p}_{k-1}.$$

Algorithm 4.2 k-th step of rational Golub-Kahan algorithm

Input: $A \in \mathbb{C}^{m\times n}$, $\boldsymbol{q}_k \in \mathbb{C}^n$, $\boldsymbol{p}_{k-1}, \boldsymbol{p}_{k-2}, \boldsymbol{x}_{k-1} \in \mathbb{C}^m$.
Output: $\boldsymbol{p}_k, \boldsymbol{x}_k \in \mathbb{C}^m$, $d_k, \beta_{k-1}, \gamma_{k-2} \in \mathbb{C}$.

$\boldsymbol{w} = A\boldsymbol{q}_k$
$\beta_{k-1} = \boldsymbol{w}^H \boldsymbol{p}_{k-1}$
$\gamma_{k-2} = \boldsymbol{w}^H \boldsymbol{p}_{k-2}$
$\boldsymbol{x}_k = \frac{\gamma_{k-2}}{\beta_{k-2}} \boldsymbol{x}_{k-1} + \beta_{k-1}\boldsymbol{p}_{k-1}$
$\boldsymbol{w} = \boldsymbol{w} - \boldsymbol{x}_k$
$d_k = \|\boldsymbol{w}\|_2$
$\boldsymbol{p}_k = \boldsymbol{w}/d_k$

Algorithm 4.3 Short-term recurrence rational Krylov approximation of $f^\diamond(A)\boldsymbol{b}$

Input: $A \in \mathbb{C}^{m\times n}, \boldsymbol{b} \in \mathbb{C}^n$, function f, real poles $\{\xi_1, \dots, \xi_{k-1}\}$.
Output: $\boldsymbol{f}_k^\diamond \in \mathcal{Q}_k(AA^H, A\boldsymbol{b}, \boldsymbol{\xi}_k)$ such that $\boldsymbol{f}_k^\diamond \approx f^\diamond(A)\boldsymbol{b}$, given by (4.2.4).

$\boldsymbol{q}_1 = \boldsymbol{b}/\|\boldsymbol{b}\|_2$
$\boldsymbol{w}_1 = (I - A^H A/\xi_1)^{-1} A^H A\boldsymbol{q}_1$ ▷ other choices can be used
Compute $\boldsymbol{q}_2$ by orthogonalizing $\boldsymbol{w}_1$ against $\boldsymbol{q}_1$.
Compute the QR decomposition $[\boldsymbol{p}_1, \boldsymbol{p}_2] \begin{bmatrix} d_1 & \beta_1 \\ 0 & d_2 \end{bmatrix} = [\boldsymbol{q}_1, \boldsymbol{q}_2]$.
Define $B_2 = \begin{bmatrix} d_1 & \beta_1 \\ 0 & d_2 \end{bmatrix}$ and $\boldsymbol{x}_2 = \beta_1 \boldsymbol{p}_1$.
for $j = 2, \dots, k-1$ **do**
    $\boldsymbol{w}_j = (I - A^H A/\xi_j)^{-1} A^H A\boldsymbol{q}_j$ ▷ other choices can be used
    Compute $\boldsymbol{q}_{j+1}$ by orthogonalizing $\boldsymbol{w}_j$ against $[\boldsymbol{q}_1, \dots, \boldsymbol{q}_j]$.
    Compute $\boldsymbol{p}_{j+1}, d_{j+1}, \beta_j, \gamma_{j-1}, \boldsymbol{x}_{j+1}$ by Algorithm 4.2 with $k = j+1$.
    Set $B_{j+1} = \begin{bmatrix} B_j & \boldsymbol{s}_{j+1} \\ 0 & d_{j+1} \end{bmatrix}$, where $\boldsymbol{s}_{j+1} = \begin{bmatrix} \frac{\gamma_{j-1}}{\beta_{j-1}} (B_j)_{1:j-1,j} \\ \beta_j \end{bmatrix}$.
end for
Set $P_k = [\boldsymbol{p}_1, \dots, \boldsymbol{p}_k]$.
Compute $f^\diamond(B_k)$, e.g. via an SVD of B_k.
Compute $\boldsymbol{f}_k^\diamond = \|\boldsymbol{b}\|_2 P_k f^\diamond(B_k)\boldsymbol{e}_1$.

This allows us to compute $d_k, \beta_{k-1}, \gamma_{k-2}$ and $\boldsymbol{p}_k$ with only two scalar products. The k-th step of this procedure is summarized in Algorithm 4.2.

Notice that during the k-th step of the procedure, we do not require the first $k-1$ columns of Q_k. Moreover, for the computation of the projected solution $\boldsymbol{f}_k^\diamond$ defined in (4.2.4), we do not need the matrix Q_k. For this reason, the computation of the matrix Q_k can be performed by using a short-term recurrence rational Lanczos algorithm, and we can keep in memory only the last two columns of Q_k. After the k-th step of the algorithm, the k-th column of the matrix B_k can be computed using the newly computed quantities and the previous column, by exploiting its rank structure. The whole procedure for the computation of $f^\diamond(A)\boldsymbol{b}$ is summarized in Algorithm 4.3. The algorithm reduces to the standard Golub-Kahan bidiagonalization if all the poles are chosen equal to infinity.

Note that computing the approximation (4.2.4) to $f^\diamond(A)\boldsymbol{b}$ still requires storing the whole basis P_k in order to perform the product $P_k f^\diamond(B_k)\boldsymbol{e}_1$. Nevertheless, the low memory requirements of the short-term recurrence can be exploited in full when the goal is the computation of the bilinear form $\boldsymbol{w}^H f^\diamond(A)\boldsymbol{b}$ for some vector $\boldsymbol{w}$: in this setting, the approximation $\boldsymbol{w}^H P_k f^\diamond(B_k)\boldsymbol{e}_1$ can be computed by updating the vector $\boldsymbol{w}^H P_k$ while the Krylov basis is being constructed, bypassing the need to store the whole matrix P_k. Alternatively, it is possible to compute $\boldsymbol{f}_k^\diamond = \|\boldsymbol{b}\|_2 P_k f^\diamond(B_k)\boldsymbol{e}_1$ without storing the whole basis P_k by using a two-pass approach similar to the one described in [31]: in a first pass one computes the vector $\boldsymbol{y}_k = \|\boldsymbol{b}\|_2 f^\diamond(B_k)\boldsymbol{e}_1$, and then in a second pass the product $\boldsymbol{f}_k^\diamond = P_k \boldsymbol{y}_k$ is formed by computing a linear combination of the columns of P_k as they are generated. Both passes can be performed using a low-memory implementation.

Remark 4.2.8. In Algorithm 4.3 we implicitly assume that $\beta_i \neq 0$ for each i, a hypothesis that is

almost always satisfied in practice. However, it can be shown that there must be at least one nonzero off-diagonal entry in the k-th column of B_k, so the algorithm could be easily modified to avoid the issue of $\beta_i = 0$.

4.3 Error bounds

In this section we prove some error bounds for the approximation of $f^\diamond(A)\boldsymbol{b}$ using polynomial and rational Krylov subspace methods. These bounds link the approximation error with the error of polynomial and rational approximations of f in the uniform norm on an interval containing the singular values of A. The results contained in this section are the analogue of those that hold for standard matrix functions, and they can be proved in a similar way.

We first give an upper and lower bound for the singular values of B_k. For convenience, given a matrix $A \in \mathbb{C}^{m\times n}$, throughout this section we are going to use an extended notation for singular values, defining $\sigma_j := 0$ for all j such that $\min\{m,n\} < j \leq \max\{m,n\}$.

Lemma 4.3.1. Let σ_1 and σ_n be the first and n-th singular value of $A \in \mathbb{C}^{m\times n}$, respectively. Then the singular values of B_k are contained in the interval $[\sigma_n, \sigma_1]$.

Proof. We have

$$B_k^H B_k = Q_k^H A^H P_k P_k^H A Q_k = Q_k^H (A^H A) Q_k, \tag{4.3.1}$$

where we used the fact that $P_k P_k^H A Q_k = A Q_k$, since the columns of AQ_k span the Krylov subspace $\mathcal{Q}_k(AA^H, A\boldsymbol{b})$. Therefore, by the Cauchy Interlacing Theorem the eigenvalues of $B_k^H B_k$ are contained in the interval $[\sigma_n^2, \sigma_1^2]$, which containes the eigenvalues of $A^H A \in \mathbb{C}^{n\times n}$, and hence the singular values of B_k are contained in the interval $[\sigma_n, \sigma_1]$. □

Observe that if $A \in \mathbb{C}^{m\times n}$ is rectangular with $n > m$, we always have $\sigma_n = 0$, and hence B_k may have singular values arbitrarily close to 0 even if $\sigma_{\min\{m,n\}}(A) > 0$. This fact is going to affect the error bounds in Theorem 4.3.3 and Theorem 4.3.7.

Note that such a situation can actually happen in practice. As an example, consider the 1×2 matrix $A = \begin{bmatrix} 1 & 0 \end{bmatrix}$ and the vector $\boldsymbol{b} = \begin{bmatrix} \epsilon \\ 1 \end{bmatrix}$, for small $\epsilon > 0$. For $k = 1$, we have $Q_1 = \boldsymbol{b}/\|\boldsymbol{b}\|_2 = \frac{1}{\sqrt{1+\epsilon^2}}\boldsymbol{b}$, and $P_1 = A\boldsymbol{b}/\|A\boldsymbol{b}\|_2 = 1$. So we have $B_1 = P_1^H A Q_1 = \frac{\epsilon}{\sqrt{1+\epsilon^2}} \in \mathbb{R}^{1\times 1}$, and hence B_1 can have an arbitrarily small singular value even if $\sigma_1(A) = 1$.

4.3.1 Bounds for polynomial Krylov approximation

We first prove the error bounds in the polynomial case. Recall that a polynomial Krylov method computes an approximation to $f^\diamond(A)\boldsymbol{b}$ from the subspace $\mathcal{K}_k(AA^H, A\boldsymbol{b})$ as

$$\boldsymbol{f}_k^\diamond = P_k f^\diamond(B_k) Q_k^H \boldsymbol{b} = \|\boldsymbol{b}\|_2 P_k f^\diamond(B_k)\boldsymbol{e}_1, \tag{4.3.2}$$

where $B_k = P_k^H A Q_k$, and P_k and Q_k are the matrices computed in the Golub-Kahan bidiagonalization of A, satisfying $\operatorname{range}(P_k) = \mathcal{K}_k(AA^H, A\boldsymbol{b})$ and $\operatorname{range}(Q_k) = \mathcal{K}_k(A^H A, \boldsymbol{b})$. A key observation for proving the bounds is the exactness of the approximation (4.3.2) when f is an odd polynomial, stated in the following lemma.

Lemma 4.3.2. Assume that $f = p_{2k-1}$ is an odd polynomial with $\deg p_{2k-1} \leq 2k-1$. Then the approximation $\boldsymbol{f}_k^\diamond$ given by (4.3.2) is exact, i.e., we have $f^\diamond(A)\boldsymbol{b} = \boldsymbol{f}_k^\diamond$.

Proof. Let $p_{2k-1}(z) = zq(z^2)$, where $\deg q \leq k-1$. Using Proposition 4.1.3 and recalling (4.3.1), we have

$$\boldsymbol{f}_k^\diamond = P_k B_k q(B^H B_k) Q_k^H \boldsymbol{b} = P_k P_k^H A Q_k q(Q_k^H A^H A Q_k) Q_k^H \boldsymbol{b}.$$

Due to the exactness property of the polynomial Krylov approximation for standard matrix functions (see Lemma 2.5.2), we have

$$Q_k q(Q_k^H A^H A Q_k) Q_k^H \boldsymbol{b} = q(A^H A)\boldsymbol{b},$$

and hence we get

$$\boldsymbol{f}_k^\diamond = P_k P_k^H A q(A^H A)\boldsymbol{b} = A q(A^H A)\boldsymbol{b} = p_{2k-1}^\diamond(A)\boldsymbol{b},$$

where we used the fact that $\boldsymbol{w} = Aq(A^H A)\boldsymbol{b} \in \mathcal{K}_k(AA^H, A\boldsymbol{b})$, and therefore we have $P_k P_k^H \boldsymbol{w} = \boldsymbol{w}$. □

Lemma 4.3.2 can be used to prove the following theorem.

Theorem 4.3.3. Let $A \in \mathbb{C}^{m\times n}$, and let σ_1, σ_n and σ_m be the first, n-th and m-th singular value of A, respectively. Let $\boldsymbol{f}_k^\diamond$ be the approximation to $f^\diamond(A)\boldsymbol{b}$ given by (4.3.2). Then the following inequality holds:

$$\|f^\diamond(A)\boldsymbol{b} - \boldsymbol{f}_k^\diamond\|_2 \le 2\|\boldsymbol{b}\|_2 \min_{p\in\Pi_{k-1}} \|f(z) - p(z^2)z\|_{[\sigma_n,\sigma_1]}. \tag{4.3.3}$$

Moreover, if A is square with $\sigma_m = \sigma_n > 0$, or if $\lim_{z\to 0} f(z)/z = 0$, we also have

$$\|f^\diamond(A)\boldsymbol{b} - \boldsymbol{f}_k^\diamond\|_2 \le 2\|A\boldsymbol{b}\|_2 \min_{p\in\Pi_{k-1}} \|f(z)/z - p(z^2)\|_{[\sigma_{\max\{m,n\}},\sigma_1]}. \tag{4.3.4}$$

Proof. Let p be a polynomial with $\deg p \le k-1$. Then $p_{2k-1}(z) = p(z^2)z$ is an odd polynomial with $\deg p_{2k-1} \le 2k-1$, and by Lemma 4.3.2 we have

$$p_{2k-1}^\diamond(A)\boldsymbol{b} = P_k p_{2k-1}^\diamond(B_k) Q_k^H \boldsymbol{b}. \tag{4.3.5}$$

By adding and subtracting the quantity in (4.3.5) to $f^\diamond(A)\boldsymbol{b} - \boldsymbol{f}_k^\diamond$, we get

$$f^\diamond(A)\boldsymbol{b} - \boldsymbol{f}_k^\diamond = \big(f^\diamond(A) - p_{2k-1}^\diamond(A)\big)\boldsymbol{b} - P_k\big(f^\diamond(B_k) - p_{2k-1}^\diamond(B_k)\big)Q_k^H\boldsymbol{b}. \tag{4.3.6}$$

By invariance of the 2-norm under orthogonal transformations, we have

$$\|f^\diamond(A) - p_{2k-1}^\diamond(A)\|_2 = \|f - p_{2k-1}\|_{\Sigma(A)} \le \|f - p_{2k-1}\|_{[\sigma_{\min\{m,n\}},\sigma_1]},$$

and similarly, by Lemma 4.3.1,

$$\|f^\diamond(B_k) - p_{2k-1}^\diamond(B_k)\|_2 \le \|f - p_{2k-1}\|_{[\sigma_n,\sigma_1]}.$$

Combining the above inequalities with (4.3.6), we get

$$\|f^\diamond(A)\boldsymbol{b} - \boldsymbol{f}_k^\diamond\|_2 \le 2\|\boldsymbol{b}\|_2 \|f - p_{2k-1}\|_{[\sigma_n,\sigma_1]},$$

and by taking the minimum over $p \in \Pi_{k-1}$ we obtain (4.3.3).

To prove (4.3.4), recall that if $\sigma_m > 0$ or $\lim_{z\to 0} f(z)/z = 0$, by Proposition 4.1.5 we have $f^\diamond(A) = g(AA^H)A$, where $g(z) = f(\sqrt{z})/\sqrt{z}$, and similarly, if $\sigma_n > 0$ or $\lim_{z\to 0} f(z)/z = 0$, then $f^\diamond(B_k) = g(B_k B_k^H)B_k$. Therefore, by also using Proposition 4.1.3, we can rewrite (4.3.6) in the form

$$f^\diamond(A)\boldsymbol{b} - \boldsymbol{f}_k^\diamond = \big(g(AA^H) - p(AA^H)\big)A\boldsymbol{b} - P_k\big(g(B_kB_k^H) - p(B_kB_k^H)\big)B_kQ_k^H\boldsymbol{b}.$$

Given that the eigenvalues of $B_kB_k^H$ are the squares of the singular values of B_k, with a similar argument as before we obtain

$$\begin{aligned}\|f^\diamond(A) - \boldsymbol{f}_k^\diamond\|_2 &\le \|A\boldsymbol{b}\|_2 \Big(\|g-p\|_{[\sigma_m,\sigma_1]} + \|g-p\|_{[\sigma_n^2,\sigma_1^2]}\Big)\\ &\le 2\|A\boldsymbol{b}\|_2 \big\|f(z)/z - p(z^2)\big\|_{[\sigma_{\max\{m,n\}},\sigma_1]}.\end{aligned}$$

As before, (4.3.4) follows by taking the minimum over $p \in \Pi_{k-1}$. □

Remark 4.3.4. Observe that if the matrix A is not square, we have $\sigma_{\max\{m,n\}} = 0$, and hence the bound (4.3.4) always involves a polynomial approximation over the whole interval $[0, \sigma_1]$, even when

$\sigma_{\min\{m,n\}} > 0$. If $A \in \mathbb{C}^{m\times n}$ is rectangular with $m < n$, then the bound (4.3.3) also involves the whole interval $[0, \sigma_1]$. A possible strategy to overcome this issue is to use Proposition 4.1.7 and write

$$\boldsymbol{y} = f^\diamond(A)\boldsymbol{b} = (A^\dagger)^H f^\diamond(A^H)A\boldsymbol{b}.$$

The vector $\boldsymbol{w} = f^\diamond(A^H)A\boldsymbol{b}$ can be approximated using a Krylov method on A^H, and then $\boldsymbol{y}$ can be recovered by solving the least squares problem

$$\boldsymbol{y} = (A^\dagger)^H\boldsymbol{w} = \arg\min_{\boldsymbol{y}} \|A^H\boldsymbol{y} - \boldsymbol{w}\|_2. \tag{4.3.7}$$

By rewriting the problem in this form, if $m < n$ and $\sigma_m > 0$ we get a bound involving approximation on the smaller interval $[\sigma_m, \sigma_1]$ for the approximation of $\boldsymbol{w}$, which translates to a bound for the approximation of $\boldsymbol{y}$.

The bound (4.3.3) can be manipulated to obtain a more explicit bound. Assume that $\sigma_n > 0$, and let $I = [\sigma_n, \sigma_1]$. The polynomial $p(z^2)z$ is odd, and we can assume that f is also odd, so we have

$$\begin{aligned}\min_{p\in\Pi_{k-1}} \|f(z) - p(z^2)z\|_I &= \min_{p\in\Pi_{k-1}} \|f(z) - p(z^2)z\|_{(-I)\cup I} \\ &= \min_{q\in\Pi_{2k-1}} \|f(z) - q(z)\|_{(-I)\cup I},\end{aligned} \tag{4.3.8}$$

where we used the fact that the polynomial of best approximation on $(-I) \cup I$ for an odd function is itself odd. Bounds on the asymptotic rate of convergence for the polynomial approximation of a function on the union of disjoint intervals $(-I) \cup I$ have been developed in [44]. We remark that using [44, Theorem 1] to bound (4.3.8) would lead to the same asymptotic rate of convergence as (4.3.9), with a slightly larger constant.

The bound (4.3.3) can also be related to a polynomial approximation problem on the interval $[\sigma_n^2, \sigma_1^2]$. Indeed, we have

$$\begin{aligned}\min_{p\in\Pi_{k-1}} \|f(z) - p(z^2)z\|_{[\sigma_n,\sigma_1]} &= \sigma_1 \min_{p\in\Pi_{k-1}} \|f(z)/z - p(z^2)\|_{[\sigma_n,\sigma_1]} \\ &\leq \sigma_1 \min_{p\in\Pi_{k-1}} \|f(\sqrt{z})/\sqrt{z} - p(z)\|_{[\sigma_n^2,\sigma_1^2]}.\end{aligned}$$

Let $1 < \rho \leq \dfrac{\sigma_1 + \sigma_n}{\sigma_1 - \sigma_n}$, and denote by E_ρ the ellipse with vertices at $\pm\frac{1}{2}(\rho + \frac{1}{\rho})$ and foci at ± 1, and by $\widetilde{E}_\rho$ its image under the linear function that maps $[-1, 1]$ to the interval $[\sigma_n^2, \sigma_1^2]$. The ellipse $\widetilde{E}_\rho$ has vertices at $\frac{1}{2}(\sigma_n^2 + \sigma_1^2) \pm \frac{1}{4}(\rho + \frac{1}{\rho})(\sigma_n^2 - \sigma_1^2)$ and foci at σ_n^2 and σ_1^2. Note that for $\rho = \dfrac{\sigma_1 + \sigma_n}{\sigma_1 - \sigma_n}$, the ellipse $\widetilde{E}_\rho$ has a vertex at 0. We recall the following classical result from approximation theory.

Theorem 4.3.5 ([147, Theorem 8.2])**.** Let the function g be analytic in the interior of the ellipse $\widetilde{E}_\rho$, and assume that $\max_{z\in\widetilde{E}_\rho} |g(z)| \leq M$. Then

$$\min_{p\in\Pi_k} \|g(z) - p(z)\|_{[\sigma_n^2,\sigma_1^2]} \leq \frac{2M}{\rho - 1}\rho^{-k}.$$

If the function $f(\sqrt{z})/\sqrt{z}$ is analytic in the interior of $\widetilde{E}_\rho$, applying Theorem 4.3.5 to the bound (4.3.3) gives us

$$\begin{aligned}\|f^\diamond(A)\boldsymbol{b} - \boldsymbol{f}_k^\diamond\|_2 &\leq 2\sigma_1\|\boldsymbol{b}\|_2 \min_{p\in\Pi_{k-1}} \|f(\sqrt{z})/\sqrt{z} - p(z)\|_{[\sigma_n^2,\sigma_1^2]} \\ &\leq 4M\sigma_1\|\boldsymbol{b}\|_2 \frac{\rho}{\rho - 1}\rho^{-k},\end{aligned} \tag{4.3.9}$$

where $M = \max_{z\in\widetilde{E}_\rho} |f(\sqrt{z})/\sqrt{z}|$ and $1 < \rho \leq \dfrac{\sigma_1 + \sigma_n}{\sigma_1 - \sigma_n}$.

Remark 4.3.6. Note that if the function $f(\sqrt{z})/\sqrt{z}$ is unbounded on $\widetilde{E}_{\bar{\rho}}$ for a certain $\bar{\rho}$, Theorem 4.3.5 can only be used for $\rho < \bar{\rho}$. In such a situation, the bound (4.3.9) only makes sense for $\rho < \bar{\rho}$.

4.3.2 Bounds for rational Krylov approximation

We next prove similar error bounds for the rational approximation (4.2.4). We start by stating the result analogous to Theorem 4.3.3. Recall that the denominator in the rational Krylov space $\mathcal{Q}_k(AA^H, A\boldsymbol{b}, \boldsymbol{\xi}_k)$ is given by the polynomial $q_{k-1}(z) = \prod_{j=1,\, \xi_j \neq \infty}^{k-1} (z - \xi_j)$.

Theorem 4.3.7. Let $A \in \mathbb{C}^{m\times n}$, and let σ_1, σ_n and σ_m be the first, n-th and m-th singular value of A, respectively. Let $\boldsymbol{f}_k^\diamond$ be the approximation to $f^\diamond(A)\boldsymbol{b}$ from $\mathcal{Q}_k(AA^H, A\boldsymbol{b}, \boldsymbol{\xi}_k)$ given by (4.2.4). Then we have the inequality

$$\|f^\diamond(A)\boldsymbol{b} - \boldsymbol{f}_k^\diamond\|_2 \leq 2\|\boldsymbol{b}\|_2 \min_{p\in\Pi_{k-1}} \|f(z) - q_{k-1}(z^2)^{-1}p(z^2)z\|_{[\sigma_n,\sigma_1]}. \tag{4.3.10}$$

Moreover, if A is square with $\sigma_m = \sigma_n > 0$, or if $\lim_{z\to 0} f(z)/z = 0$, we also have

$$\|f^\diamond(A)\boldsymbol{b} - \boldsymbol{f}_k^\diamond\|_2 \leq 2\|A\boldsymbol{b}\|_2 \min_{p\in\Pi_{k-1}} \|f(z)/z - q_{k-1}(z^2)^{-1}p(z^2)\|_{[\sigma_{\max\{m,n\}},\sigma_1]}. \tag{4.3.11}$$

Note that the same issues discussed after Theorem 4.3.3 in the case of rectangular matrices also arise in the rational case, and the same approach proposed in Remark 4.3.4 can be used to address them.

Similarly to the polynomial case, the bound (4.3.10) can be rewritten by exploiting the fact that f can be assumed to be odd and hence the best approximant on a symmetric interval is odd, yielding

$$\|f^\diamond(A)\boldsymbol{b} - \boldsymbol{f}_k^\diamond\|_2 \leq 2\|\boldsymbol{b}\|_2 \min_{p\in\Pi_{2k-1}} \|f(z) - q_{k-1}(z^2)^{-1}p(z)\|_{(-I)\cup I}, \tag{4.3.12}$$

where again $I = [\sigma_n, \sigma_1]$. However, rational approximation on disjoint intervals is a complicated problem, and hence this formulation might be less useful in practice.

A more practical way to rewrite the bound (4.3.10) is the following:

$$\begin{aligned}\|f^\diamond(A)\boldsymbol{b} - \boldsymbol{f}_k^\diamond\|_2 &\leq 2\|\boldsymbol{b}\|_2 \min_{p\in\Pi_{k-1}} \|f(z) - q_{k-1}(z^2)^{-1}p(z^2)z\|_{[\sigma_n,\sigma_1]} \\ &= 2\|\boldsymbol{b}\|_2 \min_{p\in\Pi_{k-1}} \|\sqrt{z}\big(f(\sqrt{z})/\sqrt{z} - q_{k-1}(z)^{-1}p(z)\big)\|_{[\sigma_n^2,\sigma_1^2]} \\ &\leq 2\sigma_1\|\boldsymbol{b}\|_2 \min_{p\in\Pi_{k-1}} \|\big(f(\sqrt{z})/\sqrt{z} - q_{k-1}(z)^{-1}p(z)\big)\|_{[\sigma_n^2,\sigma_1^2]}.\end{aligned} \tag{4.3.13}$$

Although we get an additional factor σ_1, this bound relates the error to a uniform rational approximation problem on a real interval. This approximation problem is well-studied in the literature, and it is the same that appears when computing the standard matrix function $g(z) = f(\sqrt{z})/\sqrt{z}$ with rational Krylov methods, so it can be a viable tool for selecting good poles.

Similarly to the polynomial case, to prove Theorem 4.3.7 we require the following auxiliary lemma, which shows the exactness of the rational approximation on rational functions of the form $f(z) = q_{k-1}(z^2)^{-1}p(z^2)z$, where p is any polynomial in Π_{k-1}.

Lemma 4.3.8. Assume that f is a rational function of the form $f(z) = q_{k-1}(z^2)^{-1}p(z^2)z$, where $p \in \Pi_{k-1}$. Then the approximation $\boldsymbol{f}_k^\diamond$ from the rational Krylov subspace $\mathcal{Q}_k(AA^H, A\boldsymbol{b}, \boldsymbol{\xi}_k)$ given by (4.2.4) is exact, i.e., we have $f^\diamond(A)\boldsymbol{b} = \boldsymbol{f}_k^\diamond$.

Proof. Using Corollary 4.1.4 and (4.3.1), we have

$$\begin{aligned}\boldsymbol{f}_k^\diamond &= P_k B_k q_{k-1}(B^H B_k)^{-1} p(B_k^H B_k) Q_k^H \boldsymbol{b} \\ &= P_k P_k^H A Q_k q_{k-1}(Q_k^H A^H A Q_k)^{-1} p(Q_k^H A^H A Q_k) Q_k^H \boldsymbol{b}.\end{aligned}$$

Due to the exactness property of the rational Krylov approximation for standard matrix functions (Lemma 2.5.7), we have

$$Q_k q_{k-1}(Q_k^H A^H A Q_k) p(Q_k^H A^H A Q_k) Q_k^H \boldsymbol{b} = q_{k-1}(A^H A)^{-1} p(A^H A)\boldsymbol{b},$$

and hence we get

$$\boldsymbol{f}_k^\diamond = P_k P_k^H A q_{k-1}(A^H A)^{-1} p(A^H A)\boldsymbol{b} = A q_{k-1}(A^H A)^{-1} p(A^H A)\boldsymbol{b} = f^\diamond(A)\boldsymbol{b},$$

where we used the fact that $\boldsymbol{w} = A q_{k-1}(A^H A)^{-1} p(A^H A)\boldsymbol{b} \in \mathcal{Q}(AA^H, A\boldsymbol{b})$, and therefore we have $P_k P_k^H \boldsymbol{w} = \boldsymbol{w}$. □

We are now ready to prove Theorem 4.3.7. The proof follows the same strategy as the proof of Theorem 4.3.3.

Proof of Theorem 4.3.7. Let p be a polynomial with $\deg p \le k-1$. Then the rational function $r(z) = q_{k-1}(z^2)^{-1} p(z^2) z$ satisfies the assumptions of Lemma 4.3.8, and hence we have

$$r^\diamond(A)\boldsymbol{b} = P_k r^\diamond(B_k) Q_k^H \boldsymbol{b}. \tag{4.3.14}$$

By adding and subtracting the quantity in (4.3.14) to $f^\diamond(A)\boldsymbol{b} - \boldsymbol{f}_k^\diamond$, we get

$$f^\diamond(A)\boldsymbol{b} - \boldsymbol{f}_k^\diamond = \big(f^\diamond(A) - r^\diamond(A)\big)\boldsymbol{b} - P_k\big(f^\diamond(B_k) - r^\diamond(B_k)\big) Q_k^H \boldsymbol{b}. \tag{4.3.15}$$

Since by Lemma 4.3.1 the nonzero singular values of B_k are contained in the interval $[\sigma_n, \sigma_1]$, by invariance of the 2-norm under orthogonal transformations we have

$$\begin{aligned} \|f^\diamond(A) - r^\diamond(A)\|_2 &\le \|f - r\|_{[\sigma_{\min\{m,n\}}, \sigma_1]}, \\ \|f^\diamond(B_k) - r^\diamond(B_k)\|_2 &\le \|f - r\|_{[\sigma_n, \sigma_1]}. \end{aligned}$$

Combining the above inequalities with (4.3.15), we obtain

$$\|f^\diamond(A)\boldsymbol{b} - \boldsymbol{f}_k^\diamond\|_2 \le 2\|\boldsymbol{b}\|_2 \|f - r\|_{[\sigma_n, \sigma_1]},$$

and by taking the minimum over $p \in \Pi_{k-1}$ we obtain (4.3.10).

To prove (4.3.11), if $\sigma_n = \sigma_m > 0$ or $\lim_{z\to 0} f(z)/z = 0$, by Proposition 4.1.5 we can write $f^\diamond(A) = g(AA^H)A$ and $f^\diamond(B_k) = g(B_k B_k^H)B_k$, where $g(z) = f(\sqrt{z})/\sqrt{z}$. Thus, using also Corollary 4.1.4, we can rewrite (4.3.15) in the form

$$f^\diamond(A)\boldsymbol{b} - \boldsymbol{f}_k^\diamond = h(AA^H)A\boldsymbol{b} - P_k h(B_k B_k^H) B_k Q_k^H \boldsymbol{b},$$

where $h(z) = g(z) - q_{k-1}(z)^{-1} p(z)$. Given that the eigenvalues of $B_k B_k^H$ are the squares of the singular values of B_k, using Lemma 4.3.1 and proceeding as above we obtain

$$\begin{aligned} \|f^\diamond(A) - \boldsymbol{f}_k^\diamond\|_2 &\le \|A\boldsymbol{b}\|_2 \left(\|h\|_{[\sigma_m^2, \sigma_1^2]} + \|h\|_{[\sigma_n^2, \sigma_1^2]} \right) \\ &= 2\|A\boldsymbol{b}\|_2 \big\| f(z)/z - q_{k-1}(z^2)^{-1} p(z^2) \big\|_{[\sigma_{\max\{m,n\}}, \sigma_1]}. \end{aligned}$$

As before, (4.3.11) follows by taking the minimum over $p \in \Pi_{k-1}$. □

We mention that the results of this section can also be obtained by exploiting the link betweeen GMFs of A and standard functions of the matrix $\mathcal{A} = \begin{bmatrix} 0 & A \\ A^H & 0 \end{bmatrix}$. Indeed, it was observed in [5] that for an odd function f we have

$$f(\mathcal{A}) = \begin{bmatrix} 0 & f^\diamond(A) \\ f^\diamond(A^H) & 0 \end{bmatrix}.$$

If we define the orthogonal matrix $\mathcal{U}_{2k} = \begin{bmatrix} P_k & 0 \\ 0 & Q_k \end{bmatrix}$ and the vector $\boldsymbol{c} = \begin{bmatrix} 0 \\ \boldsymbol{b} \end{bmatrix} \in \mathbb{C}^{m+n}$, we have that $\mathcal{U}_{2k}^H \mathcal{A} \mathcal{U}_{2k} = \begin{bmatrix} 0 & B_k \\ B_k^H & 0 \end{bmatrix}$ and

$$\begin{aligned} f(\mathcal{A})\boldsymbol{c} &= \begin{bmatrix} f^\diamond(A)\boldsymbol{b} \\ 0 \end{bmatrix}, \\ \mathcal{U}_{2k} f(\mathcal{U}_{2k}^H \mathcal{A} \mathcal{U}_{2k}) \mathcal{U}_{2k}^H \boldsymbol{c} &= \begin{bmatrix} P_k f^\diamond(B_k) Q_k^H \boldsymbol{b} \\ 0 \end{bmatrix}. \end{aligned}$$

Moreover, it can be proved that the columns of $\mathcal{U}_{2k}$ are an orthonormal basis for a rational Krylov subspace $\mathcal{Q}_{2k}(\mathcal{A}, \boldsymbol{c})$, whose poles consist of a single pole at ∞, and $\pm\theta_j$, $j = 1, \dots, k-1$, where $\theta_j^2 = \xi_j$ for each j. This fact is straightforward to prove in the polynomial case, where all θ_j are equal to ∞.

An alternative derivation of the error bounds in Theorem 4.3.3 and Theorem 4.3.7 could then be obtained by combining the above fact with Lemma 4.3.1 and error bounds concerning rational Krylov approximation of standard matrix functions, such as Theorem 2.5.8.

4.3.3 An optimal pole for the shift-and-invert method

In this section we use the error bounds in Theorem 4.3.7 combined with a known result from approximation theory to find a pole that optimizes the bounds in the case of a single repeated pole located on the negative real line, which corresponds to the shift-and-invert Krylov method.

We consider the case of a nonsingular square matrix $A \in \mathbb{C}^{n\times n}$, with singular values contained in the interval $[\sigma_{\min}, \sigma_{\max}]$, with $\sigma_{\min} > 0$. With a change of variables, the bound (4.3.11) can be rewritten in the form

$$\|f^\diamond(A)\boldsymbol{b} - \boldsymbol{f}_k^\diamond\|_2 \le 2\|A\boldsymbol{b}\|_2 \min_{p\in\Pi_{k-1}} \|g(z) - q_{k-1}(z)^{-1}p(z)\|_{[\sigma_{\min}^2,\sigma_{\max}^2]},$$

where the function g is defined as $g(z) = f(\sqrt{z})/\sqrt{z}$. In the case of a single repeated pole $\xi < 0$, we have $q_{k-1}(z) = (z-\xi)^{k-1}$ and

$$\Big\{(z-\xi)^{-k+1}p(z) \,:\, p \in \Pi_{k-1}\Big\} = \Big\{p((z-\xi)^{-1}) \,:\, p \in \Pi_{k-1}\Big\}.$$

By defining $h(z) = g(z^{-1} + \xi)$, so that $g(z) = h((z-\xi)^{-1})$, we get

$$\min_{p\in\Pi_{k-1}} \|g(z) - (z-\xi)^{-k+1}p(z)\|_{[\sigma_{\min}^2,\sigma_{\max}^2]} = \min_{p\in\Pi_{k-1}} \|h(z) - p(z)\|_{[\mu_{\min},\mu_{\max}]}, \tag{4.3.16}$$

where $\mu_{\min} := (\sigma_{\max}^2 - \xi)^{-1}$ and $\mu_{\max} := (\sigma_{\min}^2 - \xi)^{-1}$. Notice that for $\xi < 0$ we indeed have $0 < \mu_{\min} \le \mu_{\max} \le -\xi^{-1}$.

The minimum in (4.3.16) can be bounded using the following result in approximation theory, adapted from [120, Proposition 3.1]. Its proof relies on classical bounds for Faber series, see [63, Corollary 2.2].

Proposition 4.3.9. Let $\xi < 0$, and assume that $h(z) = g(z^{-1} + \xi)$ is analytic in the strip $0 < \operatorname{Re} z < -\xi^{-1}$ and continuous in $[0, -\xi^{-1}]$. Then, for any integer $k \ge 1$ the following inequality holds,

$$\min_{p\in\Pi_{k-1}} \|h(z) - p(z)\|_{[\mu_{\min},\mu_{\max}]} \le 2M\frac{\rho^k}{1-\rho}, \tag{4.3.17}$$

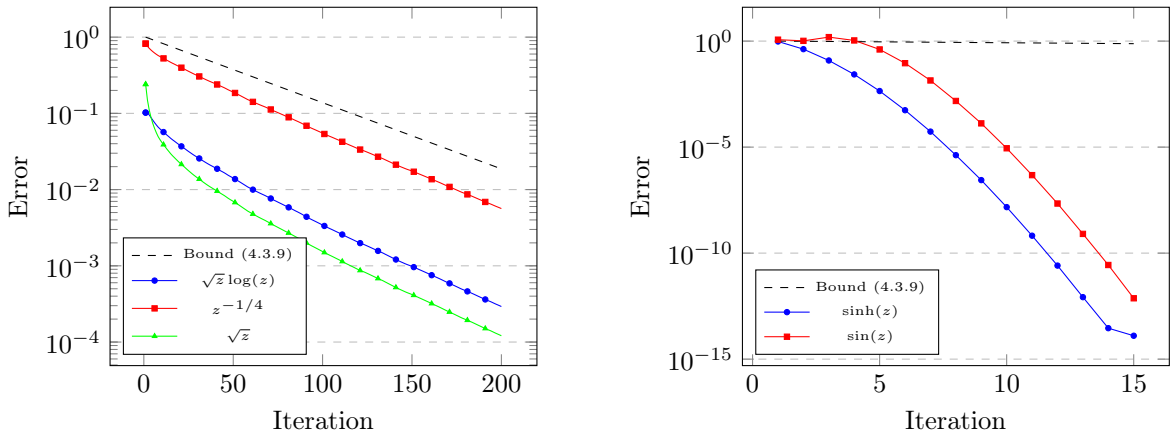
where $M = \|h(z)\|_{[0,-\xi^{-1}]}$ and

$$\rho = \max\left\{\frac{\sqrt{\sigma_{\max}^2 - \xi} - \sqrt{\sigma_{\min}^2 - \xi}}{\sqrt{\sigma_{\max}^2 - \xi} + \sqrt{\sigma_{\min}^2 - \xi}}, \frac{\sigma_{\max}\sqrt{\sigma_{\min}^2 - \xi} - \sigma_{\min}\sqrt{\sigma_{\max}^2 - \xi}}{\sigma_{\max}\sqrt{\sigma_{\min}^2 - \xi} + \sigma_{\min}\sqrt{\sigma_{\max}^2 - \xi}}\right\}.$$

It follows from the analysis after [120, Proposition 3.1] that the bound (4.3.17) is optimized by choosing $\xi = -\sigma_{\min}\sigma_{\max}$. This choice leads to the following bound for the shift-and-invert iterates:

$$\|f^\diamond(A)\boldsymbol{b} - \boldsymbol{f}_k^\diamond\|_2 \le 2\|\boldsymbol{b}\|_2 M\sqrt{\frac{\sigma_{\max}}{\sigma_{\min}}}\exp\left(-2k\sqrt{\frac{\sigma_{\min}}{\sigma_{\max}}}\right). \tag{4.3.18}$$

We remark that the original result (see [120, eq. (3.4)]) exihibited an error like $\exp\left(-2k\sqrt[4]{\frac{a}{b}}\right)$ for a symmetric matrix A with spectrum in $[a, b] \subset (0, +\infty)$, when using the shift-and-invert method with the optimal pole $\xi = -\sqrt{ab}$. The fact that in (4.3.18) we have $\sqrt{\frac{\sigma_{\min}}{\sigma_{\max}}}$ instead of $\sqrt[4]{\frac{\sigma_{\min}}{\sigma_{\max}}}$ is not surprising, since we are essentially applying the result from [120] to the matrix A^HA, whose spectrum is contained in $[\sigma_{\min}^2, \sigma_{\max}^2]$.



(a) (b)

Figure 4.1 – Convergence of the polynomial Krylov method for the approximation of $f^\diamond(A)\boldsymbol{b}$, where A is a 2000×2000 matrix whose singular values are Chebyshev points of the second kind for the interval $[10^{-1}, 10]$, and $\boldsymbol{b}$ is a random vector. Left: functions with an asymptotic convergence rate predicted by the bound (4.3.9). Right: entire functions with fast convergence.

4.4 Numerical experiments

In this section we present some numerical experiments with the purpose of illustrating the error bounds and comparing the different methods proposed in the previous sections. We first test the methods on randomly generated real matrices with a prescribed distribution of singular values, obtained by taking two random orthogonal matrices $U, V \in \mathbb{R}^{n\times n}$ and constructing $A = U\Sigma V^H$, where $\Sigma = \operatorname{diag}(\sigma_1, \ldots, \sigma_n) \in \mathbb{R}^{n\times n}$. The random unitary matrices are obtained by taking a matrix $B \in \mathbb{R}^{n\times n}$ with entries from the normal distribution $\mathcal{N}(0, 1)$ and computing the QR factorization $B = QR$. If the diagonal entries of R are nonnegative, then Q is a random orthogonal matrix from the Haar distribution, a natural uniform probability distribution on the manifold of $n \times n$ orthogonal matrices [144]. In our last experiment we use the adjacency matrix of a directed graph from the Sparse Matrix Collection [51]. For simplicity, we assume that the interval of singular values $[\sigma_n, \sigma_1]$ is known. In a practical situation, one would first need to compute a (rough) approximation of the extremal singular values.

The plots display the relative error $\|f^\diamond(A)\boldsymbol{b} - \boldsymbol{f}_k^\diamond\|_2 / \|f^\diamond(A)\boldsymbol{b}\|_2$, where $\boldsymbol{f}_k^\diamond$ is the approximation defined in (4.2.3) or (4.2.4), depending on the Krylov method that was used.

4.4.1 Accuracy of error bounds

We start by illustrating in Figure 4.1 the sharpness of the bound (4.3.9) for the polynomial Krylov method. Under the assumptions of Theorem 4.3.5, the rate of convergence in the bound only depends on the interval $[\sigma_n, \sigma_1]$, and hence we can expect it to be pessimistic for most functions. Indeed, for entire functions such as $\sinh(z)$ and $\sin(z)$ (see Figure 4.1(b)) the convergence is much faster than the bound (4.3.9); however, the bound can capture the asymptotic rate of convergence for certain functions with lower regularity such as $\sqrt{z}$, for suitable singular value distributions (see Figure 4.1(a)). This implies that, under the same assumptions of Theorem 4.3.5, it is only possible to improve the multiplicative constant in the bound (4.3.9). Note that in Figure 4.1 the multiplicative constant in the bound (4.3.9) was ignored for better visualization.

In Figure 4.2 we compare the convergence of the rational Krylov methods and we test the sharpness of the bounds (4.3.12) and (4.3.18). We use the shift-and-invert Krylov method with the repeated pole $\xi = -\sigma_{\min}\sigma_{\max}$, the extended Krylov method [59], which alternates poles at ∞ with poles at 0, and a general Krylov method with the asymptotically optimal poles for Laplace-Stieltjes functions described in Section 2.5.3. The poles were selected using the interval $[\sigma_n^2, \sigma_1^2]$, with reference to the bound (4.3.13).

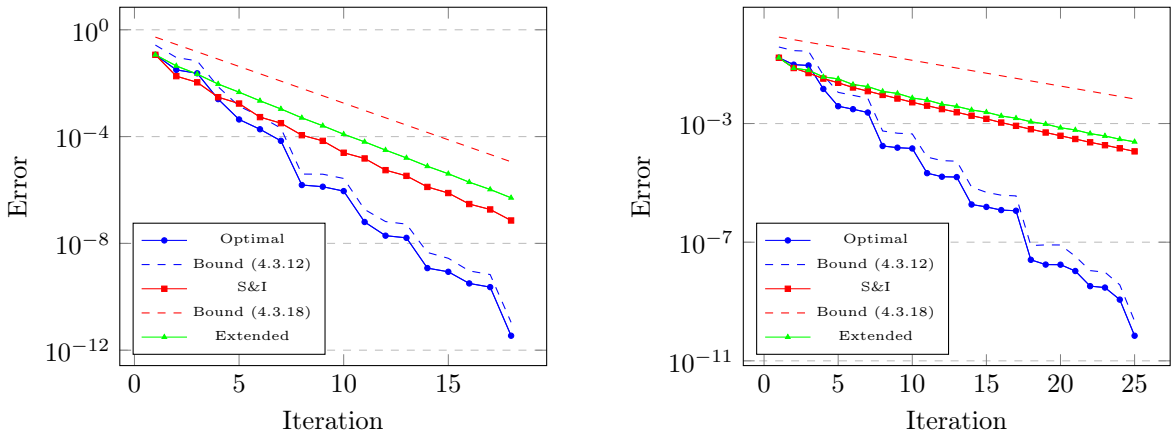


Figure 4.2 – Convergence of different rational Krylov methods for the approximation of $f^\diamond(A)\boldsymbol{b}$, where A is a 2000×2000 matrix with logspaced singular values in the interval $[1, 10]$ (left) or $[10^{-1}, 10]$ (right), $f(z) = \sqrt{z}\log(1+\sqrt{z})$, and $\boldsymbol{b}$ is a random vector.

The function $f(z) = \sqrt{z}\log(1+\sqrt{z})$ is such that the function

$$\frac{f(\sqrt{z})}{\sqrt{z}} = \frac{\log(1+\sqrt[4]{z})}{\sqrt[4]{z}}$$

is Laplace-Stieltjes, or equivalently, completely monotonic [135, Definition 1.3]. This follows from [135, Theorem 3.7] and the fact that $\log(1+z)/z$ is completely monotonic. An approximation from above to the bound (4.3.12) was evaluated using a quasi-optimal polynomial p computed by replacing the uniform norm with the 2-norm on a discrete set of points in $I \cup (-I)$. We can see in Figure 4.2 that the convergence of the rational Krylov method with asymptotically optimal poles closely follows the bound (4.3.12), and that the convergence rate of the shift-and-invert Krylov method is correctly predicted by the bound (4.3.18). The convergence speed of the extended Krylov method is comparable to that of the shift-and-invert Krylov method. Note that, as in the polynomial case, the bound (4.3.18) displayed in Figure 4.2 does not include the multiplicative constant.

4.4.2 Rectangular case

We next investigate the performance of the methods in the case of a rectangular matrix $A \in \mathbb{R}^{m\times n}$, and in particular the effectiveness of the strategy proposed in Remark 4.3.4 when $m < n$ to reduce the computation of $f^\diamond(A)\boldsymbol{b}$ to the computation of $f^\diamond(A^T)A\boldsymbol{b}$. We report in Figure 4.3 the convergence plots of the rational Krylov method with asymtotically optimal poles, for the functions $f(z) = \sqrt{z}$ and $f(z) = z\log(z)$. We can observe that the convergence is similar for the function $z\log(z)$ (Figure 4.3(b)), while there is a large benefit in using the alternative expression (4.3.7) in the case of the function $f(z) = \sqrt{z}$. This is likely due to the fact that $\sqrt{z}$ has a significantly larger derivative close to zero, and hence roundoff errors in the smallest computed singular values of the matrix B_k are extremely amplified when applying this function. Indeed, even though both functions have a derivative that diverges for $z \to 0$, the derivative of $f(z) = \sqrt{z}$ grows like $O(1/\sqrt{z})$ for small z, while the derivative of $f(z) = z\log(z)$ grows like $O(\log(z))$, and this is reflected, for instance, in the fact that for $z = 10^{-16}$ we have $\sqrt{z} = 10^{-8}$ and $z\log(z) \approx -3.7 \cdot 10^{-15}$.

We can see in Figure 4.3(a) that for the function $f(z) = \sqrt{z}$ it is not possible to get below a relative accuracy of 10^{-8} if we directly approximate $f^\diamond(A)\boldsymbol{b}$, while we can reach a relative error of about 10^{-13} if we use the connection with $f^\diamond(A^T)A\boldsymbol{b}$, since in this case the projected matrix B_k has no singular values close to zero.

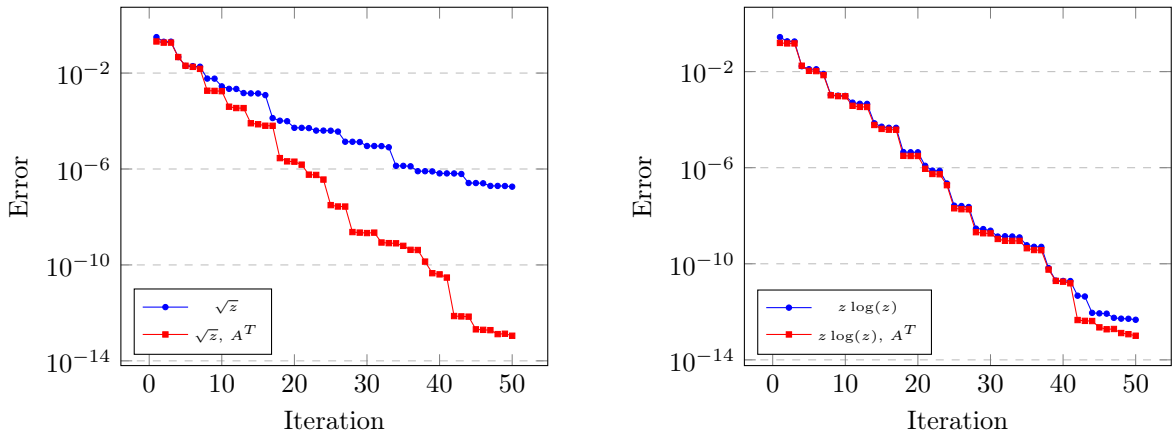


Figure 4.3 – Convergence of the rational Krylov methods with asymptotically optimal poles for the approximation of $f^\diamond(A)\boldsymbol{b}$, where A is a rectangular 1000×1500 matrix whose singular values are Chebyshev points of the second kind for the interval $[10^{-2}, 10]$. The red line shows the convergence of the method described in Remark 4.3.4, which computes $f^\diamond(A)\boldsymbol{b}$ by first computing $f^\diamond(A^T)A\boldsymbol{b}$ and then solving a least squares problem.

4.4.3 Finite precision issues

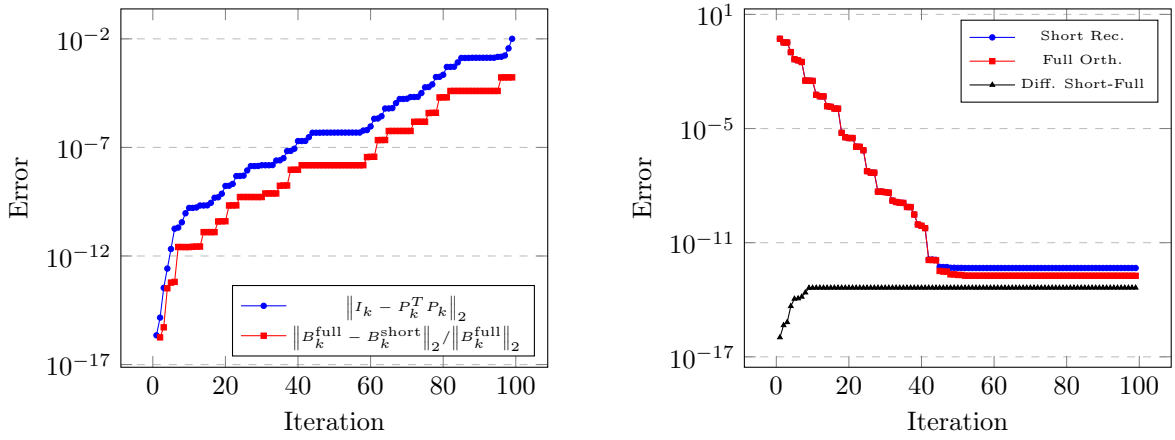
In finite precision, one of the main practical problems of Krylov methods based on a short-term recurrence (such as, for instance, the Lanczos method) is the loss of orthogonality in the computed basis vectors. This phenomenon has been studied for the polynomial Lanczos algorithm in [126]. A brief study of the problem for rational Lanczos can be found in [128]. As one may expect, the algorithm presented in Section 4.2.3 also suffers from this numerical instability. However, our experiments show that this loss of orthogonality deteriorates only slightly the accuracy of the algorithm: if the poles are chosen to guarantee a moderate number of iterations for convergence, it appears that the error produced by comparing the short-term recurrence algorithm with the one that uses full ortogonalization remains rather small, and it stops growing after a few iterations (see Figure 4.4). The effect of finite precision arithmetic and the subsequent loss of orthogonality in the Krylov basis have been already studied in [121] for the approximation of the product between a standard matrix function and a vector by means of the polynomial Lanczos algorithm; on the other hand, a theoretical analysis of the loss of orthogonality in the short-recurrence rational Lanczos algorithm, even in the context of standard matrix functions, is a challenging problem: see for example [128, Section 4].

4.4.4 A practical example

In our last experiment we compare the accuracy and execution times of polynomial and rational Krylov methods in a more practical scenario. We consider the computation of $f^\diamond(A)\boldsymbol{b}$, where A is the 8490×8490 adjacency matrix of the largest strongly connected component of the directed graph `p2p-Gnutella30` from the Sparse Matrix Collection [51] and $\boldsymbol{b}$ is the vector of all ones. This kind of expression arises when computing functions of the adjacency matrix of the associated bipartite graph. We consider the functions $f_1(z) = \sinh(z)$ and $f_2(z) = z^{1/3}$. The function f_1 is an entire function that appears in the computation of hub and authority communicabilities of nodes in a directed graph [5, 15], and the function f_2 is such that $f_2(\sqrt{z})/\sqrt{z}$ is Cauchy-Stieltjes, which allows us to use the asymptotically optimal poles from [114].

In order to simulate a real situation where the exact solution is not available, we used as a simple stopping criterion the relative difference between two consecutive approximations, i.e., we stopped the algorithm as soon as

$$\frac{\|\boldsymbol{f}_k^\diamond - \boldsymbol{f}_{k+1}^\diamond\|_2}{\|\boldsymbol{f}_k^\diamond\|_2} \leq \texttt{tol}, \tag{4.4.1}$$



(a) (b)

Figure 4.4 – Effects of the loss of orthogonality in the rational Golub-Kahan algorithm for the approximation of $f^\diamond(A)\boldsymbol{b}$, where $f(z) = \sqrt{z}$ and A is a 2000×2000 matrix with logspaced singular values in the interval $[10^{-1}, 10^2]$, for the rational Krylov method with asymptotically optimal poles. Left: loss of orthogonality and error in the projected matrix when using the short-term recurrence. Right: comparison of the error in the approximation of $f^\diamond(A)\boldsymbol{b}$ when using the short-term recurrence or full orthogonalization of the basis vectors. In black we reported the norm of the difference between the two approximations.

Table 4.1 – Number of iterations k, execution time t_k in seconds required to achieve tolerance $\texttt{tol} = 10^{-9}$, and actual error E_k at iteration k. The execution times are for the short-term recurrence implementations, obtained as an average over 10 runs. In the case of $f_2(z) = z^{1/3}$, after 2000 iterations of the polynomial method the error estimate was still `1.99e-03`.

function	polynomial			rational		
	k	t_k	E_k	k	t_k	E_k
$\sinh(z)$	11	0.0103	1.54e-10	55	10.0469	6.41e-08
$z^{1/3}$	2000	12.6563	2.48e-03	32	5.6455	1.16e-09

where `tol` is the requested error tolerance. We remark that although this kind of stopping criterion is widely used in practice, it does not provide any guarantees and it often underestimates the actual relative error, especially if the method is stagnating. Our results are summarized in Figure 4.5 and Table 4.1.

Since f_1 is an entire function, the polynomial method converges very quickly and thus outperforms any rational Krylov method. We remark that the number of iterations of the rational Krylov method for f_1 could likely be reduced with a more careful choice of poles; however, it would still be outperformed by the polynomial Krylov method in terms of execution time. On the other hand, the function f_2 is less regular, and the performance of the rational Krylov method with asymptotically optimal poles is much more competitive, both in terms of number of iterations and execution time. We can see that the error estimate (4.4.1) is quite accurate when the method converges quickly, but it stops the algorithm too early if the convergence is slower.

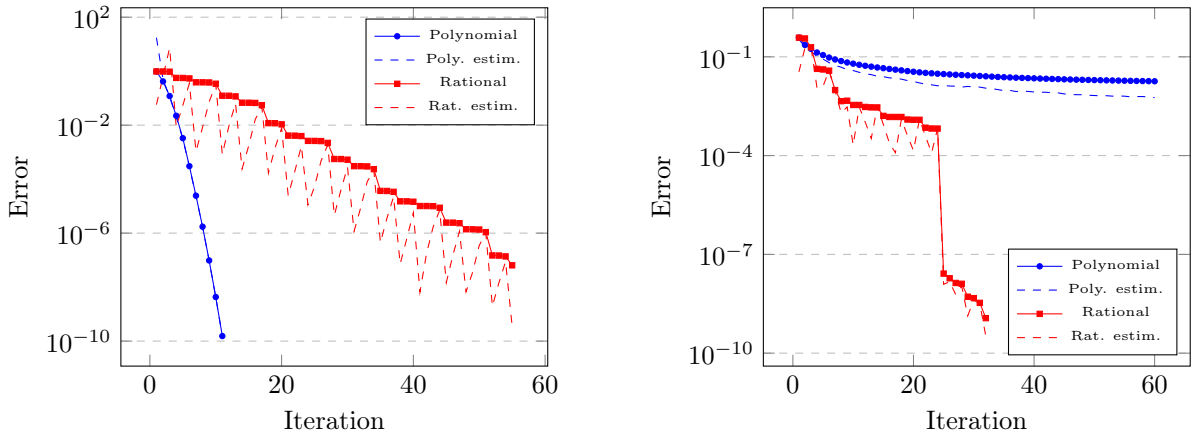


Figure 4.5 – Comparison between polynomial and rational Krylov methods for the approximation of $f^\diamond(A)\boldsymbol{b}$, where A is the 8490×8490 adjacency matrix of the largest strongly connected component of the directed graph `p2p-Gnutella30` and $\boldsymbol{b}$ is the vector of all ones. We have set `tol` $= 10^{-9}$ in the stopping criterion (4.4.1). The dashed lines are the error estimates (4.4.1). Left: $f_1(z) = \sinh(z)$. Right: $f_2(z) = z^{1/3}$.

Chapter 5

A low-memory Lanczos method with rational Krylov compression

In this chapter we propose a new low-memory algorithm for the approximation of $f(A)\boldsymbol{b}$ in the case of Hermitian A, that combines an outer (polynomial or rational) Lanczos iteration with an inner rational Krylov subspace, which is used to compress the outer Krylov basis whenever it reaches a certain size. Similarly to [36], the inner rational Krylov subspace does not involve the matrix A, but only small matrices. This is a key observation, since constructing a basis of the inner subspace does not require the solution of linear systems with A, and hence it is cheap compared to the cost of the outer Lanczos iteration. The approximate solutions computed by the proposed algorithm coincide with the ones constructed by using the outer Krylov subspace method when f is a rational function, and for a general function they differ by a quantity that depends on the best rational approximant of f with the poles used in the inner rational Krylov subspace. In order to obtain a meaningful advantage when compressing the basis, this algorithm should be used when the outer Krylov method requires many iterations to converge, so polynomial Lanczos is an ideal candidate. Other potential options are rational Lanczos methods with repeated poles, such as the extended [59] or shift-and-invert Krylov methods [119].

If the outer Krylov basis is compressed every m iterations and the inner rational Krylov subspace uses k poles, the approach that we describe in this chapter requires the storage of approximately $m+k$ vectors. Additionally, due to the basis compression, our approximation involves the computation of matrix functions of size at most $(m+k) \times (m+k)$, so the cost of this operation does not grow with the number of iterations. This represents an important advantage with respect to the Lanczos method, since when the number of iterations is relatively large the evaluation of f on the projected matrix can become quite expensive.

We start by giving a brief overview of the literature on low-memory methods for the approximation of $f(A)\boldsymbol{b}$ in Section 5.1, and in Section 5.2 we present some results regarding rational Krylov subspaces and the computation of low-rank updates of matrix functions, which will be employed in the rest of the chapter. We then describe the low-memory algorithm in Section 5.3 and we analyze it and compare it with other low-memory methods from the literature in Section 5.4. The algorithm is tested on several problems from selected applications in Section 5.5, showing competitive performance with respect to existing low-memory methods. The contents of this chapter are based on the preprint [38].

5.1 Overview of low-memory Krylov subspace methods

In the discussion that follows, we will often need to refer to specific blocks of a Krylov subspace basis or of the associated projected matrix. In order to do this while keeping the notation as simple as possible, throughout this chapter we are often going to denote Krylov subspace matrices using bold letters, such as $\mathbf{Q}_m$, $\mathbf{H}_m$ and $\mathbf{K}_m$. This will allow us, for instance, to use the notation $Q_1, \dots, Q_s$ to refer to specific blocks of the matrix $\mathbf{Q}_m$. More details on this notation will be given in Section 5.3.

Recall that when A is Hermitian, the orthonormal basis $\mathbf{Q}_m = [\boldsymbol{q}_1 \ \dots \ \boldsymbol{q}_m]$ of the polynomial

Krylov subspace $\mathcal{K}_m(A, \boldsymbol{b})$ can be constructed with the Lanczos algorithm, by exploiting a short-term recurrence. The product $f(A)\boldsymbol{b}$ can then be approximated by the Lanczos approximation

$$\boldsymbol{f}_m := \mathbf{Q}_m f(\mathbf{T}_m)\boldsymbol{e}_1\|\boldsymbol{b}\|_2, \qquad \mathbf{T}_m := \mathbf{Q}_m^H A \mathbf{Q}_m.$$

Since in the Lanczos algorithm each new basis vector is orthogonalized only against the last two basis vectors, we need to keep in memory only three vectors in order to compute the basis $\mathbf{Q}_m$ and the projected matrix $\mathbf{T}_m$. However, forming the approximate solution $\boldsymbol{f}_m$ in a straightforward way still requires the full basis $\mathbf{Q}_m$. When the matrix A is very large, there may be a limit on the maximum number of basis vectors that can be stored, so with a standard implementation of the Lanczos method there is a limit on the number of iterations of Lanczos that can be performed and hence on the attainable accuracy in the approximation of $f(A)\boldsymbol{b}$. In the literature, several strategies have been proposed to deal with memory limitations. See for instance the recent surveys [91, 92] for a comparison of several low-memory methods.

A simple but effective approach is the two-pass Lanczos method [31, 79]. With this approach, the Lanczos method is first run once to determine the projected matrix $\mathbf{T}_m$ and compute the short vector $\boldsymbol{t}_m = f(\mathbf{T}_m)\boldsymbol{e}_1\|\boldsymbol{b}\|_2$. After $\boldsymbol{t}_m$ has been computed, the Lanczos method is run for a second time to form the product $\boldsymbol{f}_m = \mathbf{Q}_m\boldsymbol{t}_m$ as the columns of $\mathbf{Q}_m$ are computed. This method doubles the number of matrix-vector products with A with respect to standard Lanczos, but it requires the storage of only the last three basis vectors.

Another possibility is the multishift conjugate gradient method [73, 150]. This method is based on an explicit approximation of f with a rational function r expressed in partial fraction form. Then, $f(A)\boldsymbol{b} \approx r(A)\boldsymbol{b}$ is approximated by using the conjugate gradient method to solve each of the linear systems that appear in the partial fraction representation of $r(A)\boldsymbol{b}$. This can be done efficiently by exploiting the shift invariance of Krylov subspaces, i.e., the fact that $\mathcal{K}_m(A, \boldsymbol{b}) = \mathcal{K}_m(A + \theta I, \boldsymbol{b})$ for any $\theta \in \mathbb{C}$, in order to use a single Krylov subspace to obtain approximate solutions to all the linear systems simultaneously. Compared to Lanczos, this method requires performing additional vector operations and storing vectors proportionally to the number of poles of the rational approximant r.

When f is a Cauchy-Stieltjes function, it is possible to use a restarting strategy that is similar to restarted Krylov subspace methods for linear systems [61, 71, 72]. By exploiting the Cauchy-Stieltjes integral representation of the function f, the error after a certain number of Lanczos iterations can be written as $f(A)\boldsymbol{b} - \boldsymbol{f}_m = f_m(A)\boldsymbol{q}_{m+1}$, where f_m is still a Cauchy-Stieltjes function that depends on f and $\mathbf{T}_m$. This property makes it possible to restart the Lanczos method after a certain number of iterations, and then iteratively approximate the error using the same method.

Recently, a low-memory method has been proposed to compute an approximation from a Krylov subspace to $f(A)\boldsymbol{b}$ when f is a rational function [41]. This approximation is optimal in a norm that depends on the denominator of the rational function. If d is the degree of the denominator of f, the approximation from $\mathcal{K}_m(A, \boldsymbol{b})$ can be computed with $m + d$ matrix-vector products with A, while storing approximately $2d$ vectors. This method can be extended to non-rational functions f by means of rational approximations, and it often produces approximations that are comparable or better than the Lanczos approximation.

5.2 Necessary tools

In this section we recall some results on rational Krylov subspaces and on low-rank updates of matrix functions that we are going to employ in the derivation of our algorithm in Section 5.3.

5.2.1 Properties of rational Krylov subspaces

In this chapter we are going to employ the following definition of a rational Krylov subspace, which is slightly different from Definition 2.5.5.

Definition 5.2.1. Let $A \in \mathbb{C}^{n\times n}$, $\boldsymbol{b} \in \mathbb{C}^n$ and let $\boldsymbol{\xi}_k$ be a list of poles $\{\xi_0, \dots, \xi_{k-1}\} \subset \overline{\mathbb{C}}$ which are not eigenvalues of A. We define a rational Krylov subspace as

$$\mathcal{Q}_k(A, \boldsymbol{b}, \boldsymbol{\xi}_k) := \Big\{ q_k(A)^{-1}p(A)\boldsymbol{b}, \quad \text{with } p \in \Pi_{k-1} \Big\},$$

where

$$q_k(z) := \prod_{\substack{\xi \in \boldsymbol{\xi}_k \\ \xi \neq \infty}} (z - \xi).$$

Remark 5.2.2. The key difference between Definitions 2.5.5 and 5.2.1 is that in Definition 5.2.1 we do not prescribe that $\boldsymbol{b}$ is contained in the rational Krylov subspace. As a consequence, the denominator polynomial q_k has degree k instead of $k-1$ (if all the poles are finite), and there is an additional pole ξ_0 at the beginning of the pole sequence $\boldsymbol{\xi}_k$. If we take $\xi_0 = \infty$ in Definition 5.2.1, we recover the rational Krylov subspace from Definition 2.5.5. This alternative definition is the one used in [9], and allows us to use some of their results without modification, thus simplifying the exposition in the sections that follow.

Recall that when A is Hermitian and the poles $\boldsymbol{\xi}_k$ are real we can construct a basis $\mathbf{Q}_k$ of the rational Krylov subspace $\mathcal{Q}_k(A, \boldsymbol{b}, \boldsymbol{\xi}_k)$ by employing the rational Lanczos algorithm (Algorithm 2.3), which also computes the tridiagonal matrices

$$\underline{\mathbf{H}}_{k-1} = \begin{bmatrix} \alpha_1 & \beta_1 & & \\ \beta_1 & \ddots & \ddots & \\ & \ddots & \ddots & \beta_{k-2} \\ & & \beta_{k-2} & \alpha_{k-1} \\ & & & \beta_{k-1} \end{bmatrix} \quad \text{and} \quad \underline{\mathbf{K}}_{k-1} = \begin{bmatrix} 1 & & & \\ & \ddots & & \\ & & & 1 \\ 0 & \dots & & 0 \end{bmatrix} + \begin{bmatrix} \xi_0^{-1} & & & \\ & \ddots & & \\ & & & \xi_{k-1}^{-1} \end{bmatrix} \underline{\mathbf{H}}_{k-1},$$

with the convention that the inverse of an infinite pole equals zero. Together with $\mathbf{Q}_k$, the matrices $\underline{\mathbf{H}}_{k-1}$ and $\underline{\mathbf{K}}_{k-1}$ satisfy the rational Arnoldi relation

$$A\mathbf{Q}_k\underline{\mathbf{K}}_{k-1} = \mathbf{Q}_k\underline{\mathbf{H}}_{k-1}. \tag{5.2.1}$$

Note that even with the different definition of $\mathcal{Q}_k(A, \boldsymbol{b}, \boldsymbol{\xi}_k)$ given in Definition 5.2.1, we can still use Algorithm 2.3 to generate the basis by simply giving ξ_0 as input instead of setting it to ∞ in line 1.

Another important ingredient for rational Krylov methods is the computation of the projected matrix $\mathbf{T}_k = \mathbf{Q}_k^H A \mathbf{Q}_k$. We have already seen that we can avoid the explicit computation of $\mathbf{T}_k$ by exploiting the Arnoldi relation (5.2.1) if we assume that $\xi_{k-1} = \infty$. In this case, the last row of $\underline{\mathbf{K}}_{k-1}$ is zero and the matrix $\mathbf{K}_{k-1}$ is invertible, so by writing

$$\mathbf{T}_k = \begin{bmatrix} \mathbf{T}_{k-1} & \boldsymbol{b}_k \\ \boldsymbol{b}_k^H & * \end{bmatrix},$$

with $\mathbf{T}_{k-1} \in \mathbb{C}^{(k-1)\times(k-1)}$ and $\boldsymbol{b}_k \in \mathbb{R}^{k-1}$, and multiplying (5.2.1) on the right by $\mathbf{K}_{k-1}^{-1}$ and on the left by $\mathbf{Q}_k^H$, we obtain

$$\mathbf{T}_{k-1} = \mathbf{H}_{k-1}\mathbf{K}_{k-1}^{-1} \quad \text{and} \quad \boldsymbol{b}_k = \beta_{k-1}\mathbf{K}_{k-1}^{-H}\boldsymbol{e}_{k-1}. \tag{5.2.2}$$

If we additionally assume that there exists $s < k-1$ such that $\xi_s = \infty$, then the matrix $\mathbf{K}_{k-1}$ has the block triangular structure

$$\mathbf{K}_{k-1} = \begin{bmatrix} \mathbf{K}_s & \xi_{s-1}^{-1}\beta_{s-1}\boldsymbol{e}_s\boldsymbol{e}_1^T \\ & \widehat{\mathbf{K}}_{k-1} \end{bmatrix}, \quad \text{and} \quad \mathbf{K}_{k-1}^{-1} = \begin{bmatrix} \mathbf{K}_s^{-1} & -\xi_{s-1}^{-1}\beta_{s-1}\mathbf{K}_s^{-1}\boldsymbol{e}_s\boldsymbol{e}_1^T\widehat{\mathbf{K}}_{k-1}^{-1} \\ & \widehat{\mathbf{K}}_{k-1}^{-1} \end{bmatrix}$$

with $\mathbf{K}_s \in \mathbb{C}^{s\times s}$ and $\widehat{\mathbf{K}}_{k-1} \in \mathbb{C}^{(k-1-s)\times(k-1-s)}$. Since $\mathbf{T}_{k-1}$ is Hermitian and $\boldsymbol{e}_1^T\widehat{\mathbf{K}}_{k-1} = \boldsymbol{e}_1^T$, exploiting (5.2.2) we have

$$\mathbf{T}_{k-1} = \begin{bmatrix} \mathbf{T}_s & \boldsymbol{b}_{s+1}\boldsymbol{e}_1^T \\ \boldsymbol{e}_1\boldsymbol{b}_{s+1}^H & \widehat{\mathbf{T}}_{k-1} \end{bmatrix}, \quad \widehat{\mathbf{T}}_{k-1} = \widehat{\mathbf{H}}_{k-1}\widehat{\mathbf{K}}_{k-1}^{-1} - \xi_{s-1}^{-1}\beta_{s-1}^2(\boldsymbol{e}_s^T\mathbf{K}_s^{-1}\boldsymbol{e}_s)\boldsymbol{e}_1\boldsymbol{e}_1^T, \tag{5.2.3}$$

with $\mathbf{T}_s \in \mathbb{C}^{s\times s}$ and $\boldsymbol{b}_{s+1} \in \mathbb{C}^s$ defined according to (5.2.2). Note that only the $(1,1)$ entry of $\widehat{\mathbf{T}}_{k-1}$ is different from the corresponding element of $\widehat{\mathbf{H}}_{k-1}\widehat{\mathbf{K}}_{k-1}^{-1}$.

We conclude this section with a simple result that will be useful in Section 5.2.2.

Proposition 5.2.3. Let $A \in \mathbb{C}^{n\times n}$, $\boldsymbol{b} \in \mathbb{C}^n$ and let $\boldsymbol{\xi}_k$ be a sequence of k poles. Denoting by $\mathbf{Q}_k$ an orthonormal basis of $\mathcal{Q}_k(A, \boldsymbol{b}, \boldsymbol{\xi}_k)$, for any matrix U with orthonormal columns such that range$(\mathbf{Q}_k) \subseteq$ range(U) we have

$$U\mathcal{Q}_k(U^HAU, U^H\boldsymbol{b}, \boldsymbol{\xi}_k) = \mathcal{Q}_k(A, \boldsymbol{b}, \boldsymbol{\xi}_k).$$

Proof. We first prove this result in the polynomial case, that is when all the poles are equal to infinity, and then we generalize the proof to the rational case.

We are going to prove by induction on $h \leq k$ that

$$U(U^HAU)^hU^H\boldsymbol{b} = UU^HA^h\boldsymbol{b},$$

which in particular implies

$$U\sum_{j=0}^{k}\gamma_j(U^HAU)^jU^H\boldsymbol{b} = UU^H\sum_{j=0}^{k}\gamma_jA^j\boldsymbol{b}, \tag{5.2.4}$$

for any choice of coefficients $\gamma_j \in \mathbb{C}$. Note that this is a slightly stronger result that, since range$(\mathbf{Q}_k) \subseteq$ range(U), for $h < k$ reduces to the thesis in the polynomial Krylov case. If $h = 0$ the statement is straightforward. For a fixed $1 \leq h \leq k$, we have

$$U(U^HAU)^hU^H\boldsymbol{b} = UU^HAU(U^HAU)^{h-1}U^H\boldsymbol{b} = UU^HAUU^HA^{h-1}\boldsymbol{b} = UU^HA^h\boldsymbol{b},$$

where the second equality follows from the inductive hypothesis and the third one is due to

$$A^{h-1}\boldsymbol{b} \in \mathcal{K}_k(A, \boldsymbol{b}) \subseteq \text{range}(U).$$

For the rational Krylov case, note that

$$\mathcal{Q}_k(A, \boldsymbol{b}, \boldsymbol{\xi}_k) = \mathcal{K}_k(A, q(A)^{-1}\boldsymbol{b}),$$

where $q(x) := \prod_{\xi\in\boldsymbol{\xi}_k, \xi\neq\infty}(x - \xi)$. In particular, denoting by $\boldsymbol{c} := q(A)^{-1}\boldsymbol{b}$, we deduce from the polynomial case that

$$U\mathcal{K}_k(U^HAU, U^H\boldsymbol{c}) = \mathcal{K}_k(A, \boldsymbol{c}) = \mathcal{Q}_k(A, \boldsymbol{b}, \boldsymbol{\xi}_k),$$

and hence to conclude it is sufficient to prove that $U^H\boldsymbol{c} = q(U^HAU)^{-1}U^H\boldsymbol{b}$. Since the degree of q is bounded by k, by (5.2.4) we have

$$UU^H\boldsymbol{b} = UU^Hq(A)\boldsymbol{c} = Uq(U^HAU)U^H\boldsymbol{c},$$

and multiplying both sides on the left by $q(U^HAU)^{-1}U^H$ we conclude. □

5.2.2 Low-rank updates of matrix functions

In this section we recall a result from [9] on the computation of a low-rank update of a matrix function, and we slightly modify it to fit our purposes.

Theorem 5.2.4 ([9, Theorem 3.3])**.** Let $A \in \mathbb{C}^{n\times n}$ be a Hermitian matrix, $B = [\boldsymbol{b}_1\ \boldsymbol{b}_2] \in \mathbb{C}^{n\times 2}$ and $\Delta \in \mathbb{C}^{2\times 2}$ Hermitian. Let $r = p/q$ be a rational function with complex conjugate roots and poles, and denote by $\boldsymbol{\xi}_k$ the sequence of cardinality $k := \max\{\deg p + 1, \deg q\}$ that contains the poles of r and infinity in the remaining elements. We have

$$r(A + B\Delta B^H) - r(A) = \mathbf{Q}_k\left(r(\mathbf{Q}_k^H(A + B\Delta B^H)\mathbf{Q}_k) - r(\mathbf{Q}_k^HA\mathbf{Q}_k)\right)\mathbf{Q}_k^H,$$

where $\mathbf{Q}_k$ is an orthonormal basis of the subspace $\mathcal{Q}_k(A, \boldsymbol{b}_1, \boldsymbol{\xi}_k) + \mathcal{Q}_k(A, \boldsymbol{b}_2, \boldsymbol{\xi}_k)$.

The following proposition extends Theorem 5.2.4 to the case where $\mathbf{Q}_k$ is replaced by a matrix U such that range$(U) \supseteq \mathcal{Q}_k(A, \boldsymbol{b}_1, \boldsymbol{\xi}_k) + \mathcal{Q}_k(A, \boldsymbol{b}_2, \boldsymbol{\xi}_k)$.

Proposition 5.2.5. Let A, B, Δ, r and $\boldsymbol{\xi}_k$ be defined as in Theorem 5.2.4. Then, we have

$$r(A + B\Delta B^H) - r(A) = U\left(r(U^H(A + B\Delta B^H)U) - r(U^H AU)\right)U^H,$$

where U is a matrix with orthonormal columns such that $\text{range}(U) \supseteq \mathcal{Q}_k(A, \boldsymbol{b}_1, \boldsymbol{\xi}_k) + \mathcal{Q}_k(A, \boldsymbol{b}_2, \boldsymbol{\xi}_k)$.

Proof. First of all, we note that since $\text{range}(U) \supseteq \mathcal{Q}_k(A, \boldsymbol{b}_1, \boldsymbol{\xi}_k) + \mathcal{Q}_k(A, \boldsymbol{b}_2, \boldsymbol{\xi}_k)$ we have

$$UU^H\mathbf{Q}_k = \mathbf{Q}_k,$$

where $\mathbf{Q}_k$ is an orthonormal basis of $\mathcal{Q}_k(A, \boldsymbol{b}_1, \boldsymbol{\xi}_k) + \mathcal{Q}_k(A, \boldsymbol{b}_2, \boldsymbol{\xi}_k)$. Moreover, by Theorem 5.2.4 we have

$$r(A + B\Delta B^H) - r(A) = \mathbf{Q}_k Y_k(r)\mathbf{Q}_k^H = UU^H\mathbf{Q}_k Y_k(r)\mathbf{Q}_k^H UU^H,$$

with

$$Y_k(r) := r(\mathbf{Q}_k^H(A + B\Delta B^H)\mathbf{Q}_k) - r(\mathbf{Q}_k^H A\mathbf{Q}_k).$$

To conclude it is sufficient to prove that

$$U^H\mathbf{Q}_k Y_k(r)\mathbf{Q}_k^H U = r(U^H(A + B\Delta B^H)U) - r(U^H AU).$$

It follows from Proposition 5.2.3 that $U^H\mathbf{Q}_k$ is an orthonormal basis of the subspace

$$\mathcal{Q}_k(U^H AU, U^H\boldsymbol{b}_1, \boldsymbol{\xi}_k) + \mathcal{Q}_k(U^H AU, U^H\boldsymbol{b}_2, \boldsymbol{\xi}_k) = U^H(\mathcal{Q}_k(A, \boldsymbol{b}_1, \xi_k) + \mathcal{Q}_k(A, \boldsymbol{b}_2, \xi_k)),$$

and by Theorem 5.2.4 we obtain

$$\begin{aligned}
&r(U^H(A + B\Delta B^H)U) - r(U^H AU)\\
&\quad= U^H\mathbf{Q}_k\left(r(\mathbf{Q}_k^H UU^H(A + B\Delta B^H)UU^H\mathbf{Q}_k) - r(\mathbf{Q}_k^H UU^H AUU^H\mathbf{Q}_k)\right)\mathbf{Q}_k^H U\\
&\quad= U^H\mathbf{Q}_k(r(\mathbf{Q}_k^H(A + B\Delta B^H)\mathbf{Q}_k) - r(\mathbf{Q}_k^H A\mathbf{Q}_k))\mathbf{Q}_k^H U\\
&\quad= U^H\mathbf{Q}_k Y_k(r)\mathbf{Q}_k^H U.
\end{aligned}$$

□

The following immediate corollaries are going to be fundamental in Section 5.3 for deriving the low-memory algorithm for $f(A)\boldsymbol{b}$.

Corollary 5.2.6. Assume that $r = p/q$ and $\boldsymbol{\xi}_k$ are defined as in Theorem 5.2.4 and let $A \in \mathbb{C}^{n\times n}$ be a Hermitian matrix, partitioned as

$$A = \begin{bmatrix} A_{11} & \\ & A_{22} \end{bmatrix} + \begin{bmatrix} & A_{12} \\ A_{12}^H & \end{bmatrix},$$

with $A_{11} \in \mathbb{C}^{m\times m}$. Assume that $A_{12} = \boldsymbol{b}\boldsymbol{c}^H$, where $\boldsymbol{b} \in \mathbb{C}^m$ and $\boldsymbol{c} \in \mathbb{C}^{n-m}$. Then

$$r(A) = \begin{bmatrix} r(A_{11}) & \\ & r(A_{22}) \end{bmatrix} + \begin{bmatrix} U_k & \\ & I \end{bmatrix} X_k(r) \begin{bmatrix} U_k^H & \\ & I \end{bmatrix}, \tag{5.2.5}$$

where U_k is an orthonormal basis of $\mathcal{Q}_k(A_{11}, \boldsymbol{b}, \boldsymbol{\xi}_k)$, I is the $(n-m)\times(n-m)$ identity matrix and

$$X_k(r) = r\left(\begin{bmatrix} U_k^H A_{11}U_k & U_k^H\boldsymbol{b}\boldsymbol{c}^H \\ \boldsymbol{c}\boldsymbol{b}^H U_k & A_{22} \end{bmatrix}\right) - \begin{bmatrix} r(U_k^H A_{11}U_k) & \\ & r(A_{22}) \end{bmatrix}.$$

In particular, for any vector of the form $\begin{bmatrix} \boldsymbol{v} \\ \mathbf{0} \end{bmatrix}$ where $\boldsymbol{v} \in \mathbb{C}^m$, we have

$$r(A)\begin{bmatrix} \boldsymbol{v} \\ \mathbf{0} \end{bmatrix} = \begin{bmatrix} \boldsymbol{y} \\ \mathbf{0} \end{bmatrix} + \begin{bmatrix} \boldsymbol{w} \\ \mathbf{0} \end{bmatrix} + \boldsymbol{r}, \tag{5.2.6}$$

where

$$\boldsymbol{y} = r(A_{11})\boldsymbol{v}, \qquad \boldsymbol{w} = -U_k r(U_k^H A_{11}U_k)U_k^H\boldsymbol{v},$$

and

$$\boldsymbol{r} = \begin{bmatrix} U_k & \\ & I \end{bmatrix} r\left(\begin{bmatrix} U_k^H A_{11}U_k & U_k^H\boldsymbol{b}\boldsymbol{c}^H \\ \boldsymbol{c}\boldsymbol{b}^H U_k & A_{22} \end{bmatrix}\right)\begin{bmatrix} U_k^H\boldsymbol{v} \\ 0 \end{bmatrix}.$$

Proof. We can write

$$\begin{bmatrix} & A_{12} \\ A_{12}^H & \end{bmatrix} = \begin{bmatrix} \boldsymbol{b} & \\ & \boldsymbol{c} \end{bmatrix} \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{b}^H & \\ & \boldsymbol{c}^H \end{bmatrix}.$$

Due to the block diagonal structure, we have

$$\mathcal{Q}_k\left(\begin{bmatrix} A_{11} & \\ & A_{22} \end{bmatrix}, \begin{bmatrix} \boldsymbol{b} \\ 0 \end{bmatrix}, \boldsymbol{\xi}_k\right) + \mathcal{Q}_k\left(\begin{bmatrix} A_{11} & \\ & A_{22} \end{bmatrix}, \begin{bmatrix} 0 \\ \boldsymbol{c} \end{bmatrix}, \boldsymbol{\xi}_k\right) = \operatorname{range}\left(\begin{bmatrix} U_k & \\ & V_k \end{bmatrix}\right),$$

where V_k is an orthonormal basis of $\mathcal{Q}(A_{22}, \boldsymbol{c}, \boldsymbol{\xi}_k)$. Since

$$\left(\begin{bmatrix} U_k & \\ & V_k \end{bmatrix}\right) \subseteq \operatorname{range}\left(\begin{bmatrix} U_k & \\ & I \end{bmatrix}\right),$$

we can use Proposition 5.2.5 to obtain (5.2.5), and then (5.2.6) follows as an immediate consequence. □

Corollary 5.2.7. Let A be as in Corollary 5.2.6 and let f be a function that is analytic on $\mathbb{W}(A)$. Given complex conjugate poles $\boldsymbol{\xi}_k \subset \overline{\mathbb{C}} \setminus \Lambda(A)$, let U_k be an orthonormal basis of $\mathcal{Q}_k(A_{11}, \boldsymbol{b}, \boldsymbol{\xi}_k)$. We have

$$f(A) - \begin{bmatrix} f(A_{11}) & \\ & f(A_{22}) \end{bmatrix} - \begin{bmatrix} U_k & \\ & I \end{bmatrix} X_k(f) \begin{bmatrix} U_k^H & \\ & I \end{bmatrix} = E_k(f),$$

with

$$\|E_k(f)\|_2 \leq 4 \min_{r \in \Pi_{k-1}/q} \|f - r\|_{\mathbb{W}(A)}, \tag{5.2.7}$$

where $q(x) = \prod_{\xi \in \boldsymbol{\xi}_k, \xi \neq \infty}(x - \xi)$.

Proof. The bound (5.2.7) easily follows from the exactness property of Corollary 5.2.6 for rational functions, with the same proof as [9, Theorem 4.5]. □

5.3 Algorithm description

In this section we present a low-memory implementation of a Krylov subspace method for the computation of $f(A)\boldsymbol{b}$ for a Hermitian matrix A and a vector $\boldsymbol{b}$. On a high level, we use an outer Krylov subspace, which can be either polynomial or rational, combined with an inner rational Krylov subspace that is employed to compress the outer Krylov subspace basis and reduce the memory usage. The inner Krylov subspace is constructed using the projection of A onto the outer Krylov subspace, so it does not involve any expensive operations with the matrix A. This approach is designed for scenarios where the outer Krylov method requires a large number of iterations to converge, in order to take full advantage of the basis compressions. The outer iterations should be cheaper than the solution of shifted linear systems involving A and inner poles, otherwise it would be more efficient to simply use a rational Krylov method associated with the inner poles directly on the matrix A. Therefore, an ideal choice for the outer subspace is a polynomial Krylov subspace, and other viable options are rational Krylov methods with a few repeated poles, such as extended [59] and shift-and-invert [119] Krylov methods.

The algorithm is composed of s cycles, with each cycle consisting in m iterations of the outer Krylov method, and a subsequent compression of the basis to k vectors. At any given moment, the algorithm keeps in memory at most $m + k$ basis vectors and some additional quantities (whose storage is not dependent on n), such as projected matrices of size $m + k$. In total, the algorithm performs $M = k + ms$ iterations of the outer Krylov subspace method, and the approximation computed at the end of each cycle coincides with the approximation computed by a standard implementation of the outer Krylov method, up to an error due to the rational approximation done in the inner Krylov subspace, which is usually negligible. This error is zero in the case where f is a rational function of type $(k-1, k)$, so for simplicity we start by describing the algorithm in this special case.

Let r be a rational function of type $(k-1, k)$, i.e., $r(z) = p(z)/q(z)$ with p and q polynomials of degrees at most $k-1$ and k, respectively. We assume that r has complex conjugate roots and poles. We let $\boldsymbol{\xi}_k$ be the vector of poles $\{\xi_0, \ldots, \xi_{k-1}\}$ containing the roots of q and ∞ in the remaining entries in the case $\deg(q) < k$. These poles will be used for the inner rational Krylov iteration. Our goal is to

construct an approximation of $r(A)\boldsymbol{b}$ from the outer Krylov subspace $\mathcal{Q}_M(A, \boldsymbol{b}, \boldsymbol{\theta}_M)$, where $M = k+ms$ and $\boldsymbol{\theta}_M$ is the associated sequence of complex conjugate poles $\{\theta_0, \theta_1, \ldots, \theta_{M-1}\} \subset \overline{\mathbb{C}}$. We assume that $\theta_0 = \infty$, and that $\theta_{jm+k} = \infty$ for $j = 1, \ldots, s-1$. We denote by $\boldsymbol{\theta}_\ell$ the list $\boldsymbol{\theta}_\ell = [\theta_0, \ldots, \theta_{\ell-1}]$, for $\ell < M$. Throughout this section, we are going to denote by $\mathbf{Q}_j$, $\underline{\mathbf{H}}_j$, $\underline{\mathbf{K}}_j$ and $\mathbf{T}_j$ the matrices associated with the outer Krylov subspace $\mathcal{Q}_j(A, \boldsymbol{b}, \boldsymbol{\theta}_j)$, according to the notation introduced in Section 5.2.1.

We also introduce some additional notation for the matrices associated with the outer Krylov subspace $\mathcal{Q}_M(A, \boldsymbol{b}, \boldsymbol{\theta}_M)$, in order to simplify the exposition in this section. We denote by $\widehat{Q} := \mathbf{Q}_M$ the orthonormal basis of $\mathcal{Q}_M(A, \boldsymbol{b}, \boldsymbol{\theta}_M)$ and by $\widehat{T} := \widehat{Q}^H A \widehat{Q} = \mathbf{T}_M$. We denote the approximation (2.5.6) to $r(A)\boldsymbol{b}$ from $\mathcal{Q}_M(A, \boldsymbol{b}, \boldsymbol{\theta}_M)$ by

$$\widehat{\boldsymbol{y}} := \widehat{Q}\, r(\widehat{T})\boldsymbol{e}_1 \|\boldsymbol{b}\|_2. \tag{5.3.1}$$

We split $\widehat{Q} = \begin{bmatrix} Q_1 & Q_2 & \ldots & Q_s \end{bmatrix}$, with $Q_1 \in \mathbb{C}^{n\times(k+m)}$ and $Q_i \in \mathbb{C}^{n\times m}$ for $i = 2, \ldots, s$. We also use the notation $\widehat{Q}_i = \begin{bmatrix} Q_i & \ldots & Q_s \end{bmatrix}$, so that we have $\widehat{Q}_i = \begin{bmatrix} Q_i & \widehat{Q}_{i+1} \end{bmatrix}$ for $i = 1, \ldots, s-1$. We introduce a similar notation for the matrix $\widehat{T}$. We denote by T_i the i-th block on the block diagonal of $\widehat{T}$, with $T_1 \in \mathbb{C}^{(k+m)\times(k+m)}$ and $T_i \in \mathbb{C}^{m\times m}$ for $i = 2, \ldots, s$, and by $\widehat{T}_{i+1} \in \mathbb{C}^{(s-i)m\times(s-i)m}$ the trailing block on the diagonal after T_i. So we have

$$\widehat{T} = \begin{bmatrix} T_1 & \boldsymbol{b}_1\boldsymbol{e}_1^T \\ \boldsymbol{e}_1\boldsymbol{b}_1^H & \widehat{T}_2 \end{bmatrix}, \qquad \widehat{T}_i = \begin{bmatrix} T_i & \boldsymbol{b}_i\boldsymbol{e}_1^T \\ \boldsymbol{e}_1\boldsymbol{b}_i^H & \widehat{T}_{i+1} \end{bmatrix}, \quad 2 \le i \le s-1,$$

where $\boldsymbol{b}_1 \in \mathbb{C}^{m+k}$, $\boldsymbol{b}_i \in \mathbb{C}^m$ for $i = 1, \ldots, s-1$, and $\widehat{T}_s = T_s$. The block structure of $\widehat{T}$ is a consequence of the poles at infinity in $\boldsymbol{\theta}_M$, as shown in Section 5.2.1 (see (5.2.3)).

5.3.1 The first cycle

We can split $\widehat{T}$ as

$$\widehat{T} = \begin{bmatrix} T_1 & \\ & \widehat{T}_2 \end{bmatrix} + \begin{bmatrix} & \boldsymbol{b}_1\boldsymbol{e}_1^T \\ \boldsymbol{e}_1\boldsymbol{b}_1^H & \end{bmatrix},$$

where the first term is block diagonal and second one has rank two. Let $U_1 \in \mathbb{C}^{(m+k)\times k}$ be an orthonormal basis of the rational Krylov subspace $\mathcal{Q}_k(T_1, \boldsymbol{b}_1, \boldsymbol{\xi}_k)$. Using Corollary 5.2.6, we can write

$$r(\widehat{T})\boldsymbol{e}_1\|\boldsymbol{b}\|_2 = r\left(\begin{bmatrix} T_1 & \boldsymbol{b}_1\boldsymbol{e}_1^T \\ \boldsymbol{e}_1\boldsymbol{b}_1^H & \widehat{T}_2 \end{bmatrix}\right)\begin{bmatrix} \boldsymbol{e}_1\|\boldsymbol{b}\|_2 \\ \boldsymbol{0} \end{bmatrix} = \begin{bmatrix} \widetilde{\boldsymbol{y}}_1 \\ \boldsymbol{0} \end{bmatrix} + \begin{bmatrix} \widetilde{\boldsymbol{w}}_1 \\ \boldsymbol{0} \end{bmatrix} + \widetilde{\boldsymbol{r}}_1,$$

where

$$\widetilde{\boldsymbol{y}}_1 = r(T_1)\boldsymbol{e}_1\|\boldsymbol{b}\|_2, \qquad \widetilde{\boldsymbol{w}}_1 = -U_1 r(U_1^H T_1 U_1) U_1^H \boldsymbol{e}_1\|\boldsymbol{b}\|_2,$$

and

$$\widetilde{\boldsymbol{r}}_1 = \begin{bmatrix} U_1 & \\ & I \end{bmatrix} r\left(\begin{bmatrix} U_1^H T_1 U_1 & U_1^H\boldsymbol{b}_1\boldsymbol{e}_1^T \\ \boldsymbol{e}_1\boldsymbol{b}_1^H U_1 & \widehat{T}_2 \end{bmatrix}\right)\begin{bmatrix} U_1^H\boldsymbol{e}_1\|\boldsymbol{b}\|_2 \\ 0 \end{bmatrix}.$$

Recalling the notation introduced above, we have

$$\widehat{Q}r(\widehat{T})\boldsymbol{e}_1\|\boldsymbol{b}\|_2 = \begin{bmatrix} Q_1 & \widehat{Q}_2 \end{bmatrix}\left(\begin{bmatrix} \widetilde{\boldsymbol{y}}_1 \\ \boldsymbol{0} \end{bmatrix} + \begin{bmatrix} \widetilde{\boldsymbol{w}}_1 \\ \boldsymbol{0} \end{bmatrix} + \widetilde{\boldsymbol{r}}_1\right) = Q_1\widetilde{\boldsymbol{y}}_1 + Q_1\widetilde{\boldsymbol{w}}_1 + \begin{bmatrix} Q_1 & \widehat{Q}_2 \end{bmatrix}\widetilde{\boldsymbol{r}}_1.$$

To summarize, we have obtained

$$\widehat{\boldsymbol{y}} = \widehat{Q}\, r(\widehat{T})\boldsymbol{e}_1\|\boldsymbol{b}\|_2 = \boldsymbol{y}_1 + \boldsymbol{w}_1 + \boldsymbol{r}_1, \tag{5.3.2}$$

where

$$\boldsymbol{y}_1 = Q_1 r(T_1)\boldsymbol{e}_1\|\boldsymbol{b}\|_2$$

is the approximation (2.5.6) to $r(A)\boldsymbol{b}$ from $\mathcal{Q}_{k+m}(A, \boldsymbol{b}, \boldsymbol{\theta}_{k+m})$,

$$\boldsymbol{w}_1 = -Q_1 U_1 r(U_1^H T_1 U_1) U_1^H \boldsymbol{e}_1\|\boldsymbol{b}\|_2$$

and

$$\boldsymbol{r}_1 = \begin{bmatrix} Q_1U_1 & \widehat{Q}_2 \end{bmatrix} r\left(\begin{bmatrix} U_1^H T_1 U_1 & U_1^H \boldsymbol{b}_1 \boldsymbol{e}_1^T \\ \boldsymbol{e}_1 \boldsymbol{b}_1^H U_1 & \widehat{T}_2 \end{bmatrix} \right) \begin{bmatrix} U_1^H \boldsymbol{e}_1 \|\boldsymbol{b}\|_2 \\ 0 \end{bmatrix}. \tag{5.3.3}$$

After the first $k+m$ iterations of the outer Krylov subspace method, we can compute $\boldsymbol{y}_1$ and $\boldsymbol{w}_1$, but the remainder $\boldsymbol{r}_1$ still involves $\widehat{Q}_2$ and $\widehat{T}_2$, which have not been computed yet. A crucial observation that allows us to save memory is that in (5.3.3) the basis Q_1 only appears in the product Q_1U_1, so from now on it is no longer necessary to keep in memory the whole matrix Q_1.

Introducing the notation $V_1 := Q_1$, $S_1 := T_1$, $\boldsymbol{v}_1 := \boldsymbol{e}_1\|\boldsymbol{b}\|_2$ and $\widetilde{\boldsymbol{b}}_1 := \boldsymbol{b}_1$, we can rewrite (5.3.3) as

$$\boldsymbol{r}_1 = \begin{bmatrix} V_1U_1 & \widehat{Q}_2 \end{bmatrix} r\left(\begin{bmatrix} U_1^H S_1 U_1 & U_1^H \widetilde{\boldsymbol{b}}_1 \boldsymbol{e}_1^T \\ \boldsymbol{e}_1 \widetilde{\boldsymbol{b}}_1^H U_1 & \widehat{T}_2 \end{bmatrix} \right) \begin{bmatrix} U_1^H \boldsymbol{v}_1 \\ 0 \end{bmatrix}. \tag{5.3.4}$$

The notation introduced for (5.3.4) sets us up for describing the i-th cycle of the algorithm.

5.3.2 The i-th cycle

At the beginning of the i-th cycle, with $2 \le i \le s-1$, our task is to compute the remainder $\boldsymbol{r}_{i-1}$ from the previous cycle, which is given by

$$\boldsymbol{r}_{i-1} = \begin{bmatrix} V_{i-1}U_{i-1} & \widehat{Q}_i \end{bmatrix} r\left(\begin{bmatrix} U_{i-1}^H S_{i-1} U_{i-1} & U_{i-1}^H \widetilde{\boldsymbol{b}}_{i-1} \boldsymbol{e}_1^T \\ \boldsymbol{e}_1 \widetilde{\boldsymbol{b}}_{i-1}^H U_{i-1} & \widehat{T}_i \end{bmatrix} \right) \begin{bmatrix} U_{i-1}^H \boldsymbol{v}_{i-1} \\ 0 \end{bmatrix}. \tag{5.3.5}$$

Note that (5.3.4) coincides with (5.3.5) with $i = 2$. Expanding $\widehat{T}_i$ in terms of T_i, $\boldsymbol{b}_i$ and $\widehat{T}_{i+1}$, we have

$$\begin{bmatrix} U_{i-1}^H S_{i-1} U_{i-1} & U_{i-1}^H \widetilde{\boldsymbol{b}}_{i-1} \boldsymbol{e}_1^T \\ \boldsymbol{e}_1 \widetilde{\boldsymbol{b}}_{i-1}^H U_{i-1} & \widehat{T}_i \end{bmatrix} = \begin{bmatrix} U_{i-1}^H S_{i-1} U_{i-1} & U_{i-1}^H \widetilde{\boldsymbol{b}}_{i-1} \boldsymbol{e}_1^T & \\ \boldsymbol{e}_1 \widetilde{\boldsymbol{b}}_{i-1}^H U_{i-1} & T_i & \boldsymbol{b}_i \boldsymbol{e}_1^T \\ & \boldsymbol{e}_1 \boldsymbol{b}_i^H & \widehat{T}_{i+1} \end{bmatrix}.$$

Introducing the notation $V_i := \begin{bmatrix} V_{i-1}U_{i-1} & Q_i \end{bmatrix} \in \mathbb{C}^{n \times (k+m)}$, $\boldsymbol{v}_i := \begin{bmatrix} U_{i-1}^H \boldsymbol{v}_{i-1} \\ 0 \end{bmatrix} \in \mathbb{C}^{k+m}$, $\widetilde{\boldsymbol{b}}_i := \begin{bmatrix} 0 \\ \boldsymbol{b}_i \end{bmatrix} \in \mathbb{C}^{k+m}$ and

$$S_i := \begin{bmatrix} U_{i-1}^H S_{i-1} U_{i-1} & U_{i-1}^H \widetilde{\boldsymbol{b}}_{i-1} \boldsymbol{e}_1^T \\ \boldsymbol{e}_1 \widetilde{\boldsymbol{b}}_{i-1}^H U_{i-1} & T_i \end{bmatrix},$$

we can rewrite (5.3.5) as

$$\boldsymbol{r}_{i-1} = \begin{bmatrix} V_i & \widehat{Q}_{i+1} \end{bmatrix} r\left(\begin{bmatrix} S_i & \widetilde{\boldsymbol{b}}_i \boldsymbol{e}_1^T \\ \boldsymbol{e}_1 \widetilde{\boldsymbol{b}}_i^H & \widehat{T}_{i+1} \end{bmatrix} \right) \begin{bmatrix} \boldsymbol{v}_i \\ 0 \end{bmatrix}. \tag{5.3.6}$$

The right-hand side of (5.3.6) can be computed with the same strategy that was used in the first cycle to compute $\widehat{Q} r(\widehat{T}) \boldsymbol{e}_1 \|\boldsymbol{b}\|_2$. Letting $U_i \in \mathbb{C}^{(k+m)\times k}$ be an orthonormal basis of $\mathcal{Q}_k(S_i, \widetilde{\boldsymbol{b}}_i, \boldsymbol{\xi}_k)$, we use Corollary 5.2.6 once again to obtain

$$r\left(\begin{bmatrix} S_i & \widetilde{\boldsymbol{b}}_i \boldsymbol{e}_1^T \\ \boldsymbol{e}_1 \widetilde{\boldsymbol{b}}_i^H & \widehat{T}_{i+1} \end{bmatrix} \right) \begin{bmatrix} \boldsymbol{v}_i \\ 0 \end{bmatrix} = \begin{bmatrix} \widetilde{\boldsymbol{y}}_i \\ \boldsymbol{0} \end{bmatrix} + \begin{bmatrix} \widetilde{\boldsymbol{w}}_i \\ \boldsymbol{0} \end{bmatrix} + \widetilde{\boldsymbol{r}}_i,$$

where

$$\widetilde{\boldsymbol{y}}_i = r(S_i)\boldsymbol{v}_i, \qquad \widetilde{\boldsymbol{w}}_i = -U_i r(U_i^H S_i U_i) U_i^H \boldsymbol{v}_i,$$

and

$$\widetilde{\boldsymbol{r}}_i = \begin{bmatrix} U_i & \\ & I \end{bmatrix} r\left(\begin{bmatrix} U_i^H S_i U_i & U_i^H \widetilde{\boldsymbol{b}}_i \boldsymbol{e}_1^T \\ \boldsymbol{e}_1 \widetilde{\boldsymbol{b}}_i^H U_i & \widehat{T}_{i+1} \end{bmatrix} \right) \begin{bmatrix} U_i^H \boldsymbol{v}_i \\ 0 \end{bmatrix}.$$

From this, we easily obtain

$$\boldsymbol{r}_{i-1} = V_i \widetilde{\boldsymbol{y}}_i + V_i \widetilde{\boldsymbol{w}}_i + \begin{bmatrix} V_i & \widehat{Q}_{i+1} \end{bmatrix} \widetilde{\boldsymbol{r}}_i = V_i r(S_i) \boldsymbol{v}_i + \boldsymbol{w}_i + \boldsymbol{r}_i, \tag{5.3.7}$$

where

$$\boldsymbol{w}_i = -V_i U_i r(U_i^H S_i U_i) U_i^H \boldsymbol{v}_i$$

and

$$\boldsymbol{r}_i = \begin{bmatrix} V_i & \widehat{Q}_{i+1} \end{bmatrix} \widetilde{\boldsymbol{r}}_i = \begin{bmatrix} V_i U_i & \widehat{Q}_{i+1} \end{bmatrix} r\left(\begin{bmatrix} U_i^H S_i U_i & U_i^H \widetilde{\boldsymbol{b}}_i \boldsymbol{e}_1^T \\ \boldsymbol{e}_1 \widetilde{\boldsymbol{b}}_i^H U_i & \widehat{T}_{i+1} \end{bmatrix} \right) \begin{bmatrix} U_i^H \boldsymbol{v}_i \\ 0 \end{bmatrix}. \tag{5.3.8}$$

Note that if we replace i with $i+1$ in (5.3.5) we obtain (5.3.8), i.e., we have written the remainder $\boldsymbol{r}_i$ in a form that is ready for the $(i+1)$-th cycle.

The approximate solution to $r(A)\boldsymbol{b}$ is updated with the identity

$$\boldsymbol{y}_i = \boldsymbol{y}_{i-1} + \boldsymbol{w}_{i-1} + V_i r(S_i) \boldsymbol{v}_i. \tag{5.3.9}$$

Recalling (5.3.2), in the first cycle we have $\widehat{\boldsymbol{y}} = \boldsymbol{y}_1 + \boldsymbol{w}_1 + \boldsymbol{r}_1$. It is easy to prove by induction that a similar identity holds in all subsequent cycles. Indeed, assuming that $\widehat{\boldsymbol{y}} = \boldsymbol{y}_{i-1} + \boldsymbol{w}_{i-1} + \boldsymbol{r}_{i-1}$, we have

$$\widehat{\boldsymbol{y}} = \boldsymbol{y}_{i-1} + \boldsymbol{w}_{i-1} + V_i r(S_i) \boldsymbol{v}_i + \boldsymbol{w}_i + \boldsymbol{r}_i = \boldsymbol{y}_i + \boldsymbol{w}_i + \boldsymbol{r}_i,$$

where we have used (5.3.7) and (5.3.9). Note that, similarly to the first cycle, in (5.3.8) the matrix V_i only appears multiplied by U_i, so after computing $\boldsymbol{y}_i$ and $\boldsymbol{w}_i$ it is no longer necessary to store the whole basis V_i.

5.3.3 The final cycle

The final cycle is slightly simpler, since we can compute the remainder directly instead of using the low-rank update formula. Indeed, at the beginning of the s-th cycle the remainder $\boldsymbol{r}_{s-1}$ can be computed directly from (5.3.8), which reads

$$\boldsymbol{r}_{s-1} = \begin{bmatrix} V_{s-1} U_{s-1} & Q_s \end{bmatrix} r\left(\begin{bmatrix} U_{s-1}^H S_{s-1} U_{s-1} & U_{s-1}^H \widetilde{\boldsymbol{b}}_{s-1} \boldsymbol{e}_1^T \\ \boldsymbol{e}_1 \widetilde{\boldsymbol{b}}_{s-1}^H U_{s-1} & T_s \end{bmatrix} \right) \begin{bmatrix} U_{s-1}^H \boldsymbol{v}_{s-1} \\ 0 \end{bmatrix}.$$

Using the same notation introduced in previous cycles, we define $V_s := \begin{bmatrix} V_{s-1} U_{s-1} & Q_s \end{bmatrix}$,

$$\boldsymbol{v}_s := \begin{bmatrix} U_{s-1}^H \boldsymbol{v}_{s-1} \\ 0 \end{bmatrix} \qquad \text{and} \qquad S_s := \begin{bmatrix} U_{s-1}^H S_{s-1} U_{s-1} & U_{s-1}^H \widetilde{\boldsymbol{b}}_{s-1} \boldsymbol{e}_1^T \\ \boldsymbol{e}_1 \widetilde{\boldsymbol{b}}_{s-1}^H U_{s-1} & T_s \end{bmatrix},$$

so we have

$$\boldsymbol{r}_{s-1} = V_s r(S_s) \boldsymbol{v}_s,$$

and the final approximation to $r(A)\boldsymbol{b}$ is obtained as

$$\boldsymbol{y}_s = \boldsymbol{y}_{s-1} + \boldsymbol{w}_{s-1} + V_s r(S_s) \boldsymbol{v}_s. \tag{5.3.10}$$

Note that (5.3.10) coincides with (5.3.9) where i has been replaced by s, so in the final cycle of the algorithm we compute the same update as in the other cycles, even though the derivation is different.

Since in the $(s-1)$-th cycle we have $\widehat{\boldsymbol{y}} = \boldsymbol{y}_{s-1} + \boldsymbol{w}_{s-1} + \boldsymbol{r}_{s-1}$, it follows that $\boldsymbol{y}_s = \widehat{\boldsymbol{y}}$, i.e., in the last cycle the algorithm that we just described computes the same approximation to $r(A)\boldsymbol{b}$ as $M = k + sm$ iterations of the outer Krylov subspace method. We will show in Proposition 5.4.1 that the same approximations as in the outer Krylov method are computed also in the intermediate cycles. The resulting algorithm is summarized in Algorithm 5.1.

5.4 Analysis and comparison with existing algorithms

In this section we analyze Algorithm 5.1 from both a theoretical and computational point of view, and we compare it with other low-memory methods from the literature.

Algorithm 5.1 RK-Compressed rational Lanczos for $f(A)\boldsymbol{b}$

Input: $A \in \mathbb{C}^{n\times n}, \boldsymbol{b} \in \mathbb{C}^n$, function f, $k, s, m \in \mathbb{N}$, complex conjugate inner poles $\boldsymbol{\xi}_k = \{\xi_0, \dots, \xi_{k-1}\}$, real outer poles $\boldsymbol{\theta}_M = \{\theta_0, \dots, \theta_{M-1}\}$ such that $M = k + sm$, $\theta_0 = \infty$ and $\theta_{k+im} = \infty$ for $i = 1, \dots, s-1$.

Output: $\boldsymbol{y}_s \approx f(A)\boldsymbol{b}$, such that $\boldsymbol{y}_s \in \mathcal{Q}_M(A, \boldsymbol{b}, \boldsymbol{\theta}_M)$.

▷ First outer cycle:

Run $m + k + 1$ iterations of the rational Lanczos algorithm (Algorithm 2.3) with the outer poles $\boldsymbol{\theta}_M$ to compute Q_1, T_1 and $\boldsymbol{b}_1$.

Compute the first approximation $\boldsymbol{y}_1 = Q_1 f(T_1)\boldsymbol{e}_1\|\boldsymbol{b}\|_2$.

Define $S_1 := T_1$, $V_1 := Q_1$, $\boldsymbol{v}_1 := \boldsymbol{e}_1\|\boldsymbol{b}\|_2$ and $\widetilde{\boldsymbol{b}}_1 := \boldsymbol{b}_1$.

▷ Other outer cycles:

for $i = 2, \dots, s$ **do**

  Construct the basis U_{i-1} of the rational Krylov subspace $\mathcal{Q}_k(S_{i-1}, \widetilde{\boldsymbol{b}}_{i-1}, \boldsymbol{\xi}_k)$.

  Compute the correction $\boldsymbol{w}_{i-1} = -V_{i-1}U_{i-1}f(U_{i-1}^H S_{i-1} U_{i-1})U_{i-1}^H \boldsymbol{v}_{i-1}$.

  Run m additional iterations of the rational Lanczos algorithm with the outer poles $\boldsymbol{\theta}_M$ to compute Q_i, T_i and $\widetilde{\boldsymbol{b}}_i$.

  Define $V_i := \begin{bmatrix} V_{i-1}U_{i-1} & Q_i \end{bmatrix}$, $\boldsymbol{v}_i = \begin{bmatrix} U_{i-1}^H \boldsymbol{v}_{i-1} \\ 0 \end{bmatrix}$, $S_i = \begin{bmatrix} U_{i-1}^H S_{i-1} U_{i-1} & U_{i-1}^H \widetilde{\boldsymbol{b}}_{i-1}\boldsymbol{e}_1^T \\ \boldsymbol{e}_1 \widetilde{\boldsymbol{b}}_{i-1}^H U_{i-1} & T_i \end{bmatrix}$.

  Update the approximation $\boldsymbol{y}_i = \boldsymbol{y}_{i-1} + \boldsymbol{w}_{i-1} + V_i f(S_i)\boldsymbol{v}_i$.

end for

5.4.1 Theoretical results

We start by showing that the iterates computed by Algorithm 5.1 coincide with iterates of the rational Lanczos method associated to the outer Krylov subspace $\mathcal{Q}_M(A, \boldsymbol{b}, \boldsymbol{\theta}_M)$ when f is a rational function.

Proposition 5.4.1. When f is a rational function of type $(k-1, k)$ with poles given by $\boldsymbol{\xi}_k$, the approximations $\{\boldsymbol{y}_i\}_{i=1}^s$ computed by Algorithm 5.1 coincide with the approximations given by (2.5.6) with an outer Krylov subspace of the appropriate dimension. Precisely, for any $i = 1, \dots, s$ we have

$$\boldsymbol{y}_i = \mathbf{Q}_{k+im} f(\mathbf{T}_{k+im})\boldsymbol{e}_1\|\boldsymbol{b}\|_2, \qquad \mathbf{T}_{k+im} = \mathbf{Q}_{k+im}^H A \mathbf{Q}_{k+im},$$

where $\mathbf{Q}_{k+im}$ is the orthonormal basis of $\mathcal{Q}_{k+im}(A, \boldsymbol{b}, \boldsymbol{\theta}_{k+im})$ generated by the rational Lanczos algorithm (Algorithm 2.3).

Proof. We have already observed that $\boldsymbol{y}_1$ coincides with the approximation (2.5.6) from the Krylov subspace $\mathcal{Q}_{k+m}(A, \boldsymbol{b}, \boldsymbol{\theta}_{k+m})$, and we have shown at the end of Section 5.3.3 that $\boldsymbol{y}_s$ coincides with the approximation $\widehat{\boldsymbol{y}}$ computed by the outer Krylov method after $M = k + sm$ iterations. To show that this also holds for all other cycles, let us fix $1 < i < s$ and suppose that we run a variant of Algorithm 5.1 with s replaced by $s' = i$ and M replaced by $M' = k + im$. It is easy to see that this variant of the algorithm performs exactly the same operations as the original one up to the $(i-1)$-th cycle, and hence computes the same approximate solutions $\boldsymbol{y}_1, \dots, \boldsymbol{y}_{i-1}$. The only difference from the original algorithm is that the i-th cycle is carried out by directly computing the residual $\boldsymbol{r}_{i-1}$ as described in Section 5.3.3, instead of performing the low-rank update with Corollary 5.2.6 as in Section 5.3.2. However, as shown in Section 5.3.3 for the final cycle of Algorithm 5.1, the update formula (5.3.10) for the final approximation to $r(A)\boldsymbol{b}$ coincides with (5.3.9), so this algorithm variant also computes the same approximation $\boldsymbol{y}_i$ as the original one. Since the final approximation computed by the variant of the algorithm coincides with the approximation generated by the outer Krylov subspace $\mathcal{Q}_{k+im}(A, \boldsymbol{b}, \boldsymbol{\theta}_{k+im})$ after $M' = k + im$ iterations, we conclude that this also holds for the approximation $\boldsymbol{y}_i$ computed in the i-th cycle of the original algorithm. □

When Algorithm 5.1 is applied to a general function f, we have to use Corollary 5.2.7 instead of Corollary 5.2.6, so in each cycle we introduce an error that depends on the approximation error of f with rational functions. The following proposition provides a bound on the error of Algorithm 5.1 with respect to the rational Lanczos algorithm in the general case.

Proposition 5.4.2. Assume that Algorithm 5.1 with s cycles is applied to a function f that is analytic on $\mathbb{W}(A)$, using inner poles $\boldsymbol{\xi}_k$. The error of Algorithm 5.1 with respect to the rational Lanczos algorithm is bounded by

$$\|\widehat{\boldsymbol{y}} - \boldsymbol{y}_s\|_2 \leq 4(s-1)\|\boldsymbol{b}\|_2 \min_{r \in \Pi_{k-1}/q} \|f - r\|_{\mathbb{W}(A)}, \tag{5.4.1}$$

where q is a polynomial with roots given by the finite elements of $\boldsymbol{\xi}_k$.

Proof. For convenience, we introduce the notation $\widetilde{S}_i = \begin{bmatrix} S_i & \widetilde{\boldsymbol{b}}_i \boldsymbol{e}_1^T \\ \boldsymbol{e}_1 \widetilde{\boldsymbol{b}}_i^H & \widehat{T}_{i+1} \end{bmatrix}$. For a general function f, when we use Corollary 5.2.7 instead of Corollary 5.2.6 in the i-th cycle of Algorithm 5.1, we introduce an additional error term $E_i(f)$ in the expression for $f(\widetilde{S}_i)$, which by (5.2.7) satisfies

$$\|E_i(f)\|_2 \leq 4 \min_{r \in \Pi_{k-1}/q} \|f - r\|_{\mathbb{W}(\widetilde{S}_i)}.$$

Note that for all i, we have $\mathbb{W}(\widetilde{S}_i) \subset \mathbb{W}(A)$. The expression for $\boldsymbol{r}_{i-1}$ with the error term included becomes

$$\boldsymbol{r}_{i-1} = V_i f(S_i)\boldsymbol{v}_i + \boldsymbol{w}_i + \boldsymbol{r}_i + \boldsymbol{\epsilon}_i,$$

where

$$\boldsymbol{\epsilon}_i = \begin{bmatrix} V_i & \widehat{Q}_{i+1} \end{bmatrix} E_i(f) \begin{bmatrix} \boldsymbol{v}_i \\ 0 \end{bmatrix}.$$

Since $\begin{bmatrix} V_i & \widehat{Q}_{i+1} \end{bmatrix}$ has orthonormal columns and $\|\boldsymbol{v}_i\|_2 \leq \|\boldsymbol{b}\|_2$, we have

$$\|\boldsymbol{\epsilon}_i\|_2 \leq \|E_i(f)\|_2 \|\boldsymbol{b}\|_2 \leq 4\|\boldsymbol{b}\|_2 \min_{r \in \Pi_{k-1}/q} \|f - r\|_{\mathbb{W}(A)}.$$

It is easy to see that, using the update formula (5.3.9) for $\boldsymbol{y}_i$ as in the rational function case, the approximate solution computed in the i-th cycle satisfies the identity

$$\widehat{\boldsymbol{y}} = \boldsymbol{y}_i + \boldsymbol{w}_i + \boldsymbol{r}_i + \sum_{\ell=1}^{i} \boldsymbol{\epsilon}_\ell.$$

Since in the final cycle the remainder $\boldsymbol{r}_{s-1}$ is computed directly, the total error of Algorithm 5.1 with respect to the rational Lanczos algorithm is thus

$$\|\widehat{\boldsymbol{y}} - \boldsymbol{y}_s\|_2 \leq \sum_{\ell=1}^{s-1} \boldsymbol{\epsilon}_i \leq 4(s-1)\|\boldsymbol{b}\|_2 \min_{r \in \Pi_{k-1}/q} \|f - r\|_{\mathbb{W}(A)}. \qquad \square$$

Proposition 5.4.2 shows that if we take $\boldsymbol{\xi}_k$ as the poles of a highly accurate rational approximant of f, the iterates $\boldsymbol{y}_i$ generated by Algorithm 5.1 essentially coincide with the corresponding approximations computed by the rational Lanczos algorithm, since the error in (5.4.1) is going to be negligible compared to the error $\|f(A)\boldsymbol{b} - \widehat{\boldsymbol{y}}\|_2$ of the outer Krylov subspace method.

5.4.2 Computational discussion

In its i-th cycle, Algorithm 5.1 has to keep in memory the last two columns of Q_{i-1} required to run the rational Lanczos algorithm for the computation of Q_i, the compressed basis $V_i \in \mathbb{C}^{n \times (k+m)}$ and the projected matrix $S_i \in \mathbb{C}^{(k+m)\times(k+m)}$, as well as the vectors $\boldsymbol{y}_{i-1}$, $\boldsymbol{w}_{i-1}$, $\boldsymbol{v}_i$ and $\widetilde{\boldsymbol{b}}_i$.

Since $\theta_{k+(i-1)m} = \infty$, the projected matrix T_i can be computed by exploiting the block structure of the matrices $\underline{\mathbf{H}}_{k+im}$ and $\underline{\mathbf{K}}_{k+im}$, as shown in Section 5.2.1. In particular, all the entries of T_i except for the one in position $(1,1)$ can be computed during the iterations of the rational Lanczos algorithm in which Q_i is constructed. The top-left entry of T_i needs to be corrected according to (5.2.3), which requires us to compute $\boldsymbol{e}_{k+(i-1)m}^T \mathbf{K}_{k+(i-1)m}^{-1} \boldsymbol{e}_{k+(i-1)m}$. Since we also have $\theta_{k+(i-2)m} = \infty$, the resulting block structure implies that we can compute $\mathbf{K}_{k+(i-1)m}^{-1} \boldsymbol{e}_{k+(i-1)m}$ by considering only the trailing

principal $m \times m$ block of $\mathbf{K}_{k+(i-1)m}$, which is computed in the $(i-1)$-th cycle. As a consequence, the correction for the $(1,1)$ entry of T_i in (5.2.3) can be precomputed during the $(i-1)$-th cycle, without the need to store the whole matrices $\underline{\mathbf{H}}_{k+im}$ and $\underline{\mathbf{K}}_{k+im}$. We omit the technical details.

It is important to note that Algorithm 5.1 only computes functions of matrices of size $k \times k$ and $(k+m) \times (k+m)$, that are independent of the number of cycles s, so the cost of computing functions of projected matrices does not increase with the iteration number, in contrast with the rational Lanczos method. This can lead to significant computational savings when the number of iterations is very large.

In each cycle, Algorithm 5.1 also has to construct a k-dimensional rational Krylov subspace associated with a $(k+m) \times (k+m)$ matrix. Since k and m are typically much smaller than n, the cost of operations with matrices and vectors of size $m+k$ is usually negligible with respect to the cost of operations with vectors of size n, so we expect that the construction of the inner rational Krylov subspace will not have a significant impact on the overall performance of the method.

In addition to the parameter k, which determines the accuracy of the inner rational approximation, and hence the highest accuracy attainable by the method, we also have to choose the parameter m, which determines the number of vectors that are added to the basis between two consecutive compressions. This parameter should be chosen according to the available memory. Taking m small decreases memory requirements but increases the frequency of computations with $(k+m) \times (k+m)$ matrices.

Note that in the pseudocode of Algorithm 5.1, our method only computes the approximate solutions $\boldsymbol{y}_i$ once every m iterations. However, the algorithm can be easily adapted to compute an approximate solution in any outer iteration that employs an infinite pole, using the same update formula as in line 9 of Algorithm 5.1 but with a smaller matrix S_i. Note that the only difference with respect to the definition of S_i in line 8 is in the T_i block and in the size of the off-diagonal blocks, so the only additional operation required to compute an approximate solution is the computation of a matrix function of size at most $(k+m) \times (k+m)$. It is easy to see that an approximate solution computed in this way also coincides with the one constructed in the corresponding iteration of the outer Krylov method, via the same argument used in the proof of Proposition 5.4.1. This technique is employed in our implementation of Algorithm 5.1.

Instead of running the method for a fixed number of cycles, in practice it is desirable to run it until we obtain a solution with a certain accuracy. A commonly used stopping criterion that can be employed for this purpose is the norm of the difference of two consecutive approximations. This simple criterion offers no guarantee on the final error and it may underestimate it in practice, especially when convergence is slow, but we found it to be accurate enough for our test problems. Observe that in order to check the convergence condition it is not necessary to form the approximate solutions $\boldsymbol{y}_i$ of length n, but it is enough to compute differences of short vectors. Indeed, we have

$$\begin{aligned}\boldsymbol{y}_i - \boldsymbol{y}_{i-1} &= \boldsymbol{w}_{i-1} + V_i f(S_i)\boldsymbol{v}_i \\ &= \begin{bmatrix} V_{i-1}U_{i-1} & Q_i \end{bmatrix} \left(\begin{bmatrix} f(U_{i-1}^H S_{i-1} U_{i-1}) U_{i-1}^H \boldsymbol{v}_{i-1} \\ 0 \end{bmatrix} + f(S_i)\boldsymbol{v}_i \right),\end{aligned}$$

and since $V_i = \begin{bmatrix} V_{i-1}U_{i-1} & Q_i \end{bmatrix}$ has orthonormal columns we conclude that

$$\|\boldsymbol{y}_i - \boldsymbol{y}_{i-1}\|_2 = \left\| \begin{bmatrix} f(U_{i-1}^H S_{i-1} U_{i-1}) U_{i-1}^H \boldsymbol{v}_{i-1} \\ 0 \end{bmatrix} + f(S_i)\boldsymbol{v}_i \right\|_2.$$

This formulation allows us to check if the stopping criterion is satisfied without performing operations with vectors of length n, which results in some computational savings. Our implementation of Algorithm 5.1 employs this expression in its stopping criterion.

5.4.3 Comparison with other low-memory Krylov methods

In this section we briefly compare our method with other low-memory Krylov subspace methods from the literature, highlighting advantages and disadvantages of each method. Similarly to multishift conjugate gradient (CG) [73], Algorithm 5.1 is based on a rational approximant, so the attainable accuracy is ultimately limited by the available memory, since the number of vectors that must be stored is proportional to the number of poles used in the rational approximation. However, while multishift CG

requires an explicit rational approximant in partial fraction form, our method only needs the poles of the rational function. This makes Algorithm 5.1 easier to use, and less susceptible to numerical errors caused by the representation of the rational function.

Given the similarities between Algorithm 5.1 and multishift CG, it is useful to compare the cost of the two methods in more detail. For this purpose, we employ the simple computational model used in [92, Experiment 5.3], in which operations on vectors of length n such as scaling or addition are counted as one unit of work, denoted by $1\mathcal{V}$, and operations with vectors or matrices of size independent of n are not counted. As discussed in [92], when the underlying rational approximant has k poles the multishift CG algorithm must store $2k$ vectors of length n and perform approximately $5k\mathcal{V}$ operations in addition to the standard Lanczos algorithm. Algorithm 5.1 with k inner poles and compression every m outer iterations must store $k + m$ vectors of length n, and the only operation that involves vectors of length n outside of the standard Lanczos algorithm is in the compression step (line 8 in Algorithm 5.1), where the product $V_{i-1}U_{i-1}$ has to be computed for a cost of $2k(k + m)\mathcal{V}$ every m iterations, that on average amounts to $2km^{-1}(k + m)\mathcal{V}$ at each iteration. If we take $m = k$, so that Algorithm 5.1 and multishift CG have the same memory requirements and the same attainable accuracy, the cost of Algorithm 5.1 is $4k\mathcal{V}$ per iteration, which is smaller than the $5k\mathcal{V}$ cost of multishift CG. However, note that the efficiency of multishift CG can be improved by removing converged linear systems, i.e., by no longer updating the approximate solutions of linear systems when the residual becomes smaller than the requested tolerance [150, Section 5.3]. This can significantly reduce the cost of the method, since linear systems associated to different poles often have substantially different convergence rates.

In contrast with Algorithm 5.1 and multishift CG, the memory requirements of the two-pass Lanczos method [31] are independent of the target accuracy, with the exception of the projected matrix, which grows in size at each iteration. When many iterations are needed to reach convergence, the computation of functions of projected matrices of increasing size can have a significant impact on the performance of the method, although this can be mitigated by only computing the approximate solution once every $d > 1$ iterations. For the same number of Lanczos iterations, the two-pass version requires twice the number of matrix-vector products with A compared to the other methods. However, in practice the overall cost is usually less than doubled, because in the second pass it is only necessary to recompute the Krylov basis vectors, and the orthogonalization coefficients have already been computed in the first pass.

The restarted Krylov method for Cauchy-Stieltjes matrix functions [71,72] requires storing a number of vectors proportional to the restart length. Although the amount of memory available does not influence the accuracy attainable by this method, a shorter restart length can cause delays in the convergence, similarly to what happens when restarting Krylov subspace methods for the solution of linear systems. The convergence delay can be mitigated by employing deflation techniques [62]. We note that this restarted method explicitly requires the expression of the integrand function in the Cauchy-Stieltjes representation of f. The restarted method can also be applied to a function that is not Cauchy-Stieltjes by using a different integral representation, such as one based on the Cauchy integral formula. This was done in [72, Section 4.3] for the exponential function. We also mention that in contrast with the other methods discussed here, the restarted method from [71, 72] can also be used for non-Hermitian matrices.

5.5 Numerical experiments

In this section we compare numerically Algorithm 5.1, which we denote by `RKcompress`, with other Krylov subspace methods for the computation of the action of a matrix function on a vector. For all methods, we monitor the relative norm of the difference between two consecutive computed solutions and stop when it becomes smaller than a requested tolerance. This difference is monitored at every iteration in which an infinite pole is used.

The MATLAB code for reproducing the experiments in this section is available in the Github repository `https://github.com/simunec/phd-thesis-code`. We use our own implementation of Algorithm 5.1, two-pass Lanczos and multishift CG, while for the restarted Arnoldi for Cauchy-Stieltjes matrix functions we use the `funm_quad` implementation [72, 138]. Although our implementation of Algorithm 5.1 has no external dependencies, to compute the rational approximant in partial fraction

Table 5.1 – Number of iterations and final relative error for all the methods in Figure 5.1, using relative tolerance 10^{-10}.

t	10^{-5}	10^{-4}	10^{-3}	10^{-2}	10^{-1}
iter	39	119	372	1104	1650
err	3.98e-11	1.89e-10	6.54e-10	2.26e-09	3.01e-09

form employed in the multishift CG algorithm we use the implementation of the AAA algorithm [122] from the chebfun package [58]. All the experiments were performed with MATLAB R2021b on a laptop running Ubuntu 20.04, with 32 GB of RAM and an Intel Core i5-10300H CPU with clock rate 2.5 GHz, using a single thread.

5.5.1 Exponential function

As a first test problem, we consider the computation of $e^{-tA}\mathbf{1}$, where the matrix $A \in \mathbb{R}^{n^2 \times n^2}$ is the discretization of the 2D Laplace operator with zero Dirichlet boundary conditions using centered finite differences with $n+2$ points in each direction, that is

$$A = B \otimes I + I \otimes B, \quad \text{where} \quad B = (n+1)^2 \begin{bmatrix} 2 & -1 & & \\ -1 & \ddots & \ddots & \\ & \ddots & \ddots & -1 \\ & & -1 & 2 \end{bmatrix} \in \mathbb{R}^{n \times n}, \tag{5.5.1}$$

and $\mathbf{1} \in \mathbb{C}^{n^2}$ is the vector of all ones. We compare the accuracy and timing of different methods based on Krylov subspaces using as a reference solution $e^{-tB}\mathbf{1} \otimes e^{-tB}\mathbf{1}$, where the involved exponentials are computed using the MATLAB command `expm`. We set $n = 10^3$ (therefore the size of A is $10^6 \times 10^6$) and choose different values of t. As t increases, the eigenvalues of the matrix $-tA$ of which we compute the exponential become more spread out, so we expect an increasing number of iterations for the convergence of polynomial Krylov subspace methods (see, e.g., [10, Section 4]).

We use Algorithm 5.1 with all outer poles equal to ∞ and $k = 25$ inner poles of the form described in [35], which guarantee a rational approximation of e^x for $x \in (-\infty, 0]$ with an absolute error of the order of machine precision. In particular, assuming that the involved matrix has no positive eigenvalues, the inner poles are independent of the spectrum of the matrix. We compare our `RKcompress` method with the standard Lanczos algorithm (`lanczos`), the two-pass version of Lanczos (`lanczos-2p`) and the Arnoldi algorithm with full orthogonalization (`arnoldi`). Note that both `lanczos` and `arnoldi` require storing the whole Krylov basis.

In Figure 5.1, we report the time needed to compute $e^{-tA}\mathbf{1}$ with the different methods for different values of t, using a relative tolerance of 10^{-10} in the stopping criterion. For any fixed value of t, all the employed methods converge in the same number of iterations, which is reported in Table 5.1 along with the final relative error attained. All the methods stop at the same iteration and produce the same approximate solution, confirming that Algorithm 5.1 is reproducing the convergence of the Lanczos algorithm (up to the error in the approximation of e^x with a rational function, which is negligible in this case). As the required number of iterations increases, our `RKcompress` method appears to be the fastest. This is mainly due to the fact that in Algorithm 5.1 the size of the matrix functions computed during the execution of the algorithm does not increase with the size of the Krylov subspace, in contrast with the other methods.

5.5.2 Markov functions

Recall that Markov functions can be represented as

$$f(z) = \int_\alpha^\beta \frac{1}{z-t}\, \mathrm{d}\mu(t),$$

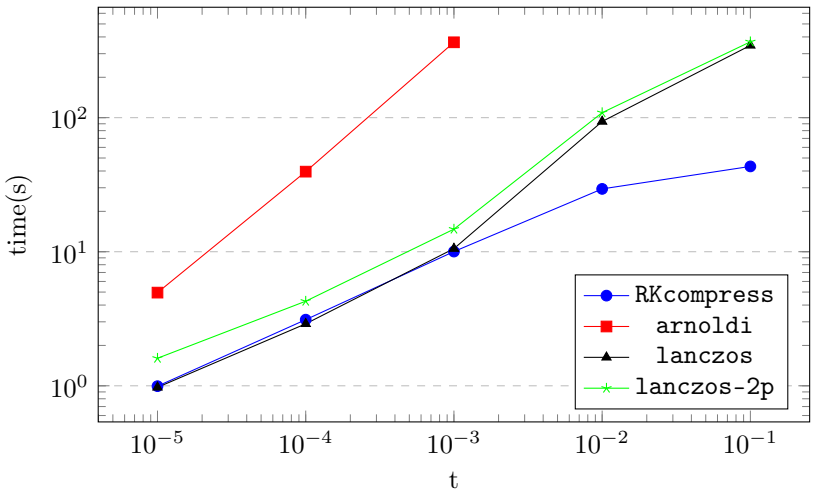


Figure 5.1 – Time needed for the computation of $e^{-tA}\mathbf{1}$ with accuracy of 10^{-10}, for different values of t and employing different methods.

where μ is a positive measure with support contained in $[\alpha, \beta]$ and $-\infty \leq \alpha < \beta < \infty$. If f is a Markov function and A is a Hermitian matrix with eigenvalues greater than β, we can use the approach from [10] mentioned in Section 2.5.3 to choose quasi-optimal inner poles for `RKcompress`. In particular, to obtain a rational approximation of f on an interval $[a, b]$ with $a > \beta$ with relative error norm bounded by ϵ it is sufficient to use k poles where

$$k \geq \log\left(\frac{4}{\epsilon}\right) \frac{\log(16(b-\beta)/(a-\beta))}{\pi^2}, \tag{5.5.2}$$

therefore the number of poles needed in `RKcompress` depends logarithmically on the condition number of $A - \beta I$. We refer to [10, Section 6.2] for more details regarding this choice of poles.

In the experiment that follows, we compute $A^{-1/2}\mathbf{1}$, where $A \in \mathbb{R}^{n^2 \times n^2}$ is a discretization of the 2D Laplace operator (5.5.1) with increasing $n = 200, 400, \ldots, 1000$, comparing several low-memory Krylov subspace methods, using a relative tolerance of 10^{-8}. The reference solution is computed by diagonalizing A, exploiting the Kronecker sum structure. We compare Algorithm 5.1 (`RKcompress`) with the two-pass Lanczos method (`lanczos-2p`), the standard multishift CG method (`msCG`), a more efficient implementation of multishift CG with removal of converged linear systems (`msCG-rem`), and the restarted Krylov method for Cauchy-Stieltjes functions, both with and without deflation, denoted by `restart-defl` and `restart`, respectively.

For Algorithm 5.1, we use outer poles all equal to ∞ and inner poles from [10] with k given by (5.5.2) and $m = k$. For a tolerance of 10^{-8}, the values of k range from $k = 26$ to $k = 32$ for the different matrix sizes. The rational approximant for multishift CG is obtained by running the AAA algorithm [122] with tolerance 10^{-12} on a discretization of the spectral interval of A; this produces a rational approximant with either 18 or 19 poles for all matrix sizes, that are fewer than the ones obtained using (5.5.2), but it does not provide any theoretical guarantee on the approximation error. For the restarted Krylov method for Cauchy-Stieltjes functions, we set a restart length equal to $2k$ (so that the memory requirement is the same as `RKcompress`) and we use the default options in `funm_quad`; we retain 5 Ritz vectors in the variant with deflation.

We report the times in Figure 5.2, the final errors in Table 5.2 and the number of iterations in Table 5.3. The methods with the shortest runtime are `RKcompress`, `msCG-rem` and `restart-defl`. Observe that `restarted-defl` requires significantly more iterations compared to the methods without restarting, so it would perform worse in a situation in which matrix-vector products with A are more expensive. Observe that for $n \geq 600$ the final error of multishift CG is significantly larger than the requested tolerance, seemingly suggesting that the rational approximant given by AAA is not accurate enough. However, it turns out that running `RKcompress` with inner poles given by the AAA approximant

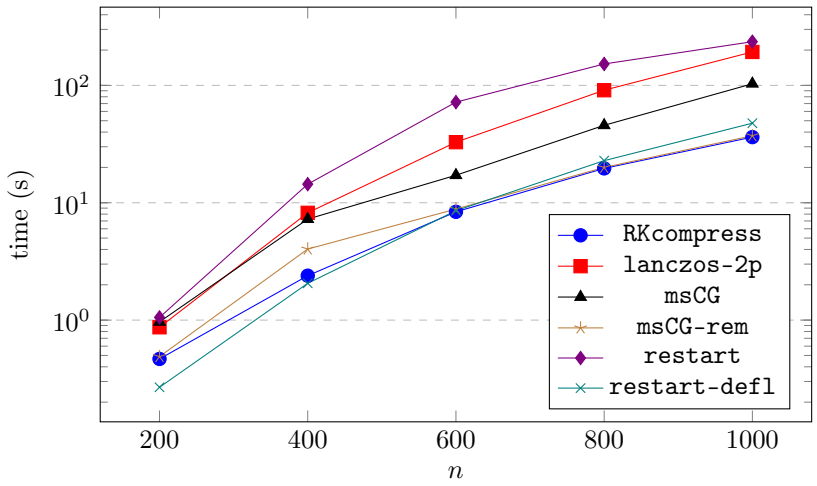


Figure 5.2 – Time needed for the computation of $A^{-1/2}\mathbf{1}$ with relative tolerance 10^{-8}, where $A \in \mathbb{C}^{n^2 \times n^2}$ is a discretization of the 2D Laplace operator for increasing n, employing different low-memory methods.

Table 5.2 – Final relative errors for the experiment in Figure 5.2.

size(A)	RKcompress	lanczos-2p	msCG	msCG-rem	restart	restart-defl
40000	9.01e-08	9.01e-08	7.81e-09	7.82e-09	2.93e-10	1.68e-11
160000	1.29e-07	1.29e-07	6.50e-09	6.50e-09	2.44e-08	1.58e-09
360000	1.70e-07	1.70e-07	2.68e-05	2.68e-05	5.29e-08	2.19e-09
640000	2.47e-07	2.47e-07	3.85e-06	3.85e-06	1.83e-05	4.34e-09
1000000	3.86e-07	3.86e-07	1.32e-06	1.32e-06	3.78e-04	2.52e-09

Table 5.3 – Number of iterations for the experiment in Figure 5.2.

size(A)	RKcompress	lanczos-2p	msCG	msCG-rem	restart	restart-defl
40000	282	282	379	379	2496	468
160000	554	554	747	747	8232	896
360000	823	823	1122	1122	16 380	1440
640000	1085	1085	1497	1497	18 600	2108
1000000	1336	1336	1872	1872	19 200	2880

Table 5.4 – Time, final error and number of iterations for the experiment in Section 5.5.3.

size(A)	time (s)		error		iter
	`RKcompress`	`rat-lanczos-2p`	`RKcompress`	`rat-lanczos-2p`	
50000	0.43	0.47	3.38e-08	7.22e-09	177
100000	1.03	1.31	1.67e-07	7.28e-08	239
150000	1.89	2.53	1.52e-07	2.82e-07	290
200000	3.49	3.97	2.55e-07	1.05e-07	331

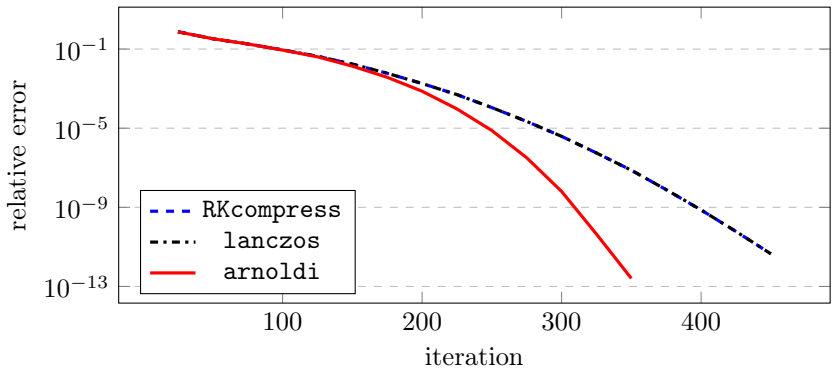


Figure 5.3 – Effect of numerical loss of orthogonality in the computation of $e^A\boldsymbol{b}$, where $A \in \mathbb{C}^{2000\times 2000}$ has logspaced eigenvalues in $[-10^4, -10^{-4}]$ and $\boldsymbol{b}$ is a random vector.

has roughly the same error as with the poles from [10], implying that the error in multishift CG should be mainly attributed to the explicit partial fraction representation of the rational function.

We note that the final error of `restart-defl` is smaller compared to the errors of `RKcompress` and `lanczos-2p`, since in the `funm_quad` implementation the approximate solutions are computed and compared only at the end of each restart cycle, hence the stopping criterion is more reliable and less likely to underestimate the error. The restarted Krylov method without deflation was unable to converge to the requested accuracy within 300 restart cycles for $n \geq 800$.

5.5.3 Rational outer Krylov method

In this experiment we examine the effectiveness of Algorithm 5.1 with a rational outer Krylov method. We let $A \in \mathbb{R}^{n\times n}$ be a discretization of the one-dimensional Laplace operator, and we approximate $A^{-1/2}\mathbf{1}$ with stopping tolerance 10^{-6}, using as reference solution the approximation computed with a rational Krylov method with tolerance 10^{-8}. We use a shift-and-invert outer Krylov method with the optimal single pole given in [10, Section 6.1]. For the inner poles in Algorithm 5.1, we use the quasi-optimal poles from [10] with k given by (5.5.2) and we set $m = 10$. We compare `RKcompress` with the two-pass rational Lanczos algorithm (`rat-lanczos-2p`) that uses the same poles (i.e., the shift-and-invert poles interleaved with an infinite pole at the iterations in which `RKcompress` performs a compression), so that the two methods have the same convergence rate, and they both check the convergence condition every 10 iterations. The times and final errors for increasing values of n are shown in Table 5.4. While both methods converge in the same number of iterations for all matrix sizes, `RKcompress` is faster than `rat-lanczos-2p` since it needs to solve half the number of shifted linear systems with the matrix A and it computes functions of smaller projected matrices.

5.5.4 Numerical loss of orthogonality

In order to investigate the behavior of Algorithm 5.1 in finite precision arithmetic, we compute $e^A\boldsymbol{b}$, where $A \in \mathbb{R}^{2000\times 2000}$ is a symmetric tridiagonal matrix with logspaced eigenvalues in the interval $[-10^4, -10^{-4}]$, and $\boldsymbol{b}$ is a random vector with unit normal entries. The convergence of `RKcompress`,

`lanczos` and `arnoldi` are compared in Figure 5.3. Both `lanczos` and `RKcompress` exhibit a delay in convergence due to the loss of orthogonality in the Krylov basis and they are essentially indistinguishable, so it appears that Algorithm 5.1 also reproduces the finite precision behavior of the Lanczos algorithm.

Chapter 6

Rational Krylov methods for fractional diffusion problems on graphs

In this chapter we consider the problem of solving a fractional diffusion equation on an undirected or directed graph [13, 131] by means of a rational Krylov method. We are going to see that this involves the computation of $f(L^T)\boldsymbol{u}_0$, where L is the Laplacian of the graph, which is a nonsymmetric matrix when the graph is directed, $\boldsymbol{u}_0$ is an initial condition and f is a function with a branch cut on $(-\infty, 0]$. Since L is a singular matrix, the function f is not analytic in a neighborhood of $\Lambda(L)$ and hence the convergence of rational Krylov methods is negatively affected. To address this issue, in Section 6.2 we introduce some techniques to remove the zero eigenvalue of L, obtaining a nonsingular Laplacian and therefore improving the convergence rate of rational Krylov methods. The applicability of this approach is not restricted to fractional diffusion problems, and it can be used when computing any function f of a matrix with a known eigenvalue close to a singularity of f. For instance, we are going to use the same approach for the approximation of the von Neumann entropy of a graph in Chapter 8. The contents of this chapter have been published in [23].

6.1 Background on graphs

A *directed graph*, or *digraph*, is a pair $G = (V, E)$, where $V = \{v_1, \dots, v_n\}$ is a set of n nodes or vertices, and $E \subseteq V \times V$ is the set of edges. A digraph can be represented with its *adjacency matrix* A, an $n \times n$ matrix whose entries are

$$A_{ij} = \begin{cases} 1 & \text{if } (i,j) \in E, \\ 0 & \text{otherwise.} \end{cases}$$

We assume that the graph has no self-loops, i.e., that $A_{ii} = 0$ for all i. The out-degree d_i of a node i is defined as the number of edges going out from i, i.e. edges of the form $(i, \ell) \in E$ for some node $\ell \in V$. The vector d of out-degrees can be computed as $\boldsymbol{d} = A\mathbf{1}$. If we denote by $D = \operatorname{diag}(\boldsymbol{d})$ the diagonal matrix of out-degrees, the (out-degree) *graph Laplacian* of G is defined as $L = D - A$.

Note that one can also define the vector of in-degrees as $\boldsymbol{d}_{\text{in}} = A^T\mathbf{1}$, and the in-degree graph Laplacian as $L_{\text{in}} = \operatorname{diag}(\boldsymbol{d}_{\text{in}}) - A$. Here, we focus solely on the out-degree graph Laplacian L, and we refer to it as the graph Laplacian whenever there is no ambiguity. Most of the properties of L also hold for L_{in}, and what we say for the out-degree graph Laplacian can be extended to the case of the in-degree Laplacian with only minor adjustments.

One can also consider weighted graphs, where each edge $(i, j) \in E$ has an associated positive weight w_{ij}, repesenting the strength of the connection between nodes i and j; if $(i, j) \notin E$, we write $w_{ij} = 0$. The matrix $W = (w_{ij})$ is a weighted adjacency matrix associated with G, and it can be used to defined a weighted vector of out-degrees $\boldsymbol{d}_W = W\mathbf{1}$ and a weighted graph Laplacian $L_W = \operatorname{diag}(\boldsymbol{d}_W) - W$. In the discussion that follows we only consider unweighted graphs for simplicity, but the techniques that we propose can be applied in the same way to weighted graphs. We also mainly focus on strongly connected graphs, i.e., graphs on which there exists a directed path from node i to

node j for any pair of nodes (i, j). Recall that a graph is strongly connected if and only if its adjacency matrix A (and hence also L) is irreducible, i.e., there exists no permutation matrix P such that $P^T AP$ is block triangular.

It follows immediately from its definition that the graph Laplacian L is a singular matrix, indeed we have $L\mathbf{1} = \mathbf{0}$.

Definition 6.1.1 (M-matrix [28])**.** A matrix $A \in \mathbb{R}^{n\times n}$ is an M-matrix if it can be written as $A = sI - B$ for some nonnegative matrix B, where $s \geq \rho(B)$, with $\rho(B)$ denoting the spectral radius of B. It is a singular M-matrix if $s = \rho(B)$.

One can easily prove the following basic result.

Proposition 6.1.2. The graph Laplacian L of a digraph G has the following properties:

1. L is a singular M-matrix.
2. The nonzero eigenvalues of L have positive real part.
3. 0 is a semisimple eigenvalue of L, i.e., its algebraic multiplicity and geometric multiplicity are the same.

These properties are fundamental for being able to define fractional powers of L, as we will see shortly.

6.1.1 Diffusion on a graph

Graph models are widely used to represent complex structures, ranging from physical and digital social networks, to transportation networks, to networks of chemical reactions and many others. It is often of interest in applications to study diffusion processes taking place on a network, that evolve in time according to the distribution of edges in the underlying graph. Such a process can be described in terms of a system of ordinary differential equations in time, with the Laplacian matrix L of the graph as the coefficient matrix.

Let us denote by $\boldsymbol{u}(t) \in \mathbb{R}^n$ a vector of concentrations at time t of a substance that is diffusing on the graph and let $\boldsymbol{u}_0$ be the vector of concentrations at time $t = 0$. Up to normalization, we can assume that $\boldsymbol{u}(t)$ is a probability vector, i.e. that $\boldsymbol{u}(t) \geq 0$ and $\boldsymbol{u}(t)^T\mathbf{1} = 1$. The diffusion equation on a directed graph reads

$$\begin{cases} \dfrac{\mathrm{d}}{\mathrm{d}t}\boldsymbol{u}(t)^T = -\boldsymbol{u}(t)^T L, & t \in (0, T], \\ \boldsymbol{u}(0) = \boldsymbol{u}_0 \geq 0, & \boldsymbol{u}_0^T\mathbf{1} = 1, \end{cases} \tag{6.1.1}$$

and the solution to this system of ordinary differential equations can be explicitly written in terms of the matrix exponential,

$$\boldsymbol{u}(t)^T = \boldsymbol{u}_0^T e^{-tL}.$$

Using properties of M-matrices, one can easily prove that e^{-tL} is a row-stochastic matrix, i.e., that it has nonnegative entries and $e^{-tL}\mathbf{1} = \mathbf{1}$, and hence the solution $\boldsymbol{u}(t)$ is a probability vector at all times $t \geq 0$. Note that this property would not be preserved if we used column vectors instead of row vectors in (6.1.1): see, e.g., the discussion in [39]. Given the explicit expression of $\boldsymbol{u}(t)$ in terms of the matrix exponential, it can be efficiently approximated for any time $t \geq 0$ using a polynomial Krylov method.

6.1.2 The fractional graph Laplacian

While a diffusion process is a local phenomenon, there are certain phenomena that allow long-range interactions and are non-local in nature: in the continuous setting, phenomena of this kind have been effectively modelled using fractional powers of the Laplace differential operator, that is $(-\Delta)^\alpha$ for $\alpha \in [1/2, 1)$ (see, e.g., [99, Definition 2]). Analogously, in the context of directed graphs, these phenomena can be well described with a fractional diffusion model, which employs a fractional power of the graph Laplacian instead of the Laplacian itself. Unlike for the continuous Laplace operator, in the discrete

case there is no need to restrict the values of α to the interval $[1/2, 1)$, and we can use any exponent $\alpha \in (0, 1)$. This modelling approach has been recently studied in [131] in the undirected case, and in [13] for more general directed graphs; we also refer to the book [118] for more information on fractional dynamics.

By Proposition 6.1.2, all eigenvalues of L are in the right half-plane and 0 is a semisimple eigenvalue, so the function $f(z) = z^\alpha$ for $\alpha \in (0, 1)$ is defined on the spectrum of L. Here we denote by z^α the branch of the fractional power with a cut on the negative real axis $(-\infty, 0]$, i.e., if $z = \rho e^{i\theta}$ with $\rho > 0$ and $\theta \in (-\pi, \pi)$, then $z^\alpha = \rho^\alpha e^{i\alpha\theta}$. With this definition, the fractional Laplacian L^α is still an M-matrix. Indeed, we have the following result.

Theorem 6.1.3 ([13, Theorem 2.7])**.** If A is a singular M-matrix with a semisimple 0 eigenvalue, then A^α is a singular M-matrix for every $\alpha \in (0, 1]$.

Moreover, since $L^\alpha \mathbf{1} = \mathbf{0}$, we can interpret the fractional graph Laplacian as the Laplacian of a weighted graph on the same set of nodes as the original graph G, and we can use it to define a fractional diffusion process on G with a system of differential equations analogous to (6.1.1):

$$\begin{cases} \dfrac{\mathrm{d}}{\mathrm{d}t}\boldsymbol{u}(t)^T = -\boldsymbol{u}(t)^T L^\alpha, & t \in (0, T], \\ \boldsymbol{u}(0) = \boldsymbol{u}_0 \geq 0, & \boldsymbol{u}_0^T \mathbf{1} = 1. \end{cases} \tag{6.1.2}$$

The solution of this system can be explicitly written in the form

$$\boldsymbol{u}(t)^T = \boldsymbol{u}_0^T e^{-tL^\alpha}, \qquad t \geq 0. \tag{6.1.3}$$

As in the case of classical diffusion, the solution $\boldsymbol{u}(t)$ to (6.1.2) is a probability vector at all times $t \geq 0$.

We mention that there are alternative approaches for modelling nonlocal phenomena on graphs, e.g., the one based on the transformed k-path Laplacian presented in [67, 68]: this model shares some properties with fractional diffusion, such as superdiffusive behavior on infinite one-dimensional graphs.

6.2 Dealing with the singularity

In this section we focus on the computational aspect of fractional dynamics, in particular on the efficient computation of the solution to the fractional diffusion equation on both undirected and directed graphs using rational Krylov subspace methods. As we mentioned, in the case of fractional diffusion on graphs the 0 eigenvalue of the Laplacian L is located on the bondary of the region of analiticity of the functions $f(z) = z^\alpha$ and $g(z) = \exp(-tz^\alpha)$. Moreover, for most directed graphs the numerical range of L intersects the negative real axis $(-\infty, 0)$, preventing us from using convergence results based on the approximation of the scalar functions f and g on $\mathbb{W}(L)$ (see, for instance, [13, Example 3.8]). The presence of an eigenvalue at 0 can also be detrimental in practice for the convergence speed of Krylov subspace methods.

With this motivation in mind, in this section we propose some techniques to remove the singularity of the graph Laplacian, which will allow us to work with nonsingular matrices and improve the convergence of Krylov subspace methods. Specifically, we propose a rank-one shift and a subspace projection that can be used to transform the graph Laplacian into a nonsingular matrix, and we provide simple formulas that link $f(L)$ and $f(L^T)\boldsymbol{b}$ with functions of the transformed matrix. We are also going to show that Krylov methods directly applied to the singular graph Laplacian can inherit the improved convergence of the projection approach, at least in exact arithmetic, provided that the initial vector $\boldsymbol{b}$ is suitably modified. We present these techniques for the case of strongly connected directed graphs. Recall that in this setting the eigenvalue 0 of the Laplacian L is simple. We mention that the idea of removing the singularity when working with a singular Laplacian has already appeared in previous literature. For instance, in [30] the authors consider finite element approximations of the pure Neumann problem, and they use projections to restrict the problem to the subspace of functions that are orthogonal to constants, where the continuous Laplace operator is invertible.

6.2.1 Rank-one shift

Recall that $L\mathbf{1} = \mathbf{0}$, and let $\boldsymbol{z} > \mathbf{0}$ be such that $\boldsymbol{z}^T L = \mathbf{0}^T$ and $\boldsymbol{z}^T\mathbf{1} = 1$. The vector $\boldsymbol{z}$ is uniquely determined by the above identities if the graph G is strongly connected (the positivity of $\boldsymbol{z}$ is a consequence of the Perron-Frobenius Theorem [116]).

The right and left eigenvectors $\mathbf{1}$ and $\boldsymbol{z}$ can be respectively completed to a right and left Jordan basis for L with two matrices R, $S \in \mathbb{C}^{n\times(n-1)}$, so that we have

$$Z^{-1}LZ = J = \begin{bmatrix} 0 & 0 \\ 0 & J_1 \end{bmatrix}, \qquad \text{where } Z = \begin{bmatrix} \mathbf{1} & R \end{bmatrix} \quad \text{and} \quad Z^{-1} = \begin{bmatrix} \boldsymbol{z}^T \\ S^T \end{bmatrix}.$$

The matrix $J_1 \in \mathbb{C}^{(n-1)\times(n-1)}$ contains all the other Jordan blocks of L, which correspond to nonzero eigenvalues.

Now, observe that $\mathbf{1}\boldsymbol{z}^T = (Z\boldsymbol{e}_1)(\boldsymbol{e}_1^T Z^{-1})$, and hence for all $\theta \in \mathbb{R}$ we have

$$L + \theta\mathbf{1}\boldsymbol{z}^T = Z \begin{bmatrix} \theta & 0 \\ 0 & J_1 \end{bmatrix} Z^{-1},$$

i.e., the matrix $L + \theta\mathbf{1}\boldsymbol{z}^T$ has the same spectrum as L, except for the eigenvalue 0 which is replaced by θ. Therefore, using basic properties of matrix functions, for any function f defined on the spectra of L and $L + \theta\mathbf{1}\boldsymbol{z}^T$ we have the identity

$$f(L + \theta\mathbf{1}\boldsymbol{z}^T) = Z \begin{bmatrix} f(\theta) & 0 \\ 0 & f(J_1) \end{bmatrix} Z^{-1} = f(L) + [f(\theta) - f(0)]\mathbf{1}\boldsymbol{z}^T. \tag{6.2.1}$$

This identity allows us to compute $f(L + \theta\mathbf{1}\boldsymbol{z}^T)$ instead of $f(L)$ and then recover the latter for a minimal cost. For any $\theta > 0$ (e.g., $\theta = 1$), the matrix $L + \theta\mathbf{1}\boldsymbol{z}^T$ is nonsingular and all its eigenvalues have strictly positive real part, so we expect Krylov subspace methods to converge faster for $L + \theta\mathbf{1}\boldsymbol{z}^T$ than for L when f has a branch cut on $(-\infty, 0]$, such as for $f(z) = z^\alpha$.

In particular, in the solution of the fractional diffusion equation (6.1.2) the objective is the computation of $f(L^T)\boldsymbol{u}_0$ for a probability vector $\boldsymbol{u}_0$ and $f(z) = \exp(-tz^\alpha)$, and from (6.2.1) we get

$$f(L^T)\boldsymbol{u}_0 = f(L^T + \theta\boldsymbol{z}\mathbf{1}^T)\boldsymbol{u}_0 + [f(0) - f(\theta)]\boldsymbol{z}. \tag{6.2.2}$$

When the matrix L is large and sparse and a rational Krylov method is used to approximate $f(L^T + \theta\boldsymbol{z}\mathbf{1}^T)$, it would be preferable to solve shifted linear systems involving the dense matrix $L^T + \theta\boldsymbol{z}\mathbf{1}^T$ without explicitly forming it, which can be done by using the Sherman-Morrison formula (2.1.1), or alternatively by using an iterative method. In our setting, for a pole $\xi \in (-\infty, 0)$ the invertibility condition $1 + \mathbf{1}^T(L^T - \xi I)^{-1}\boldsymbol{z} \neq 0$ is always satisfied (since $(L^T - \xi I)^{-1} \geq 0$, being the inverse of a nonsingular M-matrix). By applying the Sherman-Morrison formula (2.1.1) with $A = L^T - \xi I$ and observing that $(L^T - \xi I)^{-1}\boldsymbol{z} = -\xi^{-1}\boldsymbol{z}$ and $\mathbf{1}^T(L^T - \xi I)^{-1} = -\xi^{-1}\mathbf{1}^T$, we obtain

$$\begin{aligned} (L^T + \theta\boldsymbol{z}\mathbf{1}^T - \xi I)^{-1} &= (L^T - \xi I)^{-1} - \frac{\xi^{-2}\theta}{1 - \xi^{-1}\theta}\boldsymbol{z}\mathbf{1}^T \\ &= (L^T - \xi I)^{-1} + \frac{\theta}{\xi(\theta - \xi)}\boldsymbol{z}\mathbf{1}^T. \end{aligned} \tag{6.2.3}$$

Remark 6.2.1. Recalling the Jordan canonical form of L, we have

$$Z^{-1}\xi(L - \xi I)^{-1}Z = \begin{bmatrix} -1 & 0 \\ 0 & \xi(J_1 - \xi I)^{-1} \end{bmatrix},$$

where $(J_1 - \xi I)$ is invertible for all $\xi \leq 0$, since all the eigenvalues of J_1 have positive real part. By taking the limit for $\xi \to 0^-$ in the above expression we get

$$\lim_{\xi\to 0^-} \xi(L^T - \xi I)^{-1} = -\boldsymbol{z}\mathbf{1}^T, \tag{6.2.4}$$

and hence for small $\xi < 0$ and any vector $\boldsymbol{w}$ the identity (6.2.3) becomes

$$\begin{aligned}(L^T + \theta \boldsymbol{z}\mathbf{1}^T - \xi I)^{-1}\boldsymbol{w} &= (L^T - \xi I)^{-1}\boldsymbol{w} + \frac{\theta}{\xi(\theta-\xi)}(\mathbf{1}^T\boldsymbol{w})\boldsymbol{z} \\ &\approx -\xi^{-1}(\mathbf{1}^T\boldsymbol{w})\boldsymbol{z} + \frac{\theta}{\xi(\theta-\xi)}(\mathbf{1}^T\boldsymbol{w})\boldsymbol{z},\end{aligned} \tag{6.2.5}$$

i.e., we are very close to subtracting two multiples of the vector $\boldsymbol{z}$ of approximately the same length. Hence formula (6.2.5) may suffer from severe numerical cancellation when the pole ξ is very close to the origin, and therefore its use is not advised in that situation. Indeed, in our numerical experiments we observed a large loss of precision when solving linear systems with formula (6.2.5) for poles $\xi < 0$ of order 10^{-6}.

In order to address the issue mentioned in Remark 6.2.1, we now derive an alternative way to compute the solution of the shifted linear system $(L^T - \xi I)\boldsymbol{\phi} = \boldsymbol{w}$, in order to avoid the cancellation in (6.2.5) when ξ is small. We have

$$\mathbf{1}^T\boldsymbol{w} = \mathbf{1}^T(L^T - \xi I)\boldsymbol{\phi} = -\xi\,\mathbf{1}^T\boldsymbol{\phi} \implies \mathbf{1}^T\boldsymbol{\phi} = -\xi^{-1}\mathbf{1}^T\boldsymbol{w},$$

so we can define $\boldsymbol{\psi} := \boldsymbol{\phi} + \xi^{-1}(\mathbf{1}^T\boldsymbol{w})\boldsymbol{z}$, and it holds by construction that $\mathbf{1}^T\boldsymbol{\psi} = 0$. It is also straightforward to verify that $\boldsymbol{\psi}$ is the solution to the linear system

$$(L^T - \xi I)\boldsymbol{\psi} = \boldsymbol{w} - (\mathbf{1}^T\boldsymbol{w})\boldsymbol{z},$$

and that the vector $\boldsymbol{w} - (\mathbf{1}^T\boldsymbol{w})\boldsymbol{z}$ is orthogonal to $\mathbf{1}$. Hence, we can compute $\boldsymbol{\phi}$ as

$$\boldsymbol{\phi} = \boldsymbol{\psi} - \xi^{-1}(\mathbf{1}^T\boldsymbol{w})\boldsymbol{z}, \qquad \text{where} \quad (L^T - \xi I)\boldsymbol{\psi} = \boldsymbol{w} - (\mathbf{1}^T\boldsymbol{w})\boldsymbol{z}. \tag{6.2.6}$$

With formula (6.2.6), we have explicitly separated a component of the solution that is proportional to $\xi^{-1}\boldsymbol{z}$. By substituting (6.2.6) in (6.2.5), we obtain

$$\begin{aligned}(L^T + \theta \boldsymbol{z}\mathbf{1}^T - \xi I)^{-1}\boldsymbol{w} &= \boldsymbol{\phi} + \frac{\theta}{\xi(\theta-\xi)}(\mathbf{1}^T\boldsymbol{w})\boldsymbol{z} \\ &= \boldsymbol{\psi} - \xi^{-1}(\mathbf{1}^T\boldsymbol{w})\boldsymbol{z} + \frac{\theta}{\xi(\theta-\xi)}(\mathbf{1}^T\boldsymbol{w})\boldsymbol{z} \\ &= \boldsymbol{\psi} + \frac{\mathbf{1}^T\boldsymbol{w}}{\theta-\xi}\,\boldsymbol{z}.\end{aligned} \tag{6.2.7}$$

Observe that cancellation is avoided when using (6.2.7), because the subtraction of the two close multiples of the vector $\boldsymbol{z}$ is performed analytically. Moreover, because of (6.2.4) and since $(\boldsymbol{w} - (\mathbf{1}^T\boldsymbol{w})\boldsymbol{z}) \perp \mathbf{1}$, we have

$$\lim_{\xi \to 0^-} \xi(L^T - \xi I)^{-1}(\boldsymbol{w} - (\mathbf{1}^T\boldsymbol{w})\boldsymbol{z}) = \mathbf{0},$$

so for small $\xi < 0$ we do not expect $\boldsymbol{\psi}$ to have a component of order ξ^{-1} along the vector $\boldsymbol{z}$ (note that, in general, this argument fails for $\boldsymbol{\phi}$). Our numerical experiments confirm that the use of formula (6.2.7) fixes the problem of cancellation.

Remark 6.2.2. Note that if the graph is undirected, or more generally if it is balanced (i.e. each node has equal in- and out-degree), we also have $\boldsymbol{z} = \mathbf{1}$ up to a normalization factor, so no preliminary computation is needed to use this approach with the rank-one shift. On the other hand, for a general digraph it is first required to compute a nonzero vector $\boldsymbol{z}$ such that $L^T\boldsymbol{z} = \mathbf{0}$.

The problem of solving this linear system was recently discussed, for instance, in [16]. One possible approach is to compute an LDU factorization of the transpose of the graph Laplacian, $L^T = \mathcal{L}\mathcal{D}\mathcal{U}$, where $\mathcal{L}$ is unit lower triangular, $\mathcal{U}$ is unit upper triangular, and $\mathcal{D}$ is diagonal with $\mathcal{D}_{ii} > 0$ for $i = 1, \ldots, n-1$ and $\mathcal{D}_{nn} = 0$. Such a factorization always exists since L is an irreducible singular M-matrix [28], and it can be computed in a stable way using Gaussian elimination, with no pivoting required [81]. The original linear system $L^T\boldsymbol{z} = \mathbf{0}$ is thus equivalent to $\mathcal{D}\mathcal{U}\boldsymbol{z} = \mathbf{0}$, which can be solved by fixing $z_n = 1$ and

solving the lower triangular linear system $\mathcal{U}\boldsymbol{z} = \boldsymbol{e}_n$ via backward substitution. We remark that $\mathcal{L}^{-1} \geq 0$, so the vector $\boldsymbol{z}$ is nonnegative, and it can be indeed normalized so that $\boldsymbol{z}^T\mathbf{1} = 1$. We also mention that when L is sparse this method can be improved by computing the LDU factorization of $P^T L^T P$ instead of L^T, where P is a permutation matrix suitably chosen to reduce the fill-in in the factors $\mathcal{L}$ and $\mathcal{U}$. For example, the MATLAB routines `amd` and `symrcm` can be used for this purpose.

Alternatively, when L is very large and sparse, the linear system $L^T\boldsymbol{z} = \mathbf{0}$ can be solved iteratively, for instance with a preconditioned GMRES method (see [16] and references therein). Of course, if L is large and we choose to solve $L^T\boldsymbol{z} = \mathbf{0}$ iteratively, we should also use an iterative method for the solution of the shifted linear systems at each step of the rational Krylov iteration. However, here we do not address this specific subproblem, and we instead focus on the case where it is feasible to solve the linear systems with a direct method.

6.2.2 Projected Krylov methods

Another way to obtain a nonsingular matrix from the graph Laplacian is to project L on the $(n-1)$-dimensional subspace $\mathcal{S} = \operatorname{span}\{\mathbf{1}\}^\perp$. We remark that the approach presented here is similar to the one described in [33, Section 4], where the authors separate the eigenvalue 0 from the rest of the spectrum of a symmetric positive semidefinite matrix A, to compute $f(A)\boldsymbol{b}$ with an integral on a contour surrounding $\Lambda(A) \setminus \{0\}$. See also [100, 101] for a discussion of more general *spectral splitting* methods for symmetric matrices.

Let $\{\boldsymbol{q}_1, \dots, \boldsymbol{q}_{n-1}\}$ be an orthonormal basis of $\mathcal{S}$, and define $Q := \begin{bmatrix}\boldsymbol{q}_1 & \dots & \boldsymbol{q}_{n-1}\end{bmatrix}$. The matrix $\widetilde{Q} := \begin{bmatrix}Q & \frac{1}{\sqrt{n}}\mathbf{1}\end{bmatrix}$ is orthogonal, and we have $Q^TQ = I_{n-1}$ and $QQ^T = I_n - \frac{1}{n}\mathbf{1}\mathbf{1}^T$. Observe that the matrix Q^TLQ is nonsingular, since $\operatorname{range} Q = \operatorname{span}\{\mathbf{1}\}^\perp$, $\ker Q^T = \ker L = \operatorname{span}\{\mathbf{1}\}$ and $\operatorname{range} L = \operatorname{span}\{\boldsymbol{z}\}^\perp$.

We can rewrite $f(L)$ in terms of $f(Q^TLQ)$ by using some properties of matrix functions. Recalling that $L\mathbf{1} = \mathbf{0}$ and that $\mathbf{1}^TQ = \mathbf{0}^T$, we have:

$$\widetilde{Q}^T L \widetilde{Q} = \begin{bmatrix} Q^T \\ \frac{1}{\sqrt{n}}\mathbf{1}^T \end{bmatrix} L \begin{bmatrix} Q & \frac{1}{\sqrt{n}}\mathbf{1} \end{bmatrix} = \begin{bmatrix} Q^TLQ & \mathbf{0} \\ \frac{1}{\sqrt{n}}\mathbf{1}^TLQ & 0 \end{bmatrix}.$$

Now, using well known properties of matrix functions, we have

$$\widetilde{Q}^T f(L) \widetilde{Q} = f(\widetilde{Q}^T L \widetilde{Q}) = \begin{bmatrix} f(Q^TLQ) & \mathbf{0} \\ \boldsymbol{\varphi}^T & f(0) \end{bmatrix}, \tag{6.2.8}$$

for some $\boldsymbol{\varphi} \in \mathbb{R}^{n-1}$. The vector $\boldsymbol{\varphi}$ can be expressed in closed form (see, e.g., [96, Theorem 1.21]), but this is not necessary for our purposes.

Let us assume at first that our goal is to compute $f(L^T)\boldsymbol{v}$ for a vector $\boldsymbol{v}$ such that $\mathbf{1}^T\boldsymbol{v} = 0$. Using (6.2.8), we get

$$\begin{aligned} f(L^T)\boldsymbol{v} &= \widetilde{Q} f(\widetilde{Q}^T L \widetilde{Q})^T \widetilde{Q}^T \boldsymbol{v} \\ &= \begin{bmatrix} Q & \frac{1}{\sqrt{n}}\mathbf{1} \end{bmatrix} \begin{bmatrix} f(Q^TL^TQ) & \boldsymbol{\varphi} \\ \mathbf{0}^T & f(0) \end{bmatrix} \begin{bmatrix} Q^T\boldsymbol{v} \\ 0 \end{bmatrix} \\ &= Q f(Q^TL^TQ) Q^T \boldsymbol{v}. \end{aligned} \tag{6.2.9}$$

Now, consider the computation of $f(L^T)\boldsymbol{b}$ for a generic vector $\boldsymbol{b}$. If $\mathbf{1}^T\boldsymbol{b} = \beta \neq 0$, we can always write $\boldsymbol{b} = \boldsymbol{v} + \beta\boldsymbol{z}$ for some vector $\boldsymbol{v} \perp \mathbf{1}$ (recall that $\boldsymbol{z}$ satisfies $L^T\boldsymbol{z} = \mathbf{0}$ and $\mathbf{1}^T\boldsymbol{z} = 1$). Hence, using (6.2.9) we have

$$\begin{aligned} f(L^T)\boldsymbol{b} &= f(L^T)\boldsymbol{v} + \beta f(L^T)\boldsymbol{z} \\ &= Q f(Q^TL^TQ) Q^T\boldsymbol{v} + \beta f(0)\boldsymbol{z}. \end{aligned} \tag{6.2.10}$$

Using (6.2.10), we can compute $f(L^T)\boldsymbol{b}$ by using a rational Krylov method on the nonsingular projected matrix Q^TL^TQ. As in the case of the rank-one shift, this requires knowledge of $\boldsymbol{z}$, the left 0-eigenvector

of the graph Laplacian, which must be computed beforehand by solving the singular linear system $L^T\boldsymbol{z} = \mathbf{0}$ and suitably normalizing the solution.

In order to make the approach described here viable, we need to be able to compute matrix-vector products with Q efficiently: we next show that this can be done by constructing a suitable matrix Q that allows us to compute matrix-vector products $Q\boldsymbol{u}$ and $Q^T\boldsymbol{v}$ with $O(n)$ operations. Let us define the orthogonal matrix $\widetilde{Q} = \begin{bmatrix} Q & \frac{1}{\sqrt{n}}\mathbf{1} \end{bmatrix}$ as

$$\widetilde{Q} = \left[\begin{array}{cccc|c} 1/\sqrt{n} & \cdots & \cdots & 1/\sqrt{n} & 1/\sqrt{n} \\ \hline r & s & \dots & s & 1/\sqrt{n} \\ s & r & \ddots & \vdots & \vdots \\ \vdots & \ddots & \ddots & s & \vdots \\ s & \dots & s & r & 1/\sqrt{n} \end{array}\right], \qquad \text{where} \qquad \begin{aligned} s &= \frac{1}{1-n}\Big(1 + \frac{1}{\sqrt{n}}\Big), \\ r &= s + 1, \end{aligned}$$

so that we have

$$Q = \frac{1}{\sqrt{n}}\begin{bmatrix} 1 \\ \mathbf{0}_{n-1} \end{bmatrix}\mathbf{1}_{n-1}^T + s\begin{bmatrix} 0 \\ \mathbf{1}_{n-1} \end{bmatrix}\mathbf{1}_{n-1}^T + \begin{bmatrix} \mathbf{0}_{n-1}^T \\ I_{n-1} \end{bmatrix}. \tag{6.2.11}$$

Now, for any vectors $\boldsymbol{v} \in \mathbb{R}^n$ and $\boldsymbol{u} \in \mathbb{R}^{n-1}$ we have

$$\begin{aligned} Q\boldsymbol{u} &= \frac{1}{\sqrt{n}}\mathbf{1}_{n-1}^T\boldsymbol{u}\begin{bmatrix} 1 \\ 0_{n-1} \end{bmatrix} + s\mathbf{1}_{n-1}^T\boldsymbol{u}\begin{bmatrix} 0 \\ \mathbf{1}_{n-1} \end{bmatrix} + \begin{bmatrix} 0 \\ \boldsymbol{u} \end{bmatrix}, \\ Q^T\boldsymbol{v} &= \frac{1}{\sqrt{n}}v_1\mathbf{1}_{n-1} + s\sum_{j=2}^{n} v_j\mathbf{1}_{n-1} + \begin{bmatrix} v_2 \\ \vdots \\ v_n \end{bmatrix}, \\ (Q^TL^TQ - \xi I_{n-1})^{-1}\boldsymbol{u} &= Q^T(L^T - \xi I)^{-1}Q\boldsymbol{u}, \end{aligned} \tag{6.2.12}$$

where we denoted by v_j the entries of $\boldsymbol{v}$. The last equality in (6.2.12) follows from Lemma 6.2.4(b) in Section 6.2.3 below.

The matrix-vector products $Q\boldsymbol{u}$ and $Q^T\boldsymbol{v}$ can hence be computed with $O(n)$ arithmetic operations, and the solution of a shifted linear system with Q^TL^TQ can be reduced with $O(n)$ cost to the solution of a shifted linear system with L^T.

6.2.3 Implicit projection

In this section we show that in exact arithmetic the Krylov methods for $f(L^T)\boldsymbol{v}$ and $Qf(Q^TL^TQ)Q^T\boldsymbol{v}$ construct precisely the same approximations after an equal number of iterations, so it is actually not necessary to perform the projection explicitly. For this purpose, we examine how rational Krylov methods for the computation of $f(L^T)\boldsymbol{b}$ are related to their projected counterpart, i.e., to methods that first approximate $f(Q^TL^TQ)Q^T\boldsymbol{b}$ using rational Krylov subspaces and then use (6.2.9) to compute $f(L^T)\boldsymbol{b}$, in the case of an initial vector $\boldsymbol{b} \perp \mathbf{1}$. Note that the assumption $\boldsymbol{b} \perp \mathbf{1}$ is not satisfied when computing the solution to the fractional diffusion equation, since in that case the initial vector $\boldsymbol{u}_0$ is a probability vector, and hence $\mathbf{1}^T\boldsymbol{u}_0 = 1$. However, with the same procedure used in identity (6.2.10), the results of this section can be used with minor modifications for any initial vector $\boldsymbol{b}$.

We are going to prove our result in a slightly more general scenario: let $A \in \mathbb{R}^{n\times n}$, and let $\boldsymbol{x}$ be a left eigenvector of A such that $\boldsymbol{x}^TA = \lambda\boldsymbol{x}^T$. The specific case of the graph Laplacian will then correspond to $A = L^T$, $\boldsymbol{x} = \mathbf{1}$ and $\lambda = 0$. Let Q be an $n \times (n-1)$ matrix whose columns are an orthonormal basis of $\operatorname{span}\{\boldsymbol{x}\}^\perp$. If $\boldsymbol{b} \perp \boldsymbol{x}$, the same argument used in the proof of (6.2.9) gives us

$$f(A)\boldsymbol{b} = Qf(Q^TAQ)Q^T\boldsymbol{b}. \tag{6.2.13}$$

Denoting by V_k the orthonormal basis of $\mathcal{Q}_k(A, \boldsymbol{b}, \boldsymbol{\xi}_k)$ constructed with the rational Arnoldi algorithm, we have the approximation

$$f(A)\boldsymbol{b} \approx \boldsymbol{f}_k = V_kf(A_k)\boldsymbol{e}_1\|\boldsymbol{b}\|_2, \qquad A_k := V_k^TAV_k. \tag{6.2.14}$$

Alternatively, if we work with the right hand side of (6.2.13) and run the rational Arnoldi algorithm using the matrix Q^TAQ and the vector $Q^T\boldsymbol{b}$, after k iterations we have constructed the matrix $U_k := [\boldsymbol{u}_1 \dots \boldsymbol{u}_k] \in \mathbb{R}^{(n-1)\times k}$, whose columns are an orthonormal basis for $\mathcal{Q}_k(Q^TAQ, Q^T\boldsymbol{b}, \boldsymbol{\xi}_k)$. Then we have the approximation

$$Qf(Q^TAQ)Q^T\boldsymbol{b} \approx \widetilde{\boldsymbol{f}}_k = QU_kf(B_k)\boldsymbol{e}_1\|Q^T\boldsymbol{b}\|_2, \qquad B_k := U_k^T(Q^TAQ)U_k. \tag{6.2.15}$$

Due to (6.2.13), $\widetilde{\boldsymbol{f}}_k$ is also an approximation to $f(A)\boldsymbol{b}$. We will refer to the method described by equation (6.2.15) as a *projected rational Krylov method.*

The main result of this section is the following.

Theorem 6.2.3. Let $\boldsymbol{f}_k$ and $\widetilde{\boldsymbol{f}}_k$ be the approximations to $f(A)\boldsymbol{b}$ defined in (6.2.14) and (6.2.15), respectively, where $\boldsymbol{b} \perp \boldsymbol{x}$. Then, in exact arithmetic we have $\boldsymbol{f}_k = \widetilde{\boldsymbol{f}}_k$.

We start by establishing a few basic properties that will be useful in the proof of Theorem 6.2.3.

Lemma 6.2.4. Using the same notation as above, the following properties hold:

(a) $\mathcal{Q}_k(A, \boldsymbol{b}, \boldsymbol{\xi}_k) \perp \boldsymbol{x}$ and $QQ^T\boldsymbol{v} = \boldsymbol{v}$ for all $\boldsymbol{v} \in \mathcal{Q}_k(A, \boldsymbol{b}, \boldsymbol{\xi}_k)$.

(b) $(Q^TAQ - \xi I)^{-1} = Q^T(A - \xi I)^{-1}Q$ for any ξ such that the two inverses exist.

(c) $\mathcal{Q}_k(Q^TAQ, Q^T\boldsymbol{b}, \boldsymbol{\xi}_k) = Q^T\mathcal{Q}_k(A, \boldsymbol{b}, \boldsymbol{\xi}_k)$ and $Q\mathcal{Q}_k(Q^TAQ, Q^T\boldsymbol{b}, \boldsymbol{\xi}_k) = \mathcal{Q}_k(A, \boldsymbol{b}, \boldsymbol{\xi}_k)$.

(d) Let $\boldsymbol{w}_k \in \mathcal{Q}_k(A, \boldsymbol{b}, \boldsymbol{\xi}_k)$ and $\boldsymbol{z}_k = Q^T\boldsymbol{w}_k \in \mathcal{Q}_k(Q^TAQ, Q^T\boldsymbol{b}, \boldsymbol{\xi}_k)$. We have

$$(A - \xi_k I)^{-1}A\boldsymbol{w}_k \in \mathcal{Q}_k(A, \boldsymbol{b}, \boldsymbol{\xi}_k) \iff (Q^TAQ - \xi_k I)^{-1}Q^TAQ\boldsymbol{z}_k \in \mathcal{Q}_k(Q^TAQ, Q^T\boldsymbol{b}, \boldsymbol{\xi}_k).$$

This implies that $\boldsymbol{w}_k$ can be used to expand the subspace $\mathcal{Q}_k(A, \boldsymbol{b}, \boldsymbol{\xi}_k)$ with the next pole ξ_k if and only if $\boldsymbol{z}_k$ can be used to expand $\mathcal{Q}_k(Q^TAQ, Q^T\boldsymbol{b}, \boldsymbol{\xi}_k)$.

Proof. (a) Let $\boldsymbol{v} \in \mathcal{Q}_k(A, \boldsymbol{b}, \boldsymbol{\xi}_k)$, i.e., $\boldsymbol{v} = q_{k-1}(A)^{-1}p(A)\boldsymbol{b}$. Since $\boldsymbol{x}^TA = \lambda\boldsymbol{x}^T$, we also have $\boldsymbol{x}^T\boldsymbol{v} = \boldsymbol{x}^Tq_{k-1}(A)^{-1}p(A)\boldsymbol{b} = q_{k-1}(\lambda)^{-1}p(\lambda)\boldsymbol{x}^T\boldsymbol{b} = 0$, since $\boldsymbol{b} \perp \boldsymbol{x}$. For the second part of the statement, we have $QQ^T\boldsymbol{v} = \boldsymbol{v} - \boldsymbol{x}\boldsymbol{x}^T\boldsymbol{v} = \boldsymbol{v}$ since $\boldsymbol{v} \perp \boldsymbol{x}$.

(b) Let us show that $(Q^TAQ - \xi I)Q^T(A - \xi I)^{-1}Q = I$. We have:

$$\begin{aligned}(Q^TAQ - \xi I)Q^T(A - \xi I)^{-1}Q &= Q^T(A - \xi I)QQ^T(A - \xi I)^{-1}Q \\ &= Q^T(A - \xi I)(I - \boldsymbol{x}\boldsymbol{x}^T)(A - \xi I)^{-1}Q \\ &= I - Q^T(A - \xi I)\boldsymbol{x}(\lambda - \xi)^{-1}\boldsymbol{x}^TQ = I,\end{aligned}$$

where in the last two equalities we used $\boldsymbol{x}^T(A - \xi)^{-1} = (\lambda - \xi)^{-1}\boldsymbol{x}^T$ and $\boldsymbol{x}^TQ = \boldsymbol{0}^T$.

(c) Let $\boldsymbol{v} \in \mathcal{Q}_k(Q^TAQ, Q^T\boldsymbol{b}, \boldsymbol{\xi}_k)$, so $\boldsymbol{v} = q_{k-1}(Q^TAQ)^{-1}p(Q^TAQ)Q^T\boldsymbol{b}$, for some polynomial p with $\deg p \le k - 1$. Using that $QQ^T = I - \boldsymbol{x}\boldsymbol{x}^T$ and $\boldsymbol{b} \perp \boldsymbol{x}$, we can prove that $p(Q^TAQ)Q^T\boldsymbol{b} = Q^Tp(A)\boldsymbol{b}$. Because of (b), we also have

$$\begin{aligned}(Q^TAQ - \xi_{k-1}I)^{-1}Q^Tp(A)\boldsymbol{b} &= Q^T(A - \xi_{k-1}I)^{-1}(I - \boldsymbol{x}\boldsymbol{x}^T)p(A)\boldsymbol{b} \\ &= Q^T(A - \xi_{k-1}I)^{-1}p(A)\boldsymbol{b},\end{aligned}$$

since $x^Tp(A)\boldsymbol{b} = p(\lambda)\boldsymbol{x}^T\boldsymbol{b} = 0$. Using similar operations on the other factors of q_{k-1}, we finally obtain

$$\boldsymbol{v} = q_{k-1}(Q^TAQ)^{-1}p(Q^TAQ)Q^T\boldsymbol{b} = Q^Tq_{k-1}(A)^{-1}p(A)\boldsymbol{b} \in Q^T\mathcal{Q}_k(A, \boldsymbol{b}).$$

The second identity follows from the fact that $QQ^T\boldsymbol{v} = \boldsymbol{v}$ for all $\boldsymbol{v} \in \mathcal{Q}_k(A, \boldsymbol{b}, \boldsymbol{\xi}_k)$.

(d) Using (b) and $\mathcal{Q}_k(A, \boldsymbol{b}, \boldsymbol{\xi}_k) \perp \boldsymbol{x}$, we have the following:

$$(Q^TAQ - \xi_k I)^{-1}Q^TAQ\boldsymbol{z}_k = Q^T(A - \xi_k I)^{-1}QQ^TAQQ^T\boldsymbol{w}_k = Q^T(A - \xi_k I)^{-1}A\boldsymbol{w}_k.$$

Because of (c), it follows that $(A - \xi_k I)^{-1}A\boldsymbol{w}_k \in \mathcal{Q}_k(A, \boldsymbol{b}, \boldsymbol{\xi}_k)$ if and only if $Q^T(A - \xi_k I)^{-1}A\boldsymbol{w}_k \in Q^T\mathcal{Q}_k(A, \boldsymbol{b}, \boldsymbol{\xi}_k) = \mathcal{Q}_k(Q^TAQ, Q^T\boldsymbol{b}, \boldsymbol{\xi}_k)$. The second statement follows by recalling that a vector $\boldsymbol{v}$ can be used to expand the rational Krylov subspace in the rational Arnoldi algorithm if and only if it satisfies $(A - \xi_k I)^{-1}A\boldsymbol{v} \in \mathcal{Q}_{k+1}(A, \boldsymbol{b}, \boldsymbol{\xi}_{k+1}) \setminus \mathcal{Q}_k(A, \boldsymbol{b}, \boldsymbol{\xi}_k)$. □

Having established some basic facts about the projected Krylov subspace, we are now ready to prove Theorem 6.2.3.

Proof of Theorem 6.2.3. We start by showing that there exists a $k \times k$ diagonal and orthogonal matrix D_k such that $U_k = Q^T V_k D_k$. Since $\boldsymbol{u}_1 = Q^T \boldsymbol{v}_1$ by definition, by induction it is enough to prove that $\boldsymbol{u}_{k+1} = \pm Q^T \boldsymbol{v}_{k+1}$, with the assumption that $U_k = Q^T V_k D_k$, for $k \geq 1$. If we denote by $\boldsymbol{z}_k$ a vector such that $\boldsymbol{z}_k \in \mathcal{Q}_{k+1}(Q^T AQ, Q^T \boldsymbol{b}, \boldsymbol{\xi}_{k+1}) \setminus \mathcal{Q}_k(Q^T AQ, Q^T \boldsymbol{b}, \boldsymbol{\xi}_k)$, then by Lemma 6.2.4(d) we have that $\boldsymbol{w}_k = Q\boldsymbol{z}_k \in \mathcal{Q}_{k+1}(A, \boldsymbol{b}, \boldsymbol{\xi}_{k+1}) \setminus \mathcal{Q}_k(A, \boldsymbol{b}, \boldsymbol{\xi}_k)$. We have the following chain of equalities, using Lemma 6.2.4(b) for the second equality:

$$\begin{aligned}
\operatorname{span}\{U_k,\, \boldsymbol{u}_{k+1}\} &= \operatorname{span}\left\{U_k,\, (Q^T AQ - \xi_k I)^{-1} Q^T AQ \boldsymbol{z}_k\right\} \\
&= \operatorname{span}\left\{Q^T V_k D_k,\, Q^T (A - \xi_k I)^{-1} A \boldsymbol{w}_k\right\} \\
&= Q^T \operatorname{span}\left\{V_k,\, (A - \xi_k I)^{-1} A \boldsymbol{w}_k\right\} \\
&= Q^T \operatorname{span}\{V_k,\, \boldsymbol{v}_{k+1}\} = \operatorname{span}\left\{U_k,\, Q^T \boldsymbol{v}_{k+1}\right\}.
\end{aligned}$$

The vector $Q^T \boldsymbol{v}_{k+1}$ is orthogonal to the columns of $U_k = Q^T V_k D_k$, since

$$\boldsymbol{v}_{k+1}^T QQ^T V_k = \boldsymbol{v}_{k+1}^T (I - \boldsymbol{x}\boldsymbol{x}^T) V_k = \boldsymbol{0}^T.$$

So we have $\boldsymbol{u}_{k+1} = \pm Q^T \boldsymbol{v}_{k+1}$, since both vectors are in $\mathcal{Q}_{k+1}(Q^T AQ, Q^T \boldsymbol{b}, \boldsymbol{\xi}_{k+1})$ and they are orthogonal to the columns of U_k. Hence we have obtained $U_k = Q^T V_k D_k$ and $QU_k = QQ^T V_k D_k = V_k D_k$, where D_k is a diagonal matrix whose diagonal elements are equal to ± 1.

Now it is straightforward to see that $B_k = D_k A_k D_k$ and that $\boldsymbol{f}_k = \widetilde{\boldsymbol{f}}_k$. Indeed we have

$$\begin{aligned}
\widetilde{\boldsymbol{f}}_k &= QU_k f(B_k) U_k^T Q^T \boldsymbol{b} \\
&= V_k D_k f(U_k^T Q^T AQ U_k) D_k V_k^T \boldsymbol{b} \\
&= V_k D_k f(D_k V_k^T A V_k D_k) D_k V_k^T \boldsymbol{b} \\
&= V_k f(A_k) V_k^T \boldsymbol{b} = \boldsymbol{f}_k.
\end{aligned}$$

□

The result of Theorem 6.2.3 can also be seen as a consequence of the implicit Q theorem for rational Arnoldi decompositions [26, Theorem 3.2]. Although Theorem 6.2.3 is guaranteed to hold only in exact arithmetic, we observed from our experiments that the error curves given by the two approximations (6.2.14) and (6.2.15) are almost always overlapping, so in practice the implicit projection is a valid (and cheaper) alternative to (6.2.15).

Remark 6.2.5. The approach described in this section is similar to the deflation and augmentation strategies used in the solution of linear systems with Krylov methods: the aim of these techniques is to include exact or approximate spectral information on the matrix in order to speed up the convergence. This can be done by either adding a few known eigenvectors to the Krylov subspace, or by directly solving a deflated problem constructed using the spectral information on the matrix. Since the implicit projection method constructs a Krylov subspace that is orthogonal to the vector $\boldsymbol{x}$ (see Lemma 6.2.4(a)), it can be interpreted as an implicit way to construct an augmented Krylov subspace, also containing the eigenvector $\boldsymbol{x}$. For additional details on deflation and augmentation techniques used in Krylov methods for the solution of linear systems, we refer to the review article [142, Section 9] and to the references cited therein.

6.3 Numerical experiments

In this section we test and compare the performance of the various methods for the computation of $f(A)\boldsymbol{b}$ that we presented in Section 6.2, using them to approximate the solution to the fractional diffusion equation (6.1.2) on real-world networks, both undirected and directed. Recall that the solution to (6.1.2) at time t can be expressed in the form

$$\boldsymbol{u}(t) = f(L^T)\boldsymbol{u}_0, \qquad f(z) = \exp(-tz^\alpha), \quad t \geq 0, \quad \alpha \in (0, 1). \tag{6.3.1}$$

Since the graphs that we consider are strongly connected, the eigenvalues $\lambda_1, \dots, \lambda_n$ of the graph Laplacian can be ordered in a way such that $0 = \lambda_1 < |\lambda_2| \le \cdots \le |\lambda_n|$.

We use the result of Theorem 6.2.3 to compute $f(L^T)\boldsymbol{u}_0$ via (6.2.10) in the following way: letting $\beta = \mathbf{1}^T\boldsymbol{u}_0 > 0$, there exists $\boldsymbol{w} \perp \mathbf{1}$ such that $\boldsymbol{u}_0 = \boldsymbol{w} + \beta\boldsymbol{z}$ (recall that $L^T\boldsymbol{z} = \mathbf{0}$), and thus we can compute $f(L^T)\boldsymbol{u}_0$ as

$$f(L^T)\boldsymbol{u}_0 = f(L^T)\boldsymbol{w} + \beta f(L^T)\boldsymbol{z} = f(L^T)\boldsymbol{w} + \beta\boldsymbol{z}. \tag{6.3.2}$$

Theorem 6.2.3 guarantees that, at least in exact arithmetic, a rational Krylov method for $f(L^T)\boldsymbol{w}$ yields the same approximate solution as the same Krylov method applied to the projected problem $f(Q^TL^TQ)Q^T\boldsymbol{w}$. We refer to the method obtained by using (6.3.2) and approximating $f(L^T)\boldsymbol{w}$ with a Krylov method as an *implicitly projected Krylov method*.

For rational Krylov methods, we use poles located on the negative real axis $(-\infty, 0)$. For the shift-and-invert Krylov method, we compare two different choices of poles. Recall that in the case of a symmetric positive definite matrix A, if $a > 0$ is a lower bound for the smallest eigenvalue of A and $b > 0$ is an upper bound for its largest eigenvalue, so that $\Lambda(A) \subset [a, b]$, an effective pole choice for the shift-and-invert Krylov method is given by $\xi = -\sqrt{ab}$ (see, e.g., [10, Section 6]). In analogy with this choice, we use the pole $\xi = -\sqrt{|\lambda_2\lambda_n|}$: if the graph is undirected, when we use the rank-one shift approach (6.2.2) with $\theta \ge |\lambda_2|$, this pole corresponds exactly to the optimal choice $\xi = -\sqrt{\lambda_{\min}\lambda_{\max}}$ for symmetric positive definite matrices; the same choice appears to be reasonable also for the singular graph Laplacian and in the directed case. Indeed, our experiments show that this choice always provides a reliable convergence rate.

As a second possible choice for the shift-and-invert Krylov method, we use the pole $\xi = -t^{-2/\alpha}$ proposed in [120]; this choice is based on an integral bound for the error of the shift-and-invert Krylov method, obtained using an integral expression for the function f. The choice $\xi = -t^{-2/\alpha}$ was proposed for specific functions that arise in the context of fractional differential equations, like $f(z) = \exp(-tz^\alpha)$ or $f(z) = (1 + tz^\alpha)^{-1}$, with $\alpha \in (0, 1)$ and $t > 0$. This pole choice is particularly effective when t is large, but its performance is more sensitive to changes in the parameters t and α.

Recall that the function $f(z) = \exp(-tz^\alpha)$ is a Laplace-Stieltjes function (see Section 2.2), so for a general rational Krylov method we can use the poles for Laplace-Stieltjes functions from [114] constructed via an equidistributed sequence (EDS), described in Section 2.5.3. However, from [113, Example 2.4] we also have the integral expression

$$\exp(-tz^\alpha) = \frac{1}{\pi}\int_0^\infty \frac{1}{z + x}\exp(-tx^\alpha\cos\alpha\pi)\sin(tx^\alpha\sin\alpha\pi)\,\mathrm{d}x, \qquad t > 0, \quad \alpha \in (0, 1/2),$$

which shows that $f(z) = \exp(-tz^\alpha)$ is a Cauchy-Stieltjes function for $\alpha \in (0, 1/2)$. Comparing the equidistributed poles for Laplace-Stieltjes functions with the ones for Cauchy-Stieltjes functions from [10, Theorem 6.6], we observed that the convergence is moderately faster with the latter poles; the convergence rate always satisfies the bound given in [114, Corollary 4] when $\alpha < 1/2$, and it slowly deteriorates as α increases. In light of these observations, in the experiments that follow we use the equidistributed poles for Cauchy-Stieltjes functions for all parameter choices, since they provide a better practical performance compared to the Laplace-Stieltjes poles.

To compute the pole sequence, we use the slightly larger spectral interval $[0.99 \cdot |\lambda_2|,\ 1.01 \cdot |\lambda_n|] \supset \Lambda(A)$, once again ignoring the presence of the eigenvalue of the Laplacian at 0. The resulting poles are located on the negative real axis $(-\infty, 0)$. Similarly to the choice $\xi = -\sqrt{|\lambda_2\lambda_n|}$ for the shift-and-invert Krylov method, this pole sequence is guaranteed to have the asymptotic rate of convergence given in [114, Corollary 4] on the rank-one shifted matrix $L^T + \theta\boldsymbol{z}\mathbf{1}^T$, assuming that $\alpha \in (0, 1/2)$ and that the graph is undirected (and provided that $\theta \ge |\lambda_2|$). The experiments that we performed suggest that the same choice is still very effective also in the directed case and even when applied directly to the singular graph Laplacian. Note that this method, as well as the shift-and-invert method with $\xi = -\sqrt{|\lambda_2\lambda_n|}$, requires the knowledge of the largest and smallest nonzero eigenvalues of the graph Laplacian L, which have to be computed beforehand.

All the experiments were performed in MATLAB R2021b on a laptop running Ubuntu 20.04, with 32 GB of RAM and an Intel Core i5-10300H CPU with clock rate 2.5 GHz, using the `rat_krylov` function in the Rational Krylov toolbox [25] to construct the rational Krylov subspaces. The code for running the experiments in this section is available on Github in the repository `https://github.com/simunec/`

Table 6.1 – Information on the graphs used in the experiments. A denotes the $n \times n$ adjacency matrix of the LCC of each graph.

graph	n	nnz(A)	nnz(A-A')	λ_2	λ_n
minnesota	2640	6604	0	8.45e-04	6.88e+00
Oregon-1	11174	46818	0	8.44e-02	2.39e+03
ca-HepPh	11204	235238	0	3.55e-02	4.92e+02
as-22july06	22963	96872	0	5.07e-02	2.39e+03
Roget	904	4830	4128	8.22e-02	2.27e+01
wiki-Vote	1300	39456	67204	3.73e-01	5.96e+02
enron	8271	146260	212124	7.01e-02	8.79e+02
p2p-Gnutella30	8490	31706	63412	2.47e-01	2.00e+01
hvdc1	24836	133620	4856	2.03e-04	1.80e+02

phd-thesis-code. The shifted linear systems in the shift-and-invert Krylov method were solved by computing beforehand an LU decomposition of the permuted matrix $P^T L^T P - \xi I$, where P is a fill-in reducing permutation matrix obtained using the amd MATLAB function. In the rank-one shifted and in the projected version of the methods, we used the modified Sherman-Morrison formula (6.2.7), with the vector $\boldsymbol{\psi}$ defined in (6.2.6), and the identities (6.2.12) to avoid explicitly forming the dense matrices $L^T + \theta \boldsymbol{z}\mathbf{1}^T$ and $Q^T L^T Q$. The use of (6.2.7) over (6.2.3) allowed us to avoid the cancellation in (6.2.5) for poles close to zero, as discussed after Remark 6.2.1. We set $\theta = 1$ in (6.2.2) and we used the matrix Q defined by (6.2.11).

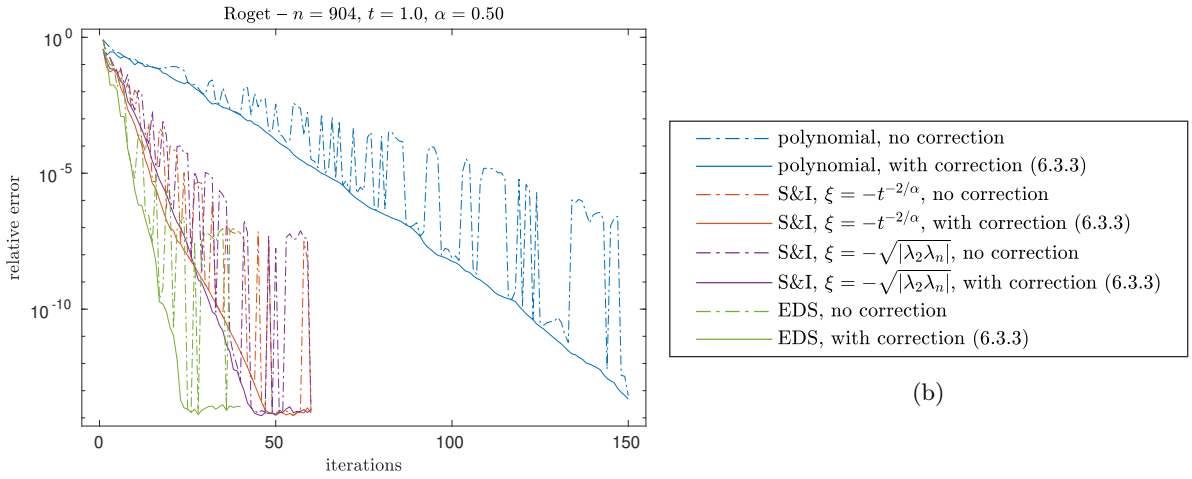
We display the relative error in the 2-norm $\|f(L^T)\boldsymbol{u}_0 - \boldsymbol{f}_k\|_2 / \|f(L^T)\boldsymbol{u}_0\|_2$, where $f(L^T)\boldsymbol{u}_0$ is obtained either by diagonalizing L or as an accurate approximation computed with a rational Krylov method when the size of the graph is large. In all our experiments, we first extracted the largest strongly connected component (LCC) of each graph and we restricted our problem to that component. Information on the number of nodes and edges of these components, as well as the maximum and minimum nonzero eigenvalues of the corresponding graph Laplacians, are reported in Table 6.1. The real-world networks that we used are available in the SuiteSparse Matrix Collection [51].

6.3.1 Error correction

As we observed in Section 6.1.2, the solution $\boldsymbol{u}(t)$ to the fractional diffusion equation (6.1.2) is a probability vector for all $t \geq 0$, and hence it is desirable for the approximations computed with Krylov subspace methods to have the same property. In our experiments, we observed that this is indeed the case for the Krylov methods applied to the shifted matrix $L^T + \boldsymbol{z}\mathbf{1}^T$, as well as for the projected and implicitly projected Krylov methods. On the other hand, in general the approximate solutions obtained by working directly with the singular graph Laplacian L^T do not have entries that sum up to 1, and often exhibit a wildly oscillating error; moreover, upon closer inspection, we observed that a significant portion of the error lied along the left null-eigenvector $\boldsymbol{z}$ of the graph Laplacian L. By subtracting a multiple of $\boldsymbol{z}$ from the approximate solution to enforce $\mathbf{1}^T \boldsymbol{f}_k = 1$, we were able to "correct" this oscillating component of the error; specifically, for each k we replaced the approximate solution $\boldsymbol{f}_k$ obtained after k iterations of a Krylov method with the corrected approximation

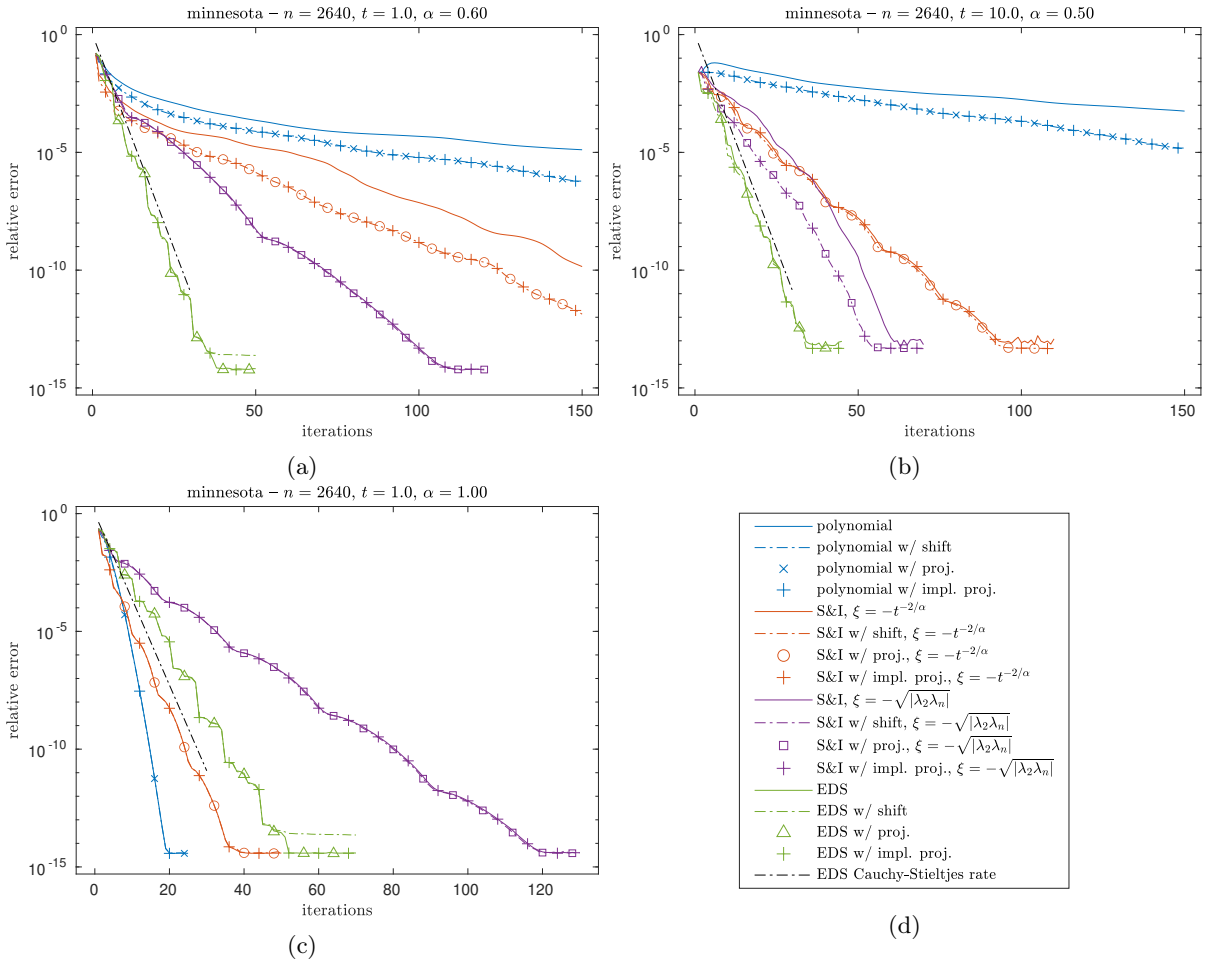
$$\boldsymbol{f}_k - (\mathbf{1}^T \boldsymbol{f}_k - 1)\boldsymbol{z}, \tag{6.3.3}$$

whose entries now sum up to 1. We found that this correction greatly reduced the error both in the undirected and in the directed case, and hence we always applied it to the standard version of Krylov methods in all our experiments. On the other hand, the error correction (6.3.3) was never needed for the shifted, projected and implicitly projected variants of the methods. The error with and without the correction (6.3.3) is illustrated for the directed graph Roget in Figure 6.1.



(a) (b)

Figure 6.1 – Convergence of Krylov methods without desingularization for the computation of the solution to the fractional diffusion equation (6.1.2) with $\alpha = 0.5$ and $t = 1$ on the LCC of the directed graph `Roget`, with and without the error correction (6.3.3).



(a) (b) (c) (d)

Figure 6.2 – Convergence of Krylov methods for the computation of the solution to the fractional diffusion equation (6.1.2) on the LCC of the undirected graph `minnesota`, for different values of α and t.

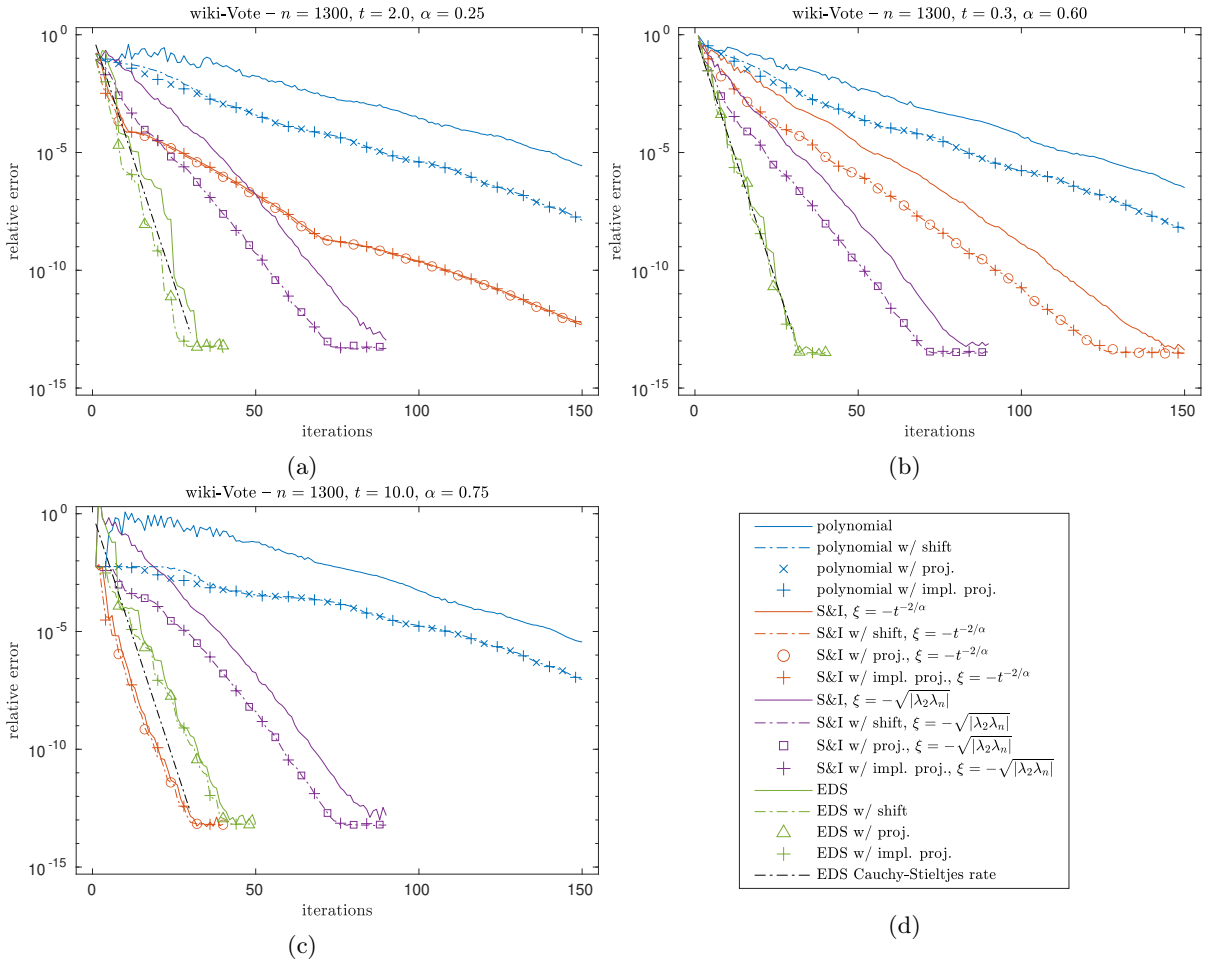


Figure 6.3 – Convergence of Krylov methods for the computation of the solution to the fractional diffusion equation (6.1.2) on the LCC of the directed graph `wiki-Vote`, for different values of α and t.

6.3.2 Small graphs

The first set of experiments is performed on graphs of moderate size (about 1000 nodes), where the solution to (6.3.1) can still be computed directly in a reasonable amount of time via an eigenvalue decomposition of the graph Laplacian. In these experiments, we used the LCC of the undirected graph `minnesota` (2640 nodes) and of the directed graph `wiki-Vote` (1300 nodes). The results for different values of t and α are shown in Figures 6.2 and 6.3. We can see that the general Krylov method with EDS poles for Cauchy-Stieltjes functions almost always converges in the smallest number of iterations, with a rate that is usually in agreement with the bound in [114, Corollary 4], even in the nonsymmetric case, except for Figure 6.2(c) and Figure 6.3(c), which use, respectively, $\alpha = 1$ and $\alpha = 0.75$. The shift-and-invert (S&I) Krylov method with $\xi = -\sqrt{|\lambda_2\lambda_n|}$ has a reliable convergence rate for all choices of the parameters, while the one with $\xi = -t^{-2/\alpha}$ is more effective for large t (see, e.g., Figure 6.3(c)); however, it is more sensitive to changes in the parameters, and sometimes converges slowly (Figure 6.2(a)). As expected, the polynomial Krylov method usually converges very slowly, except for the case $\alpha = 1$, corresponding to the matrix exponential (Figure 6.2(c)), for which polynomial Krylov methods are known to be effective.

Note that in Figure 6.2(b) we have $t^{-2/\alpha} = 10^{-4}$, and observe that the rank-one shifted shift-and-invert method with $\xi = -t^{-2/\alpha}$ attains the same accuracy as the other methods, as a result of formula (6.2.7); because of the cancellation discussed in Remark 6.2.1, the same accuracy could not be attained by using (6.2.3) in place of (6.2.7). On the other hand, for certain choices of the parameters

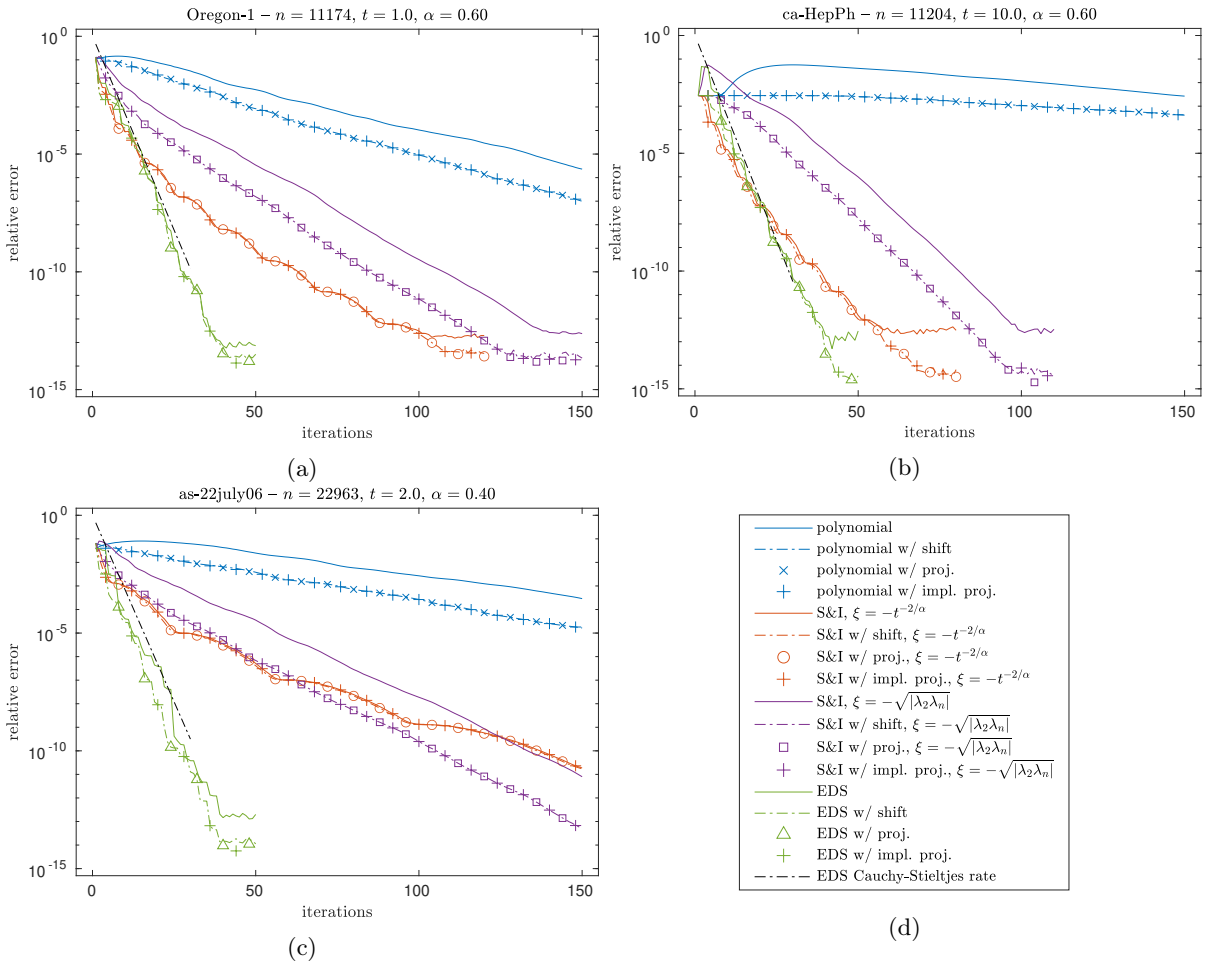


Figure 6.4 – Convergence of Krylov methods for the computation of the solution to the fractional diffusion equation (6.1.2) on the LCC of the undirected graphs `Oregon-1`, `ca-HepPh` and `as-22july06`, for different values of α and t.

in Figure 6.2, the rank-one shifted method with EDS poles attains a slightly lower final accuracy compared to the other desingularization techniques, suggesting that the numerical errors in the rank-one shift approach are still higher than in the other methods. However, this appears to be a minor issue, since we never observed the same phenomenon in the other test problems. In all other cases, the error curves of the desingularized methods are always overlapped to each other (showing, in particular, that the result of Theorem 6.2.3 also holds in finite precision arithmetic), and they often represent an improvement over the standard version of Krylov methods. Note that the desingularization techniques seem to be always effective for the polynomial Krylov method, reducing the error of at least one or two orders of magnitude compared to the standard version. We also point out that in certain cases the desingularized methods manage to attain a higher final accuracy: this is most apparent in Figure 6.3(c) for the shift-and-invert method with $\xi = -\sqrt{|\lambda_2\lambda_n|}$.

6.3.3 Large graphs

The second set of experiments deals with larger graphs with about 10000 or 20000 nodes, for which the computation of an eigenvalue decomposition of the graph Laplacian would be very expensive. Based on the results of the experiments on smaller graphs, in this case we compute the error using as the reference solution an approximation to $f(L^T)\boldsymbol{u}_0$ computed using the rational Krylov method with EDS poles with implicit projection, stopping the iterations when the 2-norm of the difference between two

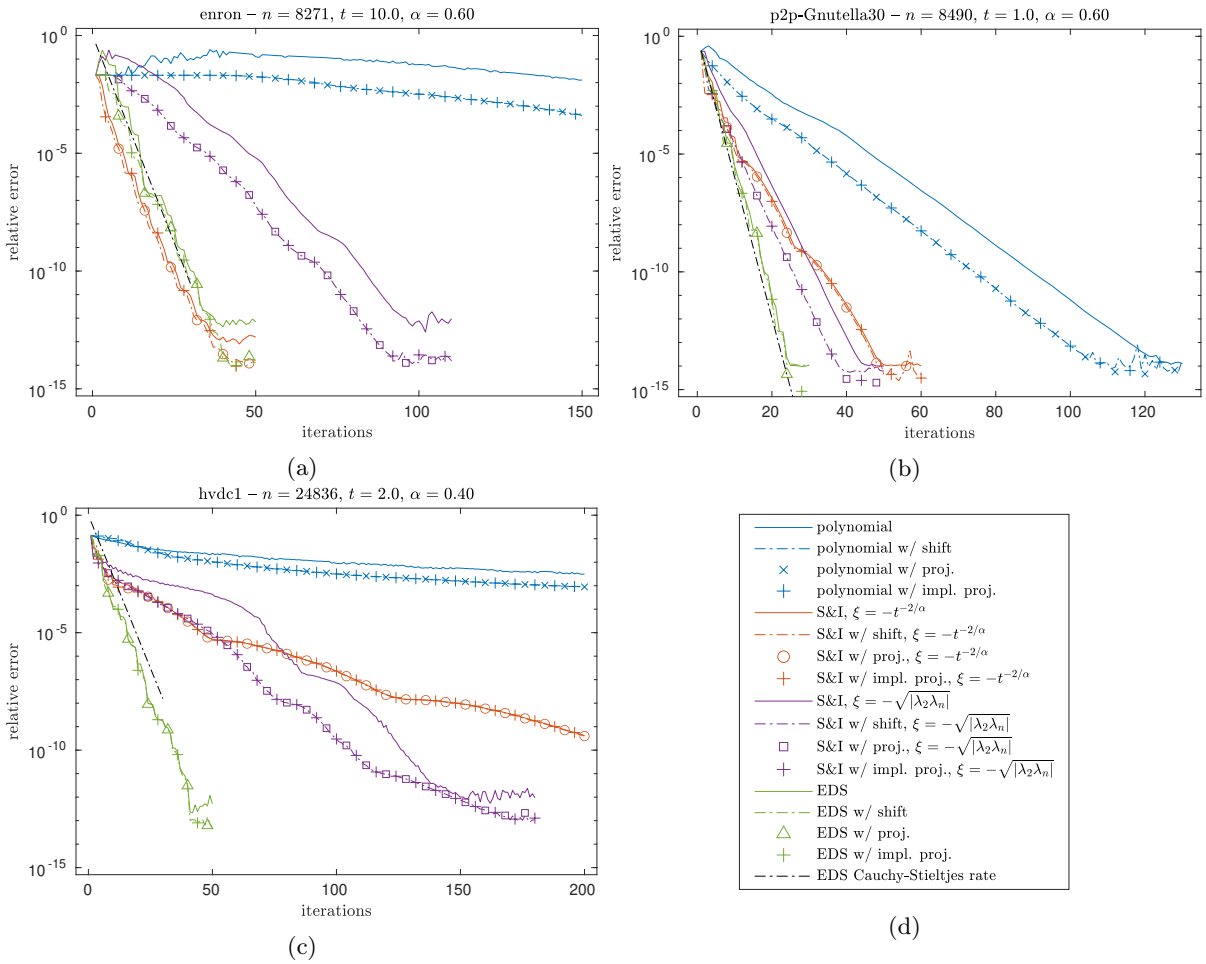


Figure 6.5 – Convergence of Krylov methods for the computation of the solution to the fractional diffusion equation (6.1.2) on the LCC of the directed graphs `enron`, `p2p-Gnutella` and `hvdc1`, for different values of α and t.

consecutive iterates is of the order of machine precision. To avoid bias in the error curves, we chose a different starting point for the EDS of the reference solution, thus producing a sequence of poles that is different from the one used to plot the error of the Krylov method with EDS poles. We used the LCC of the undirected graphs `Oregon-1` (11174 nodes), `ca-HepPh` (11204 nodes) and `as-july06` (22963 nodes), and of the directed graphs `enron` (8271 nodes), `p2p-Gnutella30` (8490 nodes) and `hvdc1` (24836 nodes). The results for different values of α and t are shown in Figures 6.4 and 6.5. The use of desingularization is again shown to be beneficial, often leading to faster convergence and attaining a better maximum accuracy. In Figure 6.4(b) and Figure 6.5(a) we can observe that the shift-and-invert method with $\xi = -t^{-2/\alpha}$ does not suffer from cancellation by virtue of (6.2.7), despite the presence of poles close to zero. These experiments also show that the polynomial Krylov method can have a variable and unpredictable convergence rate, depending on the graph: there are situations in which the convergence can take place quickly (Figure 6.5(b)) or with moderate speed (Figure 6.4(a)), but more often than not this method converges very slowly and the error practically stagnates (see, e.g., Figure 6.4(b)).

We have reported in Table 6.2 the execution times required by each method to reach a relative accuracy of 10^{-10}, using the same graphs and parameters α, t as in Figures 6.4 and 6.5. A "-" indicates that the polynomial Krylov method failed to attain the requested accuracy after 1000 iterations. For each graph, the first line in Table 6.2 lists the execution times for the standard Krylov methods, and the second line reports the average of the times for the 3 different methods with desingularization.

Table 6.2 – Execution times in seconds required to reach a relative accuracy of 10^{-10}, with reference to the plots in Figures 6.4 and 6.5. For each graph, the first line reports the time for the standard Krylov methods, and the second line reports the average time of the 3 desingularized methods. A "-" denotes failure to converge within 1000 iteration. The times are measured as an average across 10 runs, and the fastest method for each test problem is highlighted in bold.

graph	polynomial	S&I, $\xi = -t^{-2/\alpha}$	S&I, $\xi = -\sqrt{\lvert\lambda_2\lambda_n\rvert}$	EDS
`Oregon-1`	5.820	0.315	0.879	0.351
	4.082	**0.311**	0.558	0.347
`ca-HepPh`	72.092	1.600	2.167	1.470
	50.110	1.596	1.934	**1.456**
`as-22july06`	33.002	2.778	2.644	1.301
	22.172	2.756	1.668	**1.209**
`enron`	27.701	1.554	2.169	15.251
	19.934	**1.544**	1.982	14.775
`p2p-Gnutella30`	0.546	5.541	5.502	7.258
	0.351	5.482	5.415	7.265
`hvdc1`	-	8.252	2.442	1.546
	-	8.288	1.737	**1.448**

For a fixed choice of poles, the execution times of the desingularized methods are extremely similar, since these methods require the same number of iterations and roughly the same amount of work per iteration, with the explicit projection method being slightly more expensive. Observe that for the graph `p2p-Gnutella30` the polynomial Krylov method is faster than the rational methods in terms of execution time, even though it requires more iterations. However, in all the other experiments its convergence is too slow for it to be competitive.

In our implementation, the linear systems in each iteration of a rational Krylov method are solved with a direct method. This strategy suffers from severe fill-in when the network has a small-world structure: this is evident in the execution times of the EDS rational Krylov method for the directed graphs `enron` and `p2p-Gnutella30`, where a different shift is used at each iteration, and hence a new factorization has to be computed. On the other hand, this effect is less noticeable in the case of the shift-and-invert method, since the factorization is only computed once and reused in each iteration. As an example, if L is the Laplacian of the graph `enron` with nodes permuted with `amd`, the lower triangular factor in the LU factorization of $L + I$ has 1.26 million nonzero entries. In contrast, for the graph `hvdc1` the lower triangular factor of $L + I$ only has 120.714 nonzero elements. The problem of fill-in can be circumvented by using an iterative method for solving the linear systems in each iteration of a rational method: however, this strategy gives rise to a variety of questions, such as the choice of preconditioners and stopping criteria for the inner iteration, marking it as an interesting direction for future research.

Chapter 7

Overview of trace estimation

In this chapter we briefly review some methods for approximating the trace of a real symmetric matrix B that is not available explicitly and can only be accessed via matrix-vector products. In particular, we focus on stochastic trace estimators [98, 117, 129], which can be applied to any matrix B, and on probing methods [77, 145], which are specifically designed for approximating $\operatorname{tr}(f(A))$ when A is a sparse matrix. This chapter serves as an introduction to Chapters 8 and 9, in which we employ these methods in conjunction with Krylov subspace methods to approximate the von Neumann entropy and find gaps in the spectrum of a symmetric matrix.

7.1 Stochastic trace estimators

Let us consider the problem of computing $\operatorname{tr}(B)$, where $B \in \mathbb{R}^{n\times n}$ is a matrix that is not explicitly available but can be accessed via matrix-vector products $B\boldsymbol{x}$ and quadratic forms $\boldsymbol{x}^T B\boldsymbol{x}$, for $\boldsymbol{x} \in \mathbb{R}^n$. The case $B = f(A)$ is an instance of this problem, since the whole matrix $f(A)$ is expensive to compute but $f(A)\boldsymbol{x}$ and $\boldsymbol{x}^T f(A)\boldsymbol{x}$ can be efficiently approximated using Krylov subspace methods.

Stochastic trace estimators compute approximations of $\operatorname{tr}(B)$ by making use of the fact that, for any matrix B and any random vector $\boldsymbol{x}$ such that $\mathbb{E}[\boldsymbol{x}\boldsymbol{x}^T] = I$, we have $\mathbb{E}[\boldsymbol{x}^T B\boldsymbol{x}] = \operatorname{tr}(B)$, where $\mathbb{E}$ denotes the expected value. Hutchinson's trace estimator [98] is a simple stochastic estimator that generates s vectors $\boldsymbol{x}_1, \dots, \boldsymbol{x}_s$ with i.i.d. random entries, that are usually chosen as either Gaussian $\mathcal{N}(0,1)$ or Rademacher (random ± 1), and approximates $\operatorname{tr}(B)$ with

$$\operatorname{tr}_s^{\mathrm{H}}(B) := \frac{1}{s}\sum_{j=1}^{s} \boldsymbol{x}_j^T B\boldsymbol{x}_j. \tag{7.1.1}$$

In order to attain an absolute accuracy ε on $\operatorname{tr}(B)$, Hutchinson's trace estimator requires $O(\|B\|_F^2/\varepsilon^2)$ sample vectors [45,132]; the $O(1/\varepsilon^2)$ dependence on ε makes it very expensive to obtain highly accurate approximations, so this method is more suited for cheaply obtaining a rough estimate of $\operatorname{tr}(B)$.

The error from the Hutchinson's trace estimator when B is symmetric can be bounded by using the following result from [45].

Proposition 7.1.1 ([45, Theorem 1])**.** Let B be a nonzero symmetric matrix, and let $\operatorname{tr}_s^{\mathrm{H}}(B)$ denote the Hutchinson trace approximation obtained using s i.i.d. Gaussian $\mathcal{N}(0,1)$ random vectors. If

$$s \geq \frac{4}{\varepsilon^2}\big(\|B\|_F^2 + \varepsilon\|B\|_2\big)\log(2/\delta),$$

then

$$\mathbb{P}\big(|\operatorname{tr}(B) - \operatorname{tr}_s^H(B)| \geq \varepsilon\big) \leq \delta.$$

If we use vectors with Rademacher instead of Gaussian entries, the convergence rate of Hutchinson's estimator depends on $\|B - \operatorname{diag}(B)\|_F$ and $\|B - \operatorname{diag}(B)\|_2$ instead of $\|B\|_F$ and $\|B\|_2$, so in certain

Algorithm 7.1 Basic Hutch++ trace estimator

Input: $B \in \mathbb{R}^{n\times n}$ symmetric, $r, s \in \mathbb{N}$
Output: $\mathrm{tr}_{r,s}^{\mathrm{H}++}(B) \approx \mathrm{tr}(B)$
1: Sample $\boldsymbol{\Omega} \in \mathbb{R}^{n\times r}$ with random i.i.d. $\mathcal{N}(0,1)$ or ± 1 entries
2: Compute $B\boldsymbol{\Omega}$ and an orthonormal basis $\boldsymbol{Q} \in \mathbb{R}^{n\times r}$ of range$(B\boldsymbol{\Omega})$
3: Compute $\mathrm{tr}_1 = \mathrm{tr}(\boldsymbol{Q}^T B \boldsymbol{Q})$ ▷ exact trace of low-rank approx
4: Sample $\boldsymbol{X} \in \mathbb{R}^{n\times s}$ with random i.i.d. $\mathcal{N}(0,1)$ or ± 1 entries
5: Compute $\boldsymbol{Y} = (I - \boldsymbol{Q}\boldsymbol{Q}^T)\boldsymbol{X}$
6: Compute $\mathrm{tr}_2 = \frac{1}{s}\mathrm{tr}(\boldsymbol{Y}^T B \boldsymbol{Y})$ ▷ Hutchinson estimator on remainder
7: **return** $\mathrm{tr}_{r,s}^{\mathrm{H}++}(B) = \mathrm{tr}_1 + \mathrm{tr}_2$

situations it can be faster than with Gaussian vectors, for instance when B is diagonally dominant; see [45, Corollary 1] for a precise statement.

An algorithm that improves the convergence properties of Hutchinson's estimator has been proposed in [117] with the name of *Hutch++* (Algorithm 7.1). This estimator combines Hutchinson's estimator with low-rank approximation, in order to approximate $\mathrm{tr}(B)$ more efficiently when there is fast decay in the singular values of B. It samples $\Omega \in \mathbb{R}^{n\times r}$ with random i.i.d. $\mathcal{N}(0,1)$ entries, then computes $B\Omega$ and an orthonormal basis $Q \in \mathbb{R}^{n\times r}$ of range$(B\Omega)$. Then the trace of B is estimated by

$$\mathrm{tr}_{r,s}^{\mathrm{H}++}(B) := \mathrm{tr}(Q^T B Q) + \mathrm{tr}_s^{\mathrm{H}}((I - QQ^T)B(I - QQ^T)). \tag{7.1.2}$$

The Hutch++ estimator directly computes the trace of a rank r approximation of B, and then estimates the trace of the remaining part by using the Hutchinson estimator with s samples, so its total cost is r matrix-vector products and $r + s$ quadratic forms with B. It has been proven in [117, Theorem 4.1] that the complexity of Hutch++ is optimal up to logarithmic factors amongst algorithms that access a positive semidefinite matrix B via matrix-vector products.

In [129], an adaptive implementation of Hutch++ has been developed, which allows the user to prescribe a target absolute accuracy $\varepsilon > 0$ and a failure probability $\delta \in (0,1)$ as input parameters; the algorithm then selects the parameters r and s adaptively in order to satisfy the tail bound

$$\mathbb{P}\Big(|\mathrm{tr}_{r,s}^{\mathrm{H}++}(B) - \mathrm{tr}(B)| \geq \varepsilon\Big) \leq \delta \tag{7.1.3}$$

with as little computational effort as possible.

Several other papers on stochastic trace approximation have appeared in the recent literature. In particular, we mention the papers [64], which improves upon Hutch++ by exploiting the exchangeability principle for the sample vectors, and [42], in which stochastic trace estimators for the estimation of $\mathrm{tr}(f(A))$ are improved by taking into account how matrix-vector products with $f(A)$ are computed by means of Krylov subspace methods.

7.2 Probing methods

In this section we focus on the problem of approximating $\mathrm{tr}(f(A))$ for a sparse symmetric matrix A. In this setting, we can approximate $\mathrm{tr}(f(A))$ with the probing method described in [77, 145], which makes use of the sparsity of A by working with the graph $\mathcal{G}(A)$. Although we are only interested in the approximation of $\mathrm{tr}(f(A))$, the probing approach that we describe here can be used more generally to obtain sparse approximations to the whole matrix $f(A)$. We refer to [77, Section 3] for more details.

Recall that if A is an $n\times n$ symmetric matrix, $\mathcal{G}(A)$ is an undirected graph with nodes $\mathcal{V} = \{1, \ldots, n\}$ and edges $\mathcal{E} = \{(i,j) : A_{ij} \neq 0, i \neq j\}$, and that the *geodesic distance* $\mathrm{dist}(i,j)$ between two nodes i and j is the shortest length of a walk that starts with i and ends with j. A *coloring* of the graph $\mathcal{G}(A)$ is a partitioning $V_1, \ldots, V_s$ of the node set $\{1, \ldots, n\}$. Each set V_ℓ in the partition corresponds to a certain *color* ℓ, and we write $\mathrm{col}(i) := \ell$ for $i \in V_\ell$ to say that node i has color ℓ. A given coloring is a *distance-d coloring* if $\mathrm{col}(i) \neq \mathrm{col}(j)$ whenever $\mathrm{dist}(i,j) \leq d$. In other words, only nodes that are at a

Algorithm 7.2 Greedy algorithm for a distance-d coloring

Input: Graph $G = (V, E)$ with $V = \{1, \ldots, n\}$ and distance d
Output: Distance-d coloring col
1: $\text{col}(1) = 1$
2: **for** $i = 2 : n$ **do**
3: $\quad W_i = \{j \in \{1, \ldots, i-1\} : \text{dist}(i, j) \leq d\}$
4: $\quad \text{col}(i) = \min\{k > 0 : k \neq \text{col}(j)$ for all $j \in W_i$ for which $\text{col}(j)$ has been set$\}$
5: **end for**

distance strictly larger than d are allowed to have the same color. The *probing vectors* associated with the coloring are

$$\boldsymbol{v}_\ell = \sum_{i \in V_\ell} \boldsymbol{e}_i, \quad \ell = 1, \ldots, s, \tag{7.2.1}$$

i.e., $\boldsymbol{v}_\ell$ has ones in the components associated to nodes with color ℓ, and 0 in all other components. The induced *probing approximation* of the trace of $f(A)$ is

$$\text{tr}_d^{\text{P}}(f(A)) := \sum_{\ell=1}^{s} \boldsymbol{v}_\ell^T f(A) \boldsymbol{v}_\ell. \tag{7.2.2}$$

The problem of finding the minimum number s of colors required to have a distance-d coloring for a fixed d is NP-complete for general graphs [103], even for $d = 1$. From a computational point of view, it is important to keep the number of colors s as small as possible since, in view of (7.2.2), it coincides with the number of quadratic forms needed to approximate $\text{tr}_d^{\text{P}}(f(A))$. However, finding the exact minimum is not required, and we are satisfied with an algorithm that gives us a coloring that is relatively close to the optimal one. A greedy and efficient algorithm to get a quasi optimal distance-d coloring is given by Algorithm 7.2 ([136, Algorithm 4.2]), which tries to use as few colors as possible while examining nodes in a given order.

One can obtain different colorings depending on the order of the nodes used in Algorithm 7.2; sorting the nodes by descending degree usually leads to a good performance [103, 136]. The number of colors obtained with this algorithm is at most $\Delta^d + 1$, where Δ is the maximum degree of a node in $\mathcal{G}(A)$, and the cost is at most $\mathcal{O}(n\,\Delta^d)$ if the d-neighborhood W_i of each node i is computed with a traversal of the graph. Alternatively, the greedy coloring can be obtained by computing A^d, which can be done using at most $2\lfloor \log_2 d \rfloor$ matrix-matrix multiplications [96, Section 4.1]. Depending on the sparsity of A, either approach may be the more efficient one.

If A has bandwidth β, i.e., if $A_{ij} = 0$ for $|i - j| > \beta$, a simple distance-d coloring is given by

$$\text{col}(i) = \text{mod}(i - 1, d\beta + 1) + 1, \quad i = 1, \ldots, n. \tag{7.2.3}$$

This coloring is optimal with $s = d\beta + 1$ if all the entries within the bandwidth are nonzero. For a general sparse matrix A, as an alternative to Algorithm 7.2 one can use the reverse Cuthill-McKee algorithm [50] to reorder the nodes and reduce the bandwidth, and then use (7.2.3) to find a distance-d coloring [77]. Depending on the structure of the graph induced by A this can yield good results, but in many cases better colorings are obtained by using Algorithm 7.2: see for instance Example 8.2.1 in Chapter 8.

Note that approximating $\text{tr}(f(A))$ with a probing method when A is banded can be significantly more efficient than diagonalization. Indeed, to diagonalize a symmetric matrix with bandwidth β one must first reduce it to a tridiagonal form, and this costs $O(\beta n^2)$ [137]. On the other hand, if we assume that quadratic forms with $f(A)$ can be approximated with a (small) fixed number of Krylov iterations, the computation of each quadratic form costs $O(\beta n)$, and hence the cost for computing $\text{tr}_d^{\text{P}}(f(A))$ is approximatively $O(d\beta^2 n)$, which can be much less than the $O(\beta n^2)$ cost for diagonalization, as long as the requested accuracy is not too high.

An upper bound on the accuracy of the approximation of $\text{tr}(f(A))$ by $\text{tr}_d^{\text{P}}(f(A))$ is given by the following result.

Theorem 7.2.1 ([77, Theorem 4.4])**.** Let $A \in \mathbb{R}^{n\times n}$ be symmetric with $\Lambda(A) \subset [a,b]$. Let f be a function defined over $[a,b]$ and let $\mathrm{tr}_d^{\mathrm{P}}(f(A))$ be the approximation (7.2.2) of $\mathrm{tr}(f(A))$ induced by a distance-d coloring. Then

$$|\mathrm{tr}(f(A)) - \mathrm{tr}_d^{\mathrm{P}}(f(A))| \leq 2n \min_{p\in\Pi_d} \|f-p\|_{[a,b]}. \tag{7.2.4}$$

Theorem 7.2.1 shows that the error in the probing approximation of $\mathrm{tr}(f(A))$ is linked to the error in the approximation of f with polynomials. This property follows from the fact that the error in the trace can be written as

$$\mathrm{tr}(f(A)) - \mathrm{tr}_d^{\mathrm{P}}(f(A)) = \sum_{\ell=1}^{s} \sum_{\substack{i,j\in V_\ell \\ i\neq j}} [f(A)]_{ij},$$

see [77, eq. (4.1)]. Since the partitioning $V_1,\ldots,V_s$ is a distance-d coloring, for $i,j \in V_\ell$ such that $i \neq j$ we have $\mathrm{dist}(i,j) > d$. This implies that $[A^k]_{ij} = 0$ for $k = 0,\ldots,d$, i.e., any polynomial p_d with $\deg p_d \leq d$ is such that $[p_d(A)]_{ij} = 0$, and hence we have $\mathrm{tr}(p_d(A)) = \mathrm{tr}_d^{\mathrm{P}}(p_d(A))$. The bound in Theorem 7.2.1 then follows by adding and subtracting $\mathrm{tr}(p_d(A))$, similarly to the proof of Theorem 2.5.3.

Remark 7.2.2. We mention that a stochastic variant of the probing method has been analyzed in [74]. This variant replaces the nonzero entries of the probing vectors with either Gaussian or Rademacher random variables, and it has been used in past literature without a rigorous theoretical analysis, see for instance [82]. In [74, Section 3.4] the authors show that the accuracy of the Rademacher stochastic probing method depends sublinearly on the matrix dimension n, which is an improvement with respect to Theorem 7.2.1.

Chapter 8

Computation of the von Neumann entropy

In this chapter we consider the problem of approximating the von Neumann entropy [153] of a symmetric positive semidefinite matrix A, which is defined as

$$S(A) := \operatorname{tr}(-A \log A).$$

The von Neumann entropy plays an important role in several fields including quantum statistical mechanics [105], quantum information theory [11], and network science [32]. For example, computing the von Neumann entropy is necessary in order to determine the ground state of many-electron systems at finite temperature [1]. The von Neumann entropy of graphs is also an important tool in the structural characterization and comparison of complex networks [52, 93]. In recent years, a few papers have appeared devoted to this problem; see, e.g., [43, 56, 112, 156]. These papers investigate different approaches based on quadratic (Taylor) approximants, the global Lanczos algorithm, Gaussian quadrature and Chebyshev expansion. In general, the problem of computing the von Neumann entropy of a large matrix is difficult because the underlying matrix function is not analytic in a neighborhood of the spectrum when the matrix is singular; difficulties can also be expected when A has eigenvalues close to zero, which is usually the case. In particular, methods based on polynomial approximation may converge slowly in such cases.

We propose to approximate the von Neumann entropy using the trace estimation methods described in Chapter 7 and a Krylov subspace method that combines both polynomial and rational Krylov iterations. The goal is to obtain an algorithm that, given an input tolerance ε, computes an approximation of $S(A)$ with relative accuracy ε, using either a probing method or a stochastic trace estimator (in the latter case, the requested accuracy is attained with high probability). We begin by analyzing some properties of the von Neumann entropy and the function $f(z) = z \log(z)$ in Section 8.1, that allow us to obtain bounds for the error in the approximation of $S(A)$ with a probing method. We then give an overview of the algorithm in Section 8.2, and we discuss implementation details in Section 8.3. In particular, in Section 8.3.1 we propose a heuristic estimate for the error of the probing method, which can be used as a stopping criterion. In Section 8.4 we test the performance of the resulting algorithm by approximating the graph entropy of several complex networks. This chapter is mainly based on [21].

8.1 Properties of the von Neumann entropy

A *density matrix* is a self-adjoint, positive semidefinite linear operator with unit trace acting on a complex Hilbert space. In quantum mechanics, density matrices describe states of quantum mechanical systems and are usually denoted by ρ. To avoid confusion, here we prefer to denote a density matrix by A, since we often use lowercase greek letters to represent scalar quantities. Since we consider only the real finite dimensional case, in our setting a density matrix A is an $n \times n$ symmetric positive semidefinite matrix with $\operatorname{tr}(A) = 1$.

The *von Neumann entropy* [153] of a system described by the density matrix A is given by

$$S(A) := -\operatorname{tr}(A \log A) = -\sum_{\lambda \in \Lambda(A)} \lambda \log \lambda, \tag{8.1.1}$$

under the convention that $0 \cdot \log 0 = 0$. In the literature, the entropy is sometimes defined using the base-two logarithm $\log_2 x$ instead of the natural logarithm $\log x$, but since $\log_2 x = \log x / \log 2$ the two definitions are equivalent up to a scaling factor. We also mention that in the original definition the entropy includes the factor κ_B (the Boltzmann constant), which we omit.

Note that the formula (8.1.1) is well defined even without assuming that A has unit trace. Moreover, if A is given in the form $A = \gamma B$ for some $\gamma > 0$, we have the relation

$$S(A) = -\gamma \operatorname{tr}(B \log(\gamma B)) = \gamma S(B) - \gamma \log \gamma \operatorname{tr}(B), \tag{8.1.2}$$

so we can easily recover $S(A)$ from $S(B)$.

In quantum mechanics, the von Neumann entropy of a density matrix gives a measure of how much a density matrix is distant from being a *pure state*, that is a rank-1 matrix with only one nonzero eigenvalue equal to one [155]. For any density matrix of size n, we have $0 \leq S(A) \leq \log n$. Moreover, $S(A) = 0$ if and only if A is a pure state and $S(A) = \log n$ if and only if $A = \frac{1}{n} I$ [153, 155].

The von Neumann entropy also has applications in network theory [32, 43]. The von Neumann entropy of a graph G can be defined as $S(L/\operatorname{tr}(L))$, where L is the graph Laplacian of G. It should be mentioned that, strictly speaking, the graph entropy defined in this manner is not a "true" entropy, since it does not satisfy the sub-additivity requirement. In other words, the graph entropy could decrease when an edge is added to the graph, see [57]. For this reason, different notions of graph entropy have also been proposed in the literature; see, e.g., [84, 85]. These entropies still have the form of a von Neumann entropy, $S = -\operatorname{tr}(A \log A)$, but with a different definition of the density matrix A which, however, is still expressed as a function of a matrix associated to the graph. Therefore, the techniques that we develop here are still applicable, at least in principle.

8.1.1 Polynomial approximation

For a continuous function $f : [a, b] \to \mathbb{R}$, we will denote the error of the best uniform polynomial approximation of f in Π_k as

$$E_k(f, [a, b]) := \min_{p \in \Pi_k} \max_{x \in [a,b]} |f(x) - p(x)|.$$

We have already seen that when A is a symmetric matrix with $\Lambda(A) \subset [a, b]$, estimating or bounding $E_k(f, [a, b])$ is crucial in the study of polynomial approximations of the matrix function $f(A)$. The error of the best polynomial approximation also plays a key role in establishing decay bounds for the entries of $f(A)$ [17, 54, 75] and in bounding the error of a probing method for approximating $\operatorname{tr}(f(A))$ (Theorem 7.2.1).

In particular, in the case of the inverse function $f(x) = 1/x$ for $x \in [a, b]$, $0 < a < b$, we have

$$E_k(1/x, [a, b]) = \frac{(\sqrt{\kappa} + 1)^2}{2b} \left(\frac{\sqrt{\kappa} - 1}{\sqrt{\kappa} + 1} \right)^{k+1}, \tag{8.1.3}$$

where $\kappa = b/a$; see, e.g., [115, Section 4.3] and [54, Proposition 2.1].

In general, an inequality of the form

$$E_k(f, [a, b]) \leq C q^k, \qquad C > 0, \qquad 0 < q < 1, \tag{8.1.4}$$

is equivalent to the fact that f is analytic over an ellipse containing $[a, b]$; see [115, Theorem 73] for more details. However, the function $f(x) = x \log x$ is not analytic on any neighborhood of 0, so we cannot expect to find a geometric rate as in (8.1.4) if we consider $[0, b]$, $b > 0$, as the interval of definition of f. However, since f is continuous, by the Weierstrass Approximation Theorem $E_k(f, [0, b])$ must converge

to 0 as $k \to \infty$. An explicit bound is the following [156], derived by computing the coefficients of the Chebyshev expansion of $f(x) = x \log x$,

$$E_k(x \log x, [0, b]) \leq \frac{b}{2k(k+1)}, \qquad \text{for all } k \geq 1. \tag{8.1.5}$$

This shows that the decay rate of the error is algebraic in k.

In this section we derive a sharper bound on $E_k(f, [0, b])$ by exploiting an integral representation of $f(x) = x \log x$. Recall that the function $\log(1+z)/z$ is a Cauchy-Stieltjes function, with the integral expression

$$\frac{\log(1+z)}{z} = \int_1^\infty \frac{1}{s(s+z)}\, \mathrm{d}s, \qquad z > 0.$$

It is easy to check that the above identity also holds for $z \in (-1, 0)$. With the change of variable $x = 1 + z$ and some simple rearrangements, we get the following integral expression for the entropy function,

$$-x \log(x) = \int_1^\infty \frac{x(1-x)}{s(s+x-1)}\, \mathrm{d}s, \qquad x \in [0, 1]. \tag{8.1.6}$$

With the additional change of variable $s = t + 1$, we can rewrite (8.1.6) in the form

$$-x \log(x) = x(1-x) \int_0^\infty \frac{1}{(t+x)(t+1)}\, \mathrm{d}t, \qquad x \in [0, 1]. \tag{8.1.7}$$

Note that the two identities above hold also for $x = 0$ and $x = 1$, because of the factor $x(1-x)$ in front of the integral. The expression (8.1.7) shows that although the entropy function $-x \log(x)$ is not itself a Cauchy-Stieltjes function, it is the product of a Cauchy-Stieltjes function and a polynomial of degree two.

Integral representations have proved to be a useful tool for many problems related to polynomial approximations and matrix functions; see, e.g., [19, 22, 75]. The following result shows that we can derive explicit bounds for $E_k(f, [a, b])$ from an integral representation of f.

Lemma 8.1.1. Let $f(x)$ be defined for $x \in [a, b]$ by

$$f(x) = \int_0^\infty g_t(x)\ \mathrm{d}t, \tag{8.1.8}$$

where $g_t(x)$ is continuous for $(t, x) \in (0, \infty) \times [a, b]$ and the integral (8.1.8) is absolutely convergent. Then

$$E_k(f, [a, b]) \leq \int_0^\infty E_k(g_t(x), [a, b])\ \mathrm{d}t, \tag{8.1.9}$$

for all $k \geq 0$.

Proof. We can assume that the right-hand side of (8.1.9) is finite, since otherwise the bound (8.1.9) would be automatically satisfied. For all $t \in (0, \infty)$ and for any degree $k \geq 0$, since $g_t(x)$ is continuous over the compact interval $[a, b]$, there exists a unique polynomial $p_k^{(t)}(x) \in \Pi_k$ such that

$$E_k(g_t(x), [a, b]) = \max_{x \in [a,b]} |g_t(x) - p_k^{(t)}(x)|.$$

For all $x \in [a, b]$ we can define

$$p_k(x) := \int_0^\infty p_k^{(t)}(x)\ \mathrm{d}t.$$

This is well defined: for all $x \in [a, b]$, the mapping $t \mapsto p_k^{(t)}(x)$ is continuous for all $t \in (0, \infty)$ in view of [115, Theorem 24], since $g_t(x)$ is uniformly continuous for (t, x) in the compact sets contained in $(0, \infty) \times [a, b]$. Moreover, the integral is finite since

$$\begin{aligned}\int_0^\infty |p_k^{(t)}(x)|\, \mathrm{d}t &\leq \int_0^\infty |g_t(x)|\, \mathrm{d}t + \int_0^\infty |g_t(x) - p_k^{(t)}(x)|\, \mathrm{d}t \\ &\leq \int_0^\infty |g_t(x)|\, \mathrm{d}t + \int_0^\infty E_k(g_t(x), [a, b])\, \mathrm{d}t < +\infty.\end{aligned}$$

We now show that $p_k(x)$ is also a polynomial. Consider the expression

$$p_k^{(t)}(x) = \sum_{i=0}^{k} a_i(t)\, x^i,$$

which defines the coefficients $a_i(t)$ as functions of t. Formally, we have

$$p_k(x) = \sum_{i=0}^{k} \left(\int_0^\infty a_i(t)\ \mathrm{d}t \right) \cdot x^i = \sum_{i=0}^{k} \widetilde{a}_i x^i,$$

where $\widetilde{a}_i := \int_0^\infty a_i(t)\ \mathrm{d}t$. To conclude, we need to show that this expression is well defined, that is, the functions $a_i(t)$ are integrable in t. Consider $k+1$ distinct points $x_0, \ldots, x_k$ in the interval $[a, b]$ and let $\boldsymbol{a}(t) = [a_0(t), \ldots, a_k(t)]^T$, $\boldsymbol{p}(t) = [p_k^{(t)}(x_0), \ldots, p_k^{(t)}(x_k)]^T$. We have that $V\boldsymbol{a}(t) = \boldsymbol{p}(t)$, where

$$V = \begin{bmatrix} 1 & x_0 & x_0^2 & \cdots & x_0^k \\ 1 & x_1 & x_1^2 & \cdots & x_1^k \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_k & x_k^2 & \cdots & x_k^k \end{bmatrix}$$

is a Vandermonde matrix. Since V is nonsingular, we have that $\boldsymbol{a}(t) = V^{-1}\boldsymbol{p}(t)$. Then, if we let $V^{-1} = (c_{ij})_{i,j=0}^k$, we obtain the expression

$$a_i(t) = \sum_{j=0}^{k} c_{ij} p_k^{(t)}(x_j), \quad i = 0, \ldots, k,$$

where c_{ij} is independent of t for all i, j. This shows that, for all i, $a_i(t)$ is continuous for $t \in (0, \infty)$, and we have

$$|\widetilde{a}_i| \leq \int_0^\infty |a_i(t)|\ \mathrm{d}t \leq \sum_{j=0}^{k} |c_{ij}| \int_0^\infty \left| p_k^{(t)}(x_j) \right|\ \mathrm{d}t < +\infty, \quad i = 0, \ldots, k,$$

hence $\widetilde{a}_i$ is well defined for $i = 0, \ldots, k$ and $p_k(x)$ is a polynomial of degree at most k. Finally, we have

$$\begin{aligned} E_k(f, [a, b]) &\leq \max_{x \in [a,b]} |f(x) - p_k(x)| \\ &\leq \max_{x \in [a,b]} \int_0^\infty |g_t(x) - p_k^{(t)}(x)|\ \mathrm{d}t \\ &\leq \int_0^\infty E_k(g_t(x), [a, b])\ \mathrm{d}t. \end{aligned}$$

□

The following result gives a new bound for $E_k(x \log x, [a, b])$ by exploiting (8.1.7).

Theorem 8.1.2. Let $0 \leq a < b$ and $\gamma = a/b$. We have

$$E_k(x \log(x), [a, b]) \leq b\,(1 - \sqrt{\gamma}) \frac{1 + \gamma + 2k\sqrt{\gamma}}{4(k^2 - 1)} \left(\frac{1 - \sqrt{\gamma}}{1 + \sqrt{\gamma}} \right)^k, \tag{8.1.10}$$

for all $k \geq 2$.

Proof. Let $f(x) = x \log(x)$. Notice that $f(x) = bf(b^{-1}x) + x \log b$, so for $k \geq 2$ we have

$$E_k(f(x), [a, b]) = E_k(bf(b^{-1}x), [a, b]) = b\, E_k(f(x), [\gamma, 1]),$$

since we can ignore terms of degree 1 for the polynomial approximation. For $x \in [\gamma, 1]$ we can use the integral representation (8.1.7). Let $g_t(x) := \frac{x(1-x)}{(1+t)(x+t)}$ be the integrand in (8.1.7) for all $t > 0$. Note

that $g_t(x)$ is a continuous function in the variables $(t,x) \in (0,\infty)\times[a,b]$, so we can apply Lemma 8.1.1. We can write $g_t(x)$ as

$$g_t(x) = \frac{x}{x+t} - \frac{x}{1+t} = 1 - \frac{t}{x+t} - \frac{x}{1+t}, \tag{8.1.11}$$

and since $k \geq 2$ and $1 - \frac{x}{1+t}$ is a polynomial of degree 1, we get that

$$E_k(g_t(x),[\gamma,1]) = E_k(t/(x+t),[\gamma,1]) = tE_k(1/x,[\gamma+t,1+t]).$$

Hence, by using (8.1.3) and Lemma 8.1.1 we get

$$E_k(x\log(x),[a,b]) \leq b\int_0^\infty \frac{t\left(\sqrt{\kappa(t)}+1\right)^2}{2(1+t)}\left(\frac{\sqrt{\kappa(t)}-1}{\sqrt{\kappa(t)}+1}\right)^{k+1} \mathrm{d}t, \tag{8.1.12}$$

where $\kappa(t) = (1+t)/(\gamma+t)$. In order to bound the integral, consider the identities

$$\frac{\sqrt{\kappa(t)}-1}{\sqrt{\kappa(t)}+1} = \frac{\left(\sqrt{1+t}-\sqrt{\gamma+t}\right)^2}{1-\gamma}, \qquad \left(\sqrt{\kappa(t)}+1\right)^2 = \frac{\left(\sqrt{1+t}+\sqrt{\gamma+t}\right)^2}{\gamma+t}.$$

We have

$$\begin{aligned}
&\int_0^\infty \frac{t\left(1+\sqrt{\kappa(t)}\right)^2}{2(1+t)}\left(\frac{\sqrt{\kappa(t)}-1}{\sqrt{\kappa(t)}+1}\right)^{k+1} \mathrm{d}t \\
&\quad= \frac{1}{2}\int_0^\infty \frac{t}{(\gamma+t)(1+t)}\cdot\frac{\left(\sqrt{1+t}+\sqrt{\gamma+t}\right)^2\left(\sqrt{1+t}-\sqrt{\gamma+t}\right)^{2k+2}}{(1-\gamma)^{k+1}}\,\mathrm{d}t \\
&\quad\leq \frac{1}{2(1-\gamma)^{k-1}}\int_0^\infty \left(\sqrt{1+t}-\sqrt{\gamma+t}\right)^{2k}\,\mathrm{d}t.
\end{aligned} \tag{8.1.13}$$

By checking the derivative, it can be shown that

$$F(t) = \frac{\sqrt{1+t}\sqrt{\gamma+t}\left(\sqrt{1+t}-\sqrt{\gamma+t}\right)^{2k}\left(\frac{\left(\sqrt{1+t}-\sqrt{\gamma+t}\right)^4}{2k+2} - \frac{(1-\gamma)^2}{2k-2}\right)}{2\left(\sqrt{1+t}\sqrt{\gamma+t}-\gamma-t\right)\left(1+t-\sqrt{1+t}\sqrt{\gamma+t}\right)}$$

is an antiderivative of $\left(\sqrt{1+t}-\sqrt{\gamma+t}\right)^{2k}$. Since $\lim_{t\to\infty} F(t) = 0$, we deduce that

$$\begin{aligned}
\int_0^\infty \left(\sqrt{1+t}-\sqrt{\gamma+t}\right)^{2k}\,\mathrm{d}t &= \frac{\sqrt{\gamma}\left(1-\sqrt{\gamma}\right)^{2k}\left(\frac{(1-\gamma)^2}{2k-2} - \frac{\left(1-\sqrt{\gamma}\right)^4}{2k+2}\right)}{2\left(1-\sqrt{\gamma}\right)\left(\sqrt{\gamma}-\gamma\right)} \\
&= 2(1-\sqrt{\gamma})^{2k}\frac{1+\gamma+2\sqrt{\gamma}\,k}{2(k^2-1)}.
\end{aligned}$$

This, combined with (8.1.13), concludes the proof. □

Remark 8.1.3. The result of Theorem 8.1.2 is an improvement of (8.1.5). When $a > 0$, the right-hand side in (8.1.10) is the product of an algebraic and a geometric factor, so the decay is asymptotically faster. When $a = 0$, we have $\gamma = 0$ and (8.1.10) becomes

$$E_k(x\log(x),[0,b]) \leq \frac{b}{4(k^2-1)},$$

which is better than (8.1.5) for all $k \geq 2$. The new bound (8.1.10) is compared with (8.1.5) and with the polynomial approximation error in Figure 8.1.

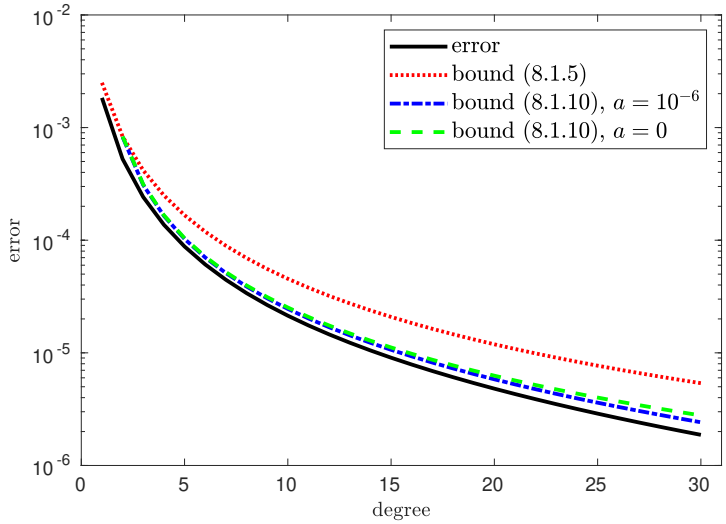


Figure 8.1 – Comparison of the bounds (8.1.5) and (8.1.10) with the error of the polynomial approximation of the entropy function $x \log x$ on the interval $[10^{-6}, 10^{-2}]$.

8.2 Computation of the von Neumann entropy

In this section we describe and analyze an approach for the approximation of the von Neumann entropy $S(A) = \operatorname{tr}(-A \log A)$, which combines a probing method or a stochastic trace estimator with polynomial and rational Krylov subspace methods for computing the matrix-vector products and quadratic forms with $f(A)$ that arise in the trace approximation.

Recall that given a distance-d coloring of the graph $\mathcal{G}(A)$ associated with a sparse matrix A, the probing approximation of $\operatorname{tr}(f(A))$ is given by

$$\operatorname{tr}_d^{\mathrm{P}}(f(A)) = \sum_{j=1}^{s} \boldsymbol{v}_\ell^T f(A) \boldsymbol{v}_\ell,$$

where s is the number of colors and $\boldsymbol{v}_\ell$ is the probing vector associated with color ℓ, which has ones on components corresponding to nodes of color ℓ and zero elsewhere. The accuracy of the probing approximation of $\operatorname{tr}(f(A))$ depends on the approximation of f with polynomials; we will focus on this aspect in Section 8.2.1, where we apply the polynomial approximation results of Section 8.1.1 in order to bound the error of the probing method for the entropy. This approach can only be used for sparse A, and it is especially effective when distance-d colorings with a small number of colors can be computed. This occurs when $\mathcal{G}(A)$ has a large-world structure, for instance if it represents a grid or a road network; see the experiments in Section 8.4.

On the other hand, as a stochastic trace estimator we use the adaptive implementation of Hutch++ from [129] for its flexibility, since it is automatically able to select its parameters (number of sample vectors for the low-rank approximation and for Hutchinson's trace estimator) according to each specific problem. This approach can be used for both sparse and dense matrices, and it particularly suited for computing cheap low-accuracy approximations (see the discussion in Section 7.1). Nevertheless, this method struggles when a high accuracy on $\operatorname{tr}(f(A))$ is required, unless there is significant decay in the singular values of $f(A)$, which can be exploited by the low-rank part of Hutch++ to quickly reduce the error.

Both the probing method and Hutch++ estimator require the approximation of several quadratic forms $\boldsymbol{b}^T f(A) \boldsymbol{b}$, where $f(x) = -x \log x$, which can be computed efficiently by using Krylov subspace methods; the Hutch++ algorithm also requires the computation of some matrix-vector products $f(A)\boldsymbol{b}$ in its low-rank approximation phase. We focus on this task in Section 8.2.2.

8.2.1 Convergence of the probing method

Let $A \in \mathbb{R}^{n\times n}$ be a sparse symmetric matrix with spectrum contained in $[a, b]$, and let $\operatorname{tr}_d^{\mathrm{P}}(f(A))$ be the approximation of the von Neumann entropy $S(A)$ obtained by using the probing method with a distance-d coloring. The bound on the polynomial approximation of the function $f(x) = x \log x$ given in Theorem 8.1.2 can be combined with Theorem 7.2.1 to obtain a bound on the error of the probing method.

Theorem 8.2.1. Let $A \in \mathbb{R}^{n\times n}$ be symmetric with $\Lambda(A) \subset [a, b]$, $0 \le a < b$, and let $\gamma = a/b$. Then

$$|S(A) - \operatorname{tr}_d^{\mathrm{P}}(-A\log(A))| \le n\, b\, (1-\sqrt{\gamma})\frac{1+\gamma+2d\sqrt{\gamma}}{2(d^2-1)}\left(\frac{1-\sqrt{\gamma}}{1+\sqrt{\gamma}}\right)^d, \tag{8.2.1}$$

for all $d \ge 2$. In particular, if $a = 0$, we have

$$|S(A) - \operatorname{tr}_d^{\mathrm{P}}(-A\log(A))| \le \frac{nb}{2(d^2-1)}, \tag{8.2.2}$$

for all $d \ge 2$.

Proof. The inequality (8.2.1) follows from Theorem 8.1.2 and Theorem 7.2.1. The inequality (8.2.2) follows from (8.2.1) with $\gamma = a/b = 0$. □

Remark 8.2.2. The bound (8.2.1) can be pessimistic in practice, especially for large values of d. We show this in Example 8.2.1 and in the experiments of Section 8.4.2. A priori bounds based on polynomial approximations often fail to capture the exact convergence behavior in many problems related to matrix functions, since a minimization problem over the spectrum of a matrix is relaxed to the whole spectral interval. This occurs in the convergence of polynomial Krylov methods [106, Section 5.6], as well as in the decay bounds on the entries of matrix functions [19, 76]. In addition, the coloring obtained via Algorithm 7.2 can sometimes return far more colors than the minimum number needed for a distance-d coloring, and this can benefit the convergence in a way that is not predicted by the bound. In Section 8.3.1 we are going to propose a more practical heuristic to predict the error with higher accuracy.

Example 8.2.1. Let us see with an example how the bound and the convergence of the probing method perform in practice. We consider the density matrix $A = L/\operatorname{tr}(L)$, where L is the Laplacian of the graph `minnesota` from the SuiteSparse Matrix Collection [51]. More precisely, we consider its largest connected component, whose graph Laplacian has size $n = 2640$ and 9244 nonzero entries. For several values of d, we compute the approximation $\operatorname{tr}_d^{\mathrm{P}}(-A\log A)$ associated with two different distance-d colorings: the first is obtained by the greedy coloring (Algorithm 7.2) after sorting the nodes by descending degree, while for the second we use the bandwidth-reducing reverse Cuthill-McKee algorithm to get a 67-banded matrix and then we use the coloring (7.2.3) for banded matrices.

In Figure 8.2 we compare $\operatorname{tr}_d^{\mathrm{P}}(-A\log(A))$ with the value of $S(A)$ obtained by diagonalizing A, considered as exact. The left plot shows the error in terms of the number of colors (which coincides with the number of probing vectors) associated with the distance-d colorings. In the right plot the errors are shown in terms of d together with the bound (8.2.2), where $b = \lambda_{\max}(A)$. For each fixed value of d, the coloring based on the bandwidth provides a smaller error than the greedy coloring, but it also uses a larger number of colors. Indeed, we can see from the left plot that with the same computational effort the greedy algorithm obtains a smaller error. On the right plot, we can observe that the bound (8.2.2) is close to the error given by the greedy coloring for small values of d, while it fails to capture the convergence behavior for large values of d.

We conclude this section by showing that for the graph entropy, i.e., when $A = L/\operatorname{tr}(L)$ and L is the Laplacian of a graph G, the approximation $\operatorname{tr}_d^{\mathrm{P}}(-A\log A)$ is a lower bound for $S(A)$. To prove this, we use the fact that A is a symmetric M-matrix (Definition 6.1.1), i.e., it can be written in the form $A = sI - B$ where $s > 0$ and B is a nonnegative matrix such that $\lambda \le s$ for all $\lambda \in \Lambda(B)$.

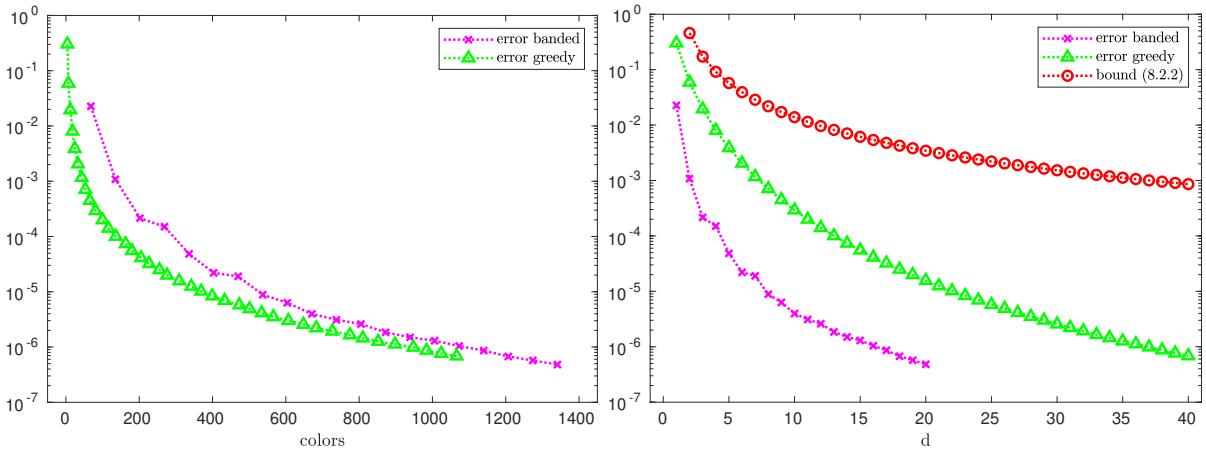


Figure 8.2 – Absolute errors of the probing approximation of $S(A)$ where the distance-d coloring is obtained either by the greedy procedure (Algorithm 7.2) or with the reverse Cuthill-McKee algorithm and the coloring (7.2.3) for banded matrices. Left: Error as a function of the number of colors. Right: Error as a function of the distance d, compared with the bound (8.2.2).

Lemma 8.2.3. Let $A = sI - B$ be a symmetric M-matrix, with $B \geq 0$ and $s \geq \|B\|_2$, and let $\mathrm{dist}(i,j)$ be the geodesic distance in the graph associated with A. Then we have $[-A\log(A)]_{ij} \leq 0$ whenever $\mathrm{dist}(i,j) \geq 2$. Moreover, if $s < e^{-1}$, we have $[-A\log(A)]_{ij} \leq 0$ for $i \neq j$ and $-A\log A$ is an M-matrix.

Proof. Assume first that $s > \|B\|_2$, so that A is nonsingular. Then $f(x) = -x\log x$ is analytic for $|x - s| < \|B\|_2$ and it can be represented by the power series

$$f(x) = \sum_{k=0}^{\infty} \frac{f^{(k)}(s)}{k!}(x-s)^k.$$

Since $\Lambda(A) \subset (s - \|B\|_2, s + \|B\|_2)$, the same series expansion holds for $f(A)$,

$$f(A) = \sum_{k=0}^{\infty} \frac{f^{(k)}(s)}{k!}(A - sI)^k = -s\log sI + (1 + \log s)B - \sum_{k=2}^{\infty} \frac{s^{-(k-1)}}{k(k-1)}B^k.$$

If $\mathrm{dist}(i,j) \geq 2$ we have $[B]_{ij} = [-A]_{ij} = 0$, and B^k is nonnegative for all $k \geq 0$, so $[f(A)]_{ij} \leq 0$. If $s < e^{-1}$, then $(1 + \log s)B$ is nonpositive so $[f(A)]_{ij} \leq 0$ for $i \neq j$, and it is an M-matrix since it is positive semidefinite [28, Theorem 4.6].

If A is singular, consider the nonsingular M-matrix $A + \varepsilon I$ for $\varepsilon > 0$. Then the results hold for $f(A+\varepsilon I)$ (we impose $s+\varepsilon < e^{-1}$ if $s < e^{-1}$) and notice that $f(A) = \lim_{\varepsilon \to 0} f(A+\varepsilon I)$. This concludes the proof, since the limit of nonsingular M-matrices is an M-matrix. □

Proposition 8.2.4. Let $A \in \mathbb{R}^{n\times n}$ be a symmetric M-matrix, and let $\mathrm{tr}_d^{\mathrm{P}}(-A\log(A))$ be the approximation (7.2.2) of $S(A)$ induced by a distance-d coloring of $\mathcal{G}(A)$ with $d \geq 1$. Then $\mathrm{tr}_d^{\mathrm{P}}(-A\log(A)) \leq S(A)$.

Proof. If $V_1, \ldots, V_s$ is the graph partitioning associated with a distance-d coloring, the error of the approximation can be written as

$$S(A) - \mathrm{tr}_d^{\mathrm{P}}(-A\log(A)) = -\sum_{\ell=1}^{s} \sum_{\substack{i,j \in V_\ell \\ i \neq j}} [-A\log(A)]_{ij}, \tag{8.2.3}$$

see [77, eq. (4.1)] for more details. By definition of a distance-d coloring, for all $i, j \in V_\ell$ such that $i \neq j$, we have $\mathrm{dist}(i,j) \geq d + 1 \geq 2$. Then, in view of Lemma 8.2.3, the right-hand side of (8.2.3) is nonnegative. □

Remark 8.2.5. In this chapter we focus only on the approximation of the entropy of sparse density matrices. However, an important case is given by $A = g(H)$, where H is the Hamiltonian of a certain quantum system and g is a function defined on the spectrum of H. Notable examples are the Gibbs state [46, 155] and the Fermi-Dirac state [1]. Despite A being a dense matrix in general, a sparse structure of H implies that A exhibits decay properties and can be well approximated by a sparse matrix [12, 14]. Moreover, since we have $S(A) = \operatorname{tr}(-g(H)\log g(H))$ one can apply the techniques described in this section to the composition $-g(x)\log g(x)$. Although the investigation of this problem is outside the scope of this thesis, it represents an interesting direction for future research.

8.2.2 Computation of quadratic forms with Krylov subspace methods

Recall that the function $f(x) = x\log x$ is the product of a Cauchy-Stieltjes function with a polynomial of degree two, see (8.1.7). Hence, we can expect to have fast convergence for a rational Krylov method for approximating $\boldsymbol{b}^T f(A)\boldsymbol{b}$ if we use the EDS poles for Cauchy-Stieltjes functions from Section 2.5.3 in conjunction with two (or more) poles at infinity, to account for the polynomial factor.

In practice, we noticed that the first few iterations of a polynomial Krylov method for the function $f(x) = x\log x$ have quite a fast convergence rate, and the convergence then becomes asymptotically much slower, especially for ill conditioned matrices. The faster initial convergence can be explained by the algebraic factor in the bound (8.1.10) for polynomial approximations of $x\log x$. Since a polynomial Krylov iteration is cheaper than a rational Krylov iteration, this observation suggests the use of a hybrid polynomial-rational method, that starts with a few polynomial Krylov steps and then switches to a rational Krylov method with quasi-optimal EDS poles for Cauchy-Stieltjes functions to achieve a higher accuracy.

As a stopping criterion for the approximation of $\boldsymbol{b}^T f(A)\boldsymbol{b}$, we use the a posteriori bounds from Theorem 3.2.3. We test the proposed approach numerically in the following example.

Example 8.2.2. Let us consider the computation of $\boldsymbol{b}^T f(A)\boldsymbol{b}$, for a 2000×2000 symmetric tridiagonal matrix A with eigenvalues given by the Chebyshev points for the interval $\Sigma = [10^{-3}, 10^3]$, a random vector $\boldsymbol{b}$ and $f(x) = x\log(x)$. We compare the polynomial Krylov method, the rational Krylov method with the EDS poles for Cauchy-Stieltjes functions from Section 2.5.3, and a hybrid polynomial-rational Krylov method with 10 poles at ∞ (which correspond to polynomial Krylov steps) followed by 10 EDS poles. For each method, we also show the upper and lower error bound from Theorem 3.2.3, as well as the heuristic estimate of the error given by the geometric mean of the upper and lower bound,

$$\texttt{est}_m = \|\boldsymbol{b}\|_2^2\sqrt{\min_{z\in\Sigma}|g_m(z)|\ \max_{z\in\Sigma}|g_m(z)|}, \tag{8.2.4}$$

which was shown to be quite accurate in Example 3.2.1. The upper and lower bounds are computed numerically by evaluating g_m on a discretization of the interval Σ. The results are shown in Figure 8.3. Since the polynomial Krylov method converges quite quickly for the first few iterations, we can see that the purely rational and the hybrid polynomial-rational method have approximately the same convergence curve, with the latter having a much lower cost for the first 10 iterations. The upper and lower bounds match the shape of the convergence curve quite well, although they are less accurate when polynomial iterations are used. The error estimate (8.2.4) is very accurate also when polynomial iterations are used, a setting in which the accuracy of the upper and lower error bounds is instead significantly reduced.

Remark 8.2.6. In the context of the graph entropy, the matrix A is a (normalized) graph Laplacian, which is a singular matrix. Therefore, in principle, it is not possible to use the EDS poles for Cauchy-Stieltjes functions, since they require the assumption that A is positive definite. However, we can use one of the desingularization strategies described in Chapter 6 to remove the 0 eigenvalue of the graph Laplacian, obtaining a matrix with spectrum contained in $[\lambda_2, \lambda_n]$, where λ_2 is the second smallest eigenvalue of A and λ_n is the largest one. In our implementation we use the implicit desingularization approach, i.e., we replace the initial vector $\boldsymbol{b}$ for the Krylov subspace with $\boldsymbol{c} = \boldsymbol{b} - \frac{\mathbf{1}^T\boldsymbol{b}}{n}\mathbf{1}$. Since $\boldsymbol{c}$ is orthogonal to the eigenvector $\mathbf{1}$ associated to the eigenvalue 0, this guarantees that the convergence of a Krylov subspace method with starting vector $\boldsymbol{c}$ is the same as for a matrix with spectrum in $[\lambda_2, \lambda_n]$, at least in exact arithmetic, see Theorem 6.2.3.

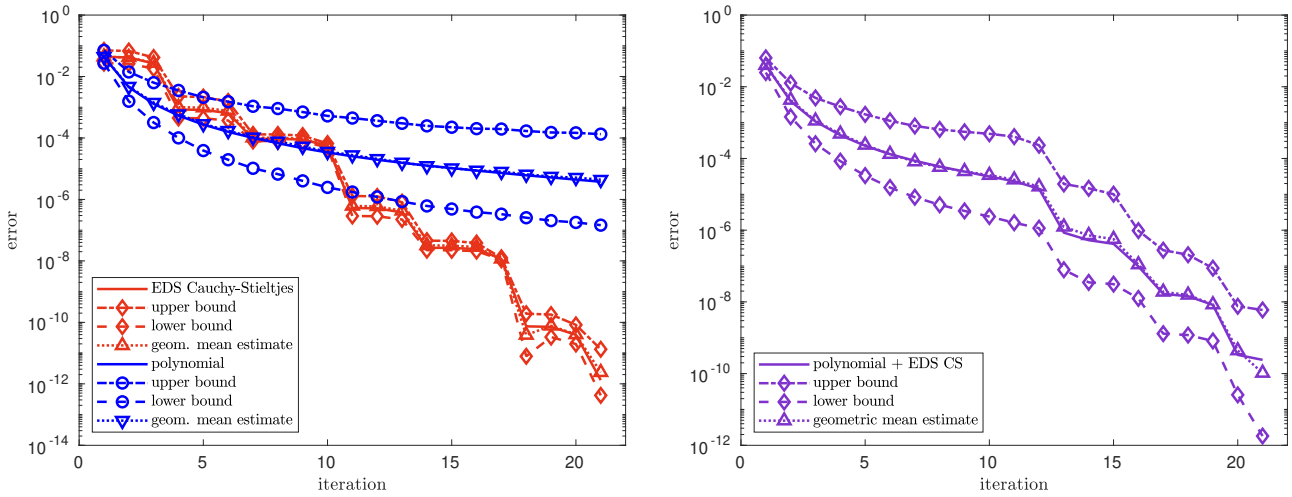


Figure 8.3 – Accuracy of error bounds and estimates for the relative error in the computation of $\boldsymbol{b}^T f(A)\boldsymbol{b}$ with Krylov subspace methods, with $f(x) = x\log(x)$ and a 2000×2000 symmetric matrix A. Left: polynomial Krylov and rational Krylov with EDS poles for Cauchy-Stieltjes function. Right: hybrid polynomial-rational Krylov with 10 poles at ∞ and 10 EDS poles.

8.3 Implementation aspects

In this section we discuss some technical details of the algorithm for computing the entropy and we briefly comment on some of the decisions that must be made in an implementation, especially concerning stopping criteria. Given a symmetric positive semidefinite matrix A with $\mathrm{tr}(A) = 1$ and a target relative accuracy ε, the algorithm should output an estimate `trest` of $\mathrm{tr}(f(A))$, where $f(x) = -x\log x$, such that

$$|\mathrm{tr}(f(A)) - \texttt{trest}| \leq \varepsilon\,\mathrm{tr}(f(A)),$$

using either the probing approach or a stochastic trace estimator. Observe that the entropy of an $n \times n$ density matrix is always bounded from above by $\log n$, but it may be in principle very small, so we prefer to aim for a certain relative accuracy rather than an absolute accuracy. Quadratic forms and matrix-vector products with $f(A)$ that arise within either trace estimation approach are computed using Krylov subspace methods, specifically using a certain number of poles at ∞ followed by EDS poles for Cauchy-Stieltjes functions, as described in Section 8.2.2 and showcased in Example 8.2.2.

Remark 8.3.1. In the following, we are going to use $\widehat{\varepsilon}$ to denote an absolute error, to distinguish it from the target relative accuracy ε. Note that we can easily transform absolute inequalities for the error into relative inequalities if we know in advance an estimate or a lower bound for $\mathrm{tr}(f(A))$. Recall that if A is an M-matrix, $\mathrm{tr}_d^{\mathrm{P}}(f(A))$ is actually a lower bound for $\mathrm{tr}(f(A))$ (Proposition 8.2.4). In the general case, any rough approximation of the entropy can be used for this purpose, since the important point is determining the order of magnitude of $\mathrm{tr}(f(A))$.

Remark 8.3.2. The error in the approximation of $S(A)$ can be divided into the error in the approximation of the trace using a probing method or a stochastic estimator, and the error in the approximation of the quadratic forms with $f(A)$ using a Krylov subspace method. For simplicity, in the following we impose that the relative error associated to each of these two components is smaller than $\varepsilon/2$.

8.3.1 Probing method implementation

We begin by observing that it is not possible to cheaply estimate the error of a probing method a posteriori, since error estimates are usually based on computing differences of approximations obtained with different values of the distance d, which in general lead to completely different colorings and hence would require computing all quadratic forms from scratch, essentially doubling the computational cost

simply to compute an error estimate. For this reason, it is best to find a value of d that ensures a relative accuracy ε a priori when using a distance-d coloring. This can be done using one of the bounds in Corollary 8.2.1, but it can often lead to unnecessary additional work, since the bounds usually overestimate the error by a couple of orders of magnitude, and hence they would recommend using a distance d much larger that necessary; see for instance Figure 8.2.

To try and overcome this issue, we also provide a heuristic criterion for choosing d that does not have the same theoretical guarantee as the bounds, but appears to work quite well in practice. In view of Corollary 8.2.1, we can expect the absolute error to behave as

$$|\mathrm{tr}(f(A)) - \mathrm{tr}_d^{\mathrm{P}}(f(A))| \sim \frac{C}{d^k} q^d, \tag{8.3.1}$$

for $k = 2$ and some parameters $C > 0$ and $q \in (0, 1)$. However, we found that sometimes the actual error behavior is better described with a different value of k, such as $k = 3$, so we do not impose the restriction $k = 2$. To estimate the values of the parameters, we compute $\mathrm{tr}_d^{\mathrm{P}}(f(A))$ for $d = 1, 2, 3$ and use the estimate

$$|\mathrm{tr}(f(A)) - \mathrm{tr}_d^{\mathrm{P}}(f(A))| \approx |\mathrm{tr}_{d+1}^{\mathrm{P}}(f(A)) - \mathrm{tr}_d^{\mathrm{P}}(f(A))|, \qquad d = 1, 2. \tag{8.3.2}$$

Assuming that (8.3.1) and (8.3.2) hold exactly and fixing a value of k, we can determine C and q by solving the equations

$$\left|\mathrm{tr}_{d+1}^{\mathrm{P}}(f(A)) - \mathrm{tr}_d^{\mathrm{P}}(f(A))\right| = \frac{C}{d^k} q^d, \qquad d = 1, 2.$$

We can evaluate the resulting estimate to heuristically determine the smallest d such that the estimate is smaller than $\widehat{\varepsilon}$, i.e., we select d as

$$d_\star = \min\left\{ d \,:\, \frac{C}{d^k} q^d \leq \widehat{\varepsilon} \right\},$$

in order to have the approximate absolute error inequality

$$\left|\mathrm{tr}(f(A)) - \mathrm{tr}_{d_\star}^{\mathrm{P}}(f(A))\right| \lesssim \widehat{\varepsilon}.$$

We found that the best results are obtained for $k = 2$ and $k = 3$, so in our implementation we use the maximum of the two corresponding estimates. Variants of this estimate include using other values of d to estimate the parameters instead of $d = 1, 2, 3$, and using four different values in order to also estimate the parameter k. However, they usually give results that are similar or sometimes worse than the estimate presented above, so they are often not worth the additional computational effort required to compute them. In particular, using four values of d increases the risk of misjudging the value of the geometric rate q, causing the estimate to be inaccurate for large values of d. The error estimate (8.3.1) is compared to the actual error and the theoretical bound (8.2.1) for two different graphs in Figure 8.4. The plot also includes a simple error estimate based on consecutive differences, which requires the computation of $\mathrm{tr}_{d+1}^{\mathrm{P}}(f(A))$ to estimate the error for $\mathrm{tr}_d^{\mathrm{P}}(f(A))$; since usually all the quadratic forms associated with the distance-$(d+1)$ coloring have to be computed from scratch, using this estimate increases the computational effort by a factor of at least two, and hence its use is not recommended.

Remark 8.3.3. The heuristic criterion for selecting d requires the computation of $\mathrm{tr}_d^{\mathrm{P}}(f(A))$ for $d = 1, 2, 3$, so it is more expensive to use than the theoretical bound (8.2.1). However, this cost is usually small compared to the cost of computing $\mathrm{tr}_d^{\mathrm{P}}(f(A))$ for the selected value of d, especially if the requested accuracy is high and hence the selected d is large. Note also that the heuristic criterion always computes $\mathrm{tr}_3^{\mathrm{P}}(f(A))$, so it does more work than necessary when $d \leq 2$ would be sufficient. Nevertheless, in such a situation the theoretical bound (8.2.1) may suggest to use an even higher value of d (see Figure 8.4) and would therefore be even more expensive.

After choosing d such that

$$|\mathrm{tr}(f(A)) - \mathrm{tr}_d^{\mathrm{P}}(f(A))| \leq \widehat{\varepsilon}, \tag{8.3.3}$$

using either the a priori bound (8.2.1) or the estimate (8.3.1), a distance-d coloring can be computed with one of the coloring algorithms described in Section 7.2, depending on the properties of the graph.

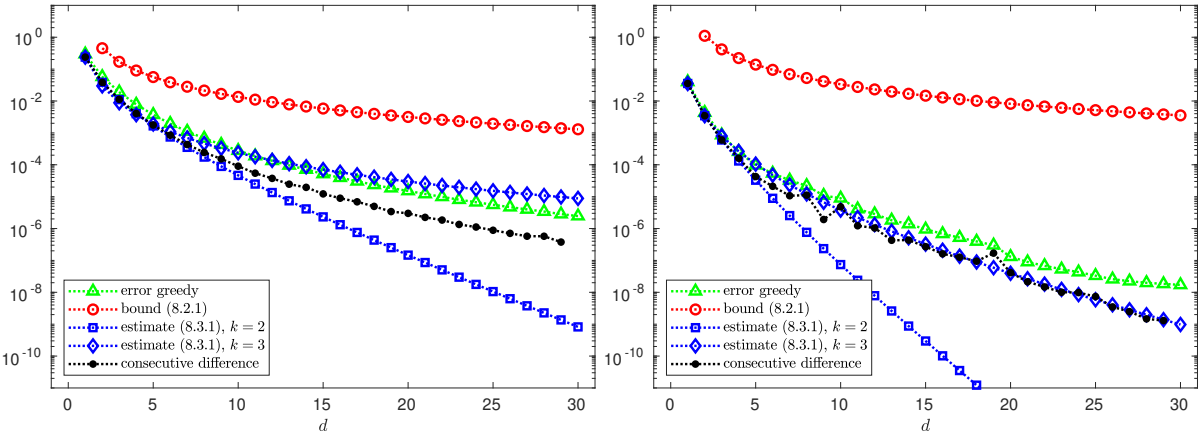


Figure 8.4 – Absolute error of the probing method with greedy coloring (Algorithm 7.2) compared with the theoretical bound (8.2.1) using $a = \lambda_2$, the heuristic error estimate (8.3.1) and the simple error estimate $|\mathrm{tr}(f(A) - \mathrm{tr}_d^{\mathrm{P}}(f(A))| \approx |\mathrm{tr}_{d+1}^{\mathrm{P}}(f(A)) - \mathrm{tr}_d^{\mathrm{P}}(f(A))|$. Left: Laplacian of the largest connected component of the graph `minnesota`, with 2640 nodes. Right: Laplacian of the largest connected component of the graph `eris1176`, with 1174 nodes.

The greedy coloring Algorithm 7.2 is usually a good choice for general graphs. In our MATLAB implementation of the probing method, we construct the greedy distance-d coloring by computing A^d, since we found it to be faster than the graph-based approach; this is likely due to the more efficient MATLAB implementation of matrix-matrix operations.

Let us now determine the accuracy required in the computation of the quadratic forms. Assume that we have

$$\mathrm{tr}_d^{\mathrm{P}}(f(A)) = \sum_{\ell=1}^{s} \boldsymbol{v}_\ell^T f(A) \boldsymbol{v}_\ell,$$

where $\{\boldsymbol{v}_\ell\}_{\ell=1}^s$ are the probing vectors used in the distance-d coloring. Let $\widehat{\psi}_\ell$ denote the approximation of $\boldsymbol{v}_\ell^T f(A) \boldsymbol{v}_\ell$ obtained with a Krylov method. Recall that $\|\boldsymbol{v}_\ell\|_2 = |V_\ell|^{1/2}$, where V_ℓ denotes the set of the partition associated to the ℓ-th color. If we impose the conditions

$$\left| \boldsymbol{v}_\ell^T f(A) \boldsymbol{v}_\ell - \widehat{\psi}_\ell \right| \leq \widehat{\varepsilon} \cdot \frac{|V_\ell|}{n} \qquad \ell = 1, \ldots, s, \tag{8.3.4}$$

where we normalized the accuracy requested for each quadratic form depending on $\|\boldsymbol{v}_\ell\|_2$, we obtain the desired absolute accuracy on the probing approximation

$$\left| \mathrm{tr}_d^{\mathrm{P}}(f(A)) - \sum_{\ell=1}^{s} \widehat{\psi}_\ell \right| \leq \widehat{\varepsilon}, \tag{8.3.5}$$

where we used the fact that $\sum_{\ell=1}^s |V_\ell| = n$. If we are aiming for a relative accuracy ε, we should select $\widehat{\varepsilon} = \frac{1}{2}\varepsilon\,\mathrm{tr}(f(A))$ in (8.3.3) and (8.3.5). In practice, $\mathrm{tr}(f(A))$ will be replaced by a rough approximation (see Remark 8.3.1). The overall probing algorithm is summarized in Algorithm 8.1.

8.3.2 Adaptive Hutch++ implementation

As a stochastic estimator, we use the MATLAB code of the adaptive Hutch++ algorithm ([129, Algorithm 3]) provided by the authors, modified to use Krylov methods for the computations with $f(A)$. This algorithm requires as inputs an absolute tolerance $\widehat{\varepsilon}$ and a failure probability δ, and outputs an approximation $\mathrm{tr}_{\mathrm{adap}}(f(A))$ such that

$$\mathbb{P}\left[|\mathrm{tr}(f(A)) - \mathrm{tr}_{\mathrm{adap}}(f(A))| \geq \widehat{\varepsilon}\right] \leq \delta.$$

Algorithm 8.1 Probing method for $S(A)$

Input: $A \in \mathbb{R}^{n\times n}$ density matrix, ε relative error tolerance
Output: $\texttt{trest} \approx S(A)$ such that $|\texttt{trest} - S(A)|/S(A) \lesssim \varepsilon$

1: Select d such that $|\mathrm{tr}_d^{\mathrm{P}}(f(A)) - S(A)|/S(A) \lesssim \varepsilon/2$, using either the bound (8.2.1) or the heuristic (8.3.1). The heuristic (8.3.1) requires the computation of $\mathrm{tr}_d^{\mathrm{P}}(f(A))$ for $d = 1, 2, 3$, which can be done by running steps 2 – 4.
2: Compute a distance-d coloring of $\mathcal{G}(A)$ with, e.g., Algorithm 7.2, and the associated probing vectors $\{\boldsymbol{v}_1, \dots, \boldsymbol{v}_s\}$.
3: For $\ell = 1, \dots, s$, compute $\widehat{\psi}_\ell \approx \boldsymbol{v}_\ell^T f(A)\boldsymbol{v}_\ell$ such that (8.3.4) holds, using a rational Krylov method with either the upper bound from Theorem 3.2.3 or the estimate (8.2.4) as stopping criterion.
4: **return** $\texttt{trest} = \sum_{\ell=1}^{s} \widehat{\psi}_\ell$, satisfying $\big|\sum_{\ell=1}^{s} \widehat{\psi}_\ell - S(A)\big|/S(A) \lesssim \varepsilon$.

Similarly to the probing method, in order to have a final relative error bounded by ε, in our implementation we use an absolute tolerance $\widehat{\varepsilon} = \frac{1}{2}\varepsilon \cdot \mathrm{tr}(f(A))$ for adaptive Hutch++, and we set the accuracy for the computation of matrix-vector products and quadratic forms in order to ensure that the total error due to the Krylov approximations remains below $\frac{1}{2}\varepsilon \cdot \mathrm{tr}(f(A))$.

The algorithm adaptively determines the number of vectors used for the low-rank approximation and for the Hutchinson trace estimation of the remainder, using matrix-vector products and quadratic forms with $f(A)$. We mention that in the cases where the basic Hutch++ trace estimator (Algorithm 7.1) only requires a quadratic form $\boldsymbol{x}^T f(A)\boldsymbol{x}$, the adaptive version developed in [129] also needs to compute $\|f(A)\boldsymbol{x}\|_2$, which is used in its stopping criterion: however, it is enough to have a rough estimate of this norm, so we can set a target accuracy for the computation of $\boldsymbol{x}^T f(A)\boldsymbol{x}$ and obtain a lower accuracy approximation of $f(A)\boldsymbol{x}$ at the same time, without any further effort.

With reference to the notation used in Algorithm 7.1, for the low-rank part we have to approximate the r quadratic forms in

$$\mathrm{tr}(\boldsymbol{Q}^T f(A)\boldsymbol{Q}) = \sum_{j=1}^{r} \boldsymbol{q}_j^T f(A)\boldsymbol{q}_j$$

with a total error that is of order $\widehat{\varepsilon}/2$. However, the value of r is unknown at the beginning of the adaptive algorithm, so we cannot simply require the error for each quadratic form to be smaller than $\widehat{\varepsilon}/2r$. Instead, in order to have the inequality

$$\Big|\,\mathrm{tr}(\boldsymbol{Q}^T f(A)\boldsymbol{Q}) - \sum_{j=1}^{r} \widehat{\psi}_j\Big| \le \frac{\widehat{\varepsilon}}{2} \qquad \text{for any } r > 0,$$

where $\widehat{\psi}_j$ denotes the approximation to $\boldsymbol{q}_j^T f(A)\boldsymbol{q}_j$, we can either require the error on each quadratic form to be smaller than $\widehat{\varepsilon}/2n$, or consider a positive sequence $\{a_j\}_{j\ge1}$, such that $\sum_{j=1}^{\infty} a_j = T < +\infty$, and impose the condition

$$\Big|\boldsymbol{q}_j^T f(A)\boldsymbol{q}_j - \widehat{\psi}_j\Big| \le \frac{\widehat{\varepsilon}}{2T} a_j,$$

so that

$$\Big|\,\mathrm{tr}(\boldsymbol{Q}^T f(A)\boldsymbol{Q}) - \sum_{j=1}^{r} \widehat{\psi}_j\Big| \le \frac{\varepsilon}{2T}\sum_{j=1}^{r} a_j \le \frac{\widehat{\varepsilon}}{2} \qquad \text{for any } r > 0.$$

With a decreasing sequence such as $a_j = j^{-\alpha}$, with $\alpha > 1$, this second approach requires less computational effort if the value of p selected by the adaptive algorithm turns out to be small. In our implementation we use the sequence $a_j = \lceil j/100\rceil^{-\alpha}$, with $\alpha = 1.1$.

Then, the Hutchinson estimator approximates the trace of the remainder $\mathrm{tr}(A_{\mathrm{rest}})$ with

$$\mathrm{tr}_s^{\mathrm{H}}(A_{\mathrm{rest}}) = \frac{1}{s}\sum_{j=1}^{s} \boldsymbol{y}_j^T f(A)\boldsymbol{y}_j, \quad \text{with} \quad \boldsymbol{y}_j = (I - \boldsymbol{Q}\boldsymbol{Q}^T)\boldsymbol{x}_j, \quad \boldsymbol{x}_j \sim N(\boldsymbol{0}, I),$$

so we can simply approximate each quadratic form $\boldsymbol{y}_j^T f(A)\boldsymbol{y}_j$ with an absolute accuracy $\widehat{\varepsilon}/2$ and obtain an approximation to $\mathrm{tr}_q^{\mathrm{Hutch}}(A_{\mathrm{rest}})$ with absolute accuracy $\widehat{\varepsilon}/2$. Thus, the total error from approximating quadratic forms is bounded by $\widehat{\varepsilon}$, and if we set the input tolerance of adaptive Hutch++ to $\widehat{\varepsilon} = \frac{1}{2}\varepsilon \cdot \mathrm{tr}(f(A))$ we get an overall relative error for $\mathrm{tr}(f(A))$ bounded by ε. In practice, we can use a previously computed rough approximation of the entropy in place of $\mathrm{tr}(f(A))$, as discussed above.

Remark 8.3.4. The vectors $\boldsymbol{q}_j$ used in the low-rank approximation step are obtained by computing $\boldsymbol{w}_j = f(A)\boldsymbol{\omega}_j$, with $\boldsymbol{\omega}_j \sim N(\boldsymbol{0}, I)$, and orthogonalizing the resulting vectors. Analyzing the effect of using approximations to $\boldsymbol{q}_j$ can be complicated and may lead to pessimistic bounds, since we then have to approximate the quadratic forms $\boldsymbol{q}_j^T f(A)\boldsymbol{q}_j$. However, the vectors $\boldsymbol{q}_j$ are a basis of a subspace that is close to the one spanned by the dominant eigenvectors of $f(A)$, and it is reasonable to expect that this property remains approximately true even if we use approximations of the vectors $\boldsymbol{q}_j$. Therefore, in our implementation we use an absolute accuracy of $\widehat{\varepsilon}$ in the computation of $\boldsymbol{w}_j = f(A)\boldsymbol{\omega}_j$, but we observed that the overall accuracy for $\mathrm{tr}(f(A))$ remains the same even if the accuracy for $\boldsymbol{w}_j$ is reduced.

8.3.3 Krylov method implementation

As a stopping criterion for the computation of $\boldsymbol{b}^T f(A)\boldsymbol{b}$ with Krylov subspace methods, we use either the a posteriori upper bound from Theorem 3.2.3 or the estimate (8.2.4). For the computation of $f(A)\boldsymbol{b}$, we use as stopping criterion the upper bound from Theorem 3.2.2, or an estimate that is the equivalent of (8.2.4) in the matrix-vector product case. For the computation of the a posteriori bounds we use the pole swapping technique with an auxiliary pole at ∞ described in Section 3.2.2.

The number of poles at ∞ is chosen in an adaptive way, switching to finite poles when the error reduction in the last few iterations of the polynomial Krylov method is "small". Specifically, we decide to switch to EDS poles after the k-th iteration if on average the last $\ell \geq 1$ iterations did not reduce the error bound or estimate `err_est` by at least a factor $c \in (0, 1)$, i.e. if

$$\frac{\texttt{err\_est}_k}{\texttt{err\_est}_{k-\ell}} \geq c^\ell.$$

In our implementation we use $\ell = 3$ and $c = 0.75$, usually leading to at most 10 polynomial Krylov iterations.

Since EDS poles are contained in $(-\infty, 0)$, each rational Krylov iteration involves the solution of a symmetric positive definite linear system, which can be computed either with a direct method using a sparse Cholesky factorization, or iteratively with the conjugate gradient method using a suitable preconditioner. Note that the same EDS poles can be used for all quadratic forms, so the number of different matrices that appear in the linear systems is usually small and independent of the total number of quadratic forms. Although this depends on the accuracy requested for the entropy, the number of EDS poles used is almost always bounded by 10, and often much smaller than that: see the numerical experiments in Section 8.4 for some examples. This is a great advantage for direct methods, especially when the Cholesky factor remains sparse, since we can compute and store a Cholesky factorization for each pole and then reuse it for all quadratic forms. If the fill-in in the Cholesky factor is moderate, the cost of a rational Krylov iteration can become comparable to the cost of a polynomial one, leading to large savings when computing many quadratic forms. Of course, for large matrices with a general sparsity structure the computation of even a single Cholesky factor may be unfeasible, so the only option is to use a preconditioned iterative method. In such a situation, it is still possible to benefit from the small number of different matrices that appear in linear systems by storing and reusing preconditioners, but the gain is less evident compared to direct methods.

8.4 Numerical experiments

The experiments in this section were done in MATLAB R2021b on a laptop with operating system Ubuntu 20.04, using a single core of an Intel i5-10300H CPU running at 2.5 GHz, with 32 GB of RAM. Since we are using MATLAB, the execution times may not reflect the performance of a high performance implementation, but they are still a useful indicator when comparing different methods.

Table 8.1 – Information on the matrices used in the experiments.

test matrix	n	$\texttt{nnz}(A)$	`fill-in`	λ_2	λ_n	entropy
`yeast`	2224	15442	3.6	4.54e-06	4.96e-03	7.055
`minnesota`	2640	9244	1.3	1.28e-07	1.04e-03	7.607
`ca-HepTh`	8638	58250	7.5	4.92e-07	1.33e-03	8.540
`bcsstk29`	13830	618678	2.9	7.22e-08	1.25e-04	9.440
`cond-mat-2005`	36458	379926	21.7	5.63e-08	8.13e-04	9.958
`loc-Brightkite`	56739	482629	32.6	7.10e-08	2.67e-03	9.896
`ut2010`	115406	687472	1.2	2.72e-10	3.44e-04	11.361
`usroads`	126146	450046	1.4	2.39e-11	2.54e-05	11.478
`com-Amazon`	334863	2186607	105.9	6.69e-10	2.97e-04	12.400
`ny2010`	350167	2059711	1.8	4.67e-12	3.63e-05	12.541
`roadNet-PA`	1087562	4170590	1.6	5.54e-13	3.37e-06	13.628

8.4.1 Test matrices

We consider a number of symmetric test matrices from the SuiteSparse Matrix Collection [51]. All matrices are treated as binary matrices, i.e., all edge weights are set to one. For each matrix, we extract the graph Laplacian associated to the largest connected component and we normalize it so that it has unit trace. We report some information on the resulting matrices in Table 8.1. For the four smallest matrices, the eigenvalues were computed via diagonalization, while for the larger matrices the eigenvalues λ_2 and λ_n were approximated using `eigs`. The cost of solving a linear system with a direct method is highly dependent on the fill-in in the Cholesky factorization; the column labelled `fill-in` in Table 8.1 contains the ratios $\texttt{nnz}(R)/\texttt{nnz}(A)$, where A is the test matrix and R is the Cholesky factor of any shifted matrix $A + \alpha I$, for $\alpha > 0$ ($\texttt{nnz}(M)$ denotes the number of nonzeros of a matrix M). All matrices have been ordered using the approximate minimum degree reordering option available in MATLAB before factorization.

8.4.2 Probing bound vs. estimate

In this experiment, we fix a relative error tolerance $\varepsilon = 10^{-3}$ and we compare the choice of d given by the theoretical bound (8.2.1) with the one provided by the heuristic estimate (8.3.1). We report in Table 8.2 the error, the execution time, the value of d and the number of colors used in the two cases. When the theoretical bound is used, the selected value of d is significantly higher compared to the one chosen by the heuristic estimate, but in both cases the overall error remains below the tolerance ε. Moreover, for certain graphs using the theoretical bound leads to greedy colorings with a number of colors equal to the number of nodes in the graph, completely negating the advantage of using a probing method. Observe that the errors obtained with the larger value of d are not much smaller than the ones for the smaller value of d; this is because in both cases the quadratic forms are computed with target relative accuracy ε, so the probing error for the larger value of d is dominated by the error in the quadratic forms. In the case of the heuristic estimate, the execution time includes the time required to run the probing method for $d = 1, 2, 3$ in order to evaluate (8.3.1). The stopping criterion for the Krylov subspace method uses the upper bound from Theorem 3.2.3. The execution time can be further reduced by using the estimate (8.2.4) for the Krylov subspace method, as the following experiment shows. The diagonalization time for the matrices used in this experiment can be found in the last column of Table 8.4. Note that for smaller matrices, diagonalization is often the fastest method, but the advantage of approximating the entropy with a probing method is already evident for matrices of size $n \approx 10000$.

Table 8.2 – Comparison of the theoretical bound (8.2.1) against the heuristic estimate (8.3.1) for choosing d, using relative tolerance $\varepsilon = 10^{-3}$ in the probing method. Top row: heuristic estimate. Bottom row: theoretical bound.

test matrix	n	error	d	colors	time (s)
`yeast`	2224	3.062e-04	3	222	0.952
		3.733e-05	25	2224	4.464
`minnesota`	2640	4.456e-04	5	24	0.196
		3.173e-05	18	255	0.779
`ca-HepTh`	8638	2.974e-04	3	252	2.273
		3.161e-05	27	8638	42.078
`bcsstk29`	13830	4.912e-05	3	176	2.292
		8.497e-05	12	2095	17.849

Table 8.3 – Comparison of the upper bound from Theorem 3.2.3 against the geometric mean estimate (8.2.4) for Krylov methods used in the probing method, using relative tolerance $\varepsilon = 10^{-5}$ for the probing method. Top row: geometric mean estimate. Bottom row: upper bound.

test matrix	n	error	poly iter	rat iter	time (s)
`yeast`	2224	4.405e-06	16140	748	7.095
		6.023e-06	16419	2237	8.231
`minnesota`	2640	5.728e-07	2983	289	1.535
		2.490e-06	2987	598	1.734
`ca-HepTh`	8638	8.195e-07	39481	2442	40.240
		2.395e-06	39747	6389	50.133
`bcsstk29`	13830	6.133e-06	4704	0	12.093
		7.545e-06	6363	44	16.047

8.4.3 Krylov bound vs. estimate

We fix an error tolerance $\varepsilon = 10^{-5}$ and compare the performance of the geometric mean error estimate (8.2.4) with the theoretical upper bound from Theorem 3.2.3 for the Krylov subspace method. The value of d for the probing method is selected using the heuristic estimate (8.3.1). The entropy error, execution time, and total number of polynomial and rational Krylov iterations are reported in Table 8.3. We can see that using the estimate instead of the theoretical bound moderately reduces the computational effort, while still attaining the requested accuracy ε on the entropy. In particular, observe that the number of rational Krylov iterations is significantly higher when using the upper bound from Theorem 3.2.3. In this experiment, all linear systems are solved with direct methods and Cholesky factorizations are stored and reused.

8.4.4 Adaptive Hutch++

Here we test the performance and the accuracy of the adaptive implementation of Hutch++ [129, Algorithm 3]. A relative accuracy ε is achieved by setting the absolute tolerance to $\varepsilon S(A)$, where $S(A)$ is computed via diagonalization and considered exact. The computational effort of Hutch++ is determined by the number of sample vectors r used for low-rank approximation, and the number of sample vectors s used for Hutchinson's estimator, as described in Section 7.1. In particular, the number of matrix-vector products is equal to r and the number of quadratic forms is equal to $r + s$. We used $\varepsilon = 10^{-2}, 10^{-3}$ as target tolerances and $\delta = 10^{-2}$ as failure probability. Matrix-vector products and quadratic forms are computed using the Krylov subspace method with the geometric mean estimate as a stopping criterion (8.2.4). In Table 8.4 we compare the results for the two tolerances, obtained as an average of 100 runs of the algorithm, including both the average and worst relative error. In the majority of cases, the worst error is below the input tolerance ε. We see that for $\varepsilon = 10^{-2}$ the

Table 8.4 – Results for Hutch++ applied to some test matrices. For each matrix, the first and second row show the results for $\varepsilon = 10^{-2}$ and $\varepsilon = 10^{-3}$, respectively. The failure probability is $\delta = 10^{-2}$ in both cases. The last column contains the diagonalization times.

test matrix	avg error	worst error	r	$r+s$	time (s)	`eig` (s)
`yeast`	2.55e-03	9.94e-03	3	282	0.421	0.508
	3.56e-04	1.16e-03	1228	2160	19.735	
`minnesota`	3.46e-03	1.07e-02	3	154	0.111	0.819
	4.53e-04	1.26e-03	1854	2684	35.721	
`ca-HepTh`	2.83e-03	9.92e-03	3	81	0.205	22.170
	3.55e-04	9.14e-04	635	3968	66.350	
`bcsstk29`	1.84e-03	6.97e-03	3	38	0.171	86.696
	2.39e-04	8.24e-04	3	1883	13.276	

computation with Hutch++ is very fast for all test matrices; on the other hand, the cost becomes significantly higher for $\varepsilon = 10^{-3}$, showing that the stochastic estimator quickly becomes inefficient as the required accuracy increases. Observe that for $\varepsilon = 10^{-2}$ adaptive Hutch++ uses only 3 matvecs for all tests problems, which is the minimum amount that can be used by the implementation in [129]. This means that the internal criteria of the algorithm have determined that spending more matvecs in the low-rank approximation is not beneficial, and hence the convergence of the method is roughly the same as for Hutchinson's estimator. A similar behavior can be observed in Table 8.7, and can be linked to the fact that for these test matrices A, the matrix function $-A \log A$ does not exhibit eigenvalue decay and hence cannot be approximated well by low-rank matrices. On the other hand, for problems where low-rank approximation is more effective, stochastic estimators that exploit it such as Hutch++ can have much faster convergence.

8.4.5 Larger matrices

In this section we test the probing method and the adaptive Hutch++ algorithm on larger matrices, for which it would be extremely expensive to compute the exact entropy. In light of the results shown in Tables 8.2 and 8.3, we select the value of d for the probing method using the heuristic estimate (8.3.1) and we use the geometric mean estimate (8.2.4) for the Krylov subspace method. The results are reported in Tables 8.5 and 8.6 for the probing method, and in Table 8.7 for Hutch++. Figure 8.5 contains a more detailed breakdown of the execution time for the probing method used on the matrices of Table 8.5. We separate the time in preprocessing, where we evaluate the heuristic (8.3.1) to select d, and the main run of the algorithm with the chosen value of d. The time for the main run is further divided in coloring, and polynomial and rational Krylov iterations. The time for the Cholesky factorizations refers to the whole process, since the factors are computed and stored when a certain pole for a rational Krylov iteration is encountered for the first time.

In Table 8.5, we consider matrices with a "large-world" sparsity structure, such as road networks, and we use a relative tolerance of $\varepsilon = 10^{-4}$. For these matrices, for which the diameter and the average path length are relatively large, it is possible to compute distance-d colorings with a relatively small number of colors, and therefore probing methods converge quickly. Moreover, Cholesky factorizations can be computed cheaply and have a small fill-in, so it is possible to rapidly solve linear systems using a direct method. On the other hand, in Table 8.6 we consider matrices with a "small-world" sparsity structure, more typical of social networks and scientific collaboration networks. These matrices require a much larger number of colors to construct distance-d colorings, even for small values of d. The cost of probing methods is thus significantly higher on this kind of problem. In Table 8.6, only polynomial Krylov iterations are used due to the low relative tolerance $\varepsilon = 10^{-2}$, so it is never necessary to solve linear systems. Recall that these matrices also have a high fill-in in the Cholesky factorizations (see Table 8.1), so the conjugate gradient method with a suitable preconditioner is likely to be much more efficient than a direct method for solving a linear system.

In Table 8.7 we show the results for the adaptive Hutch++ algorithm, using relative tolerance

Table 8.5 – Results for the probing method applied to test matrices with large-world sparsity structure, using relative tolerance $\varepsilon = 10^{-4}$.

test matrix	n	d	colors	poly iter	rat iter	time (s)
`ut2010`	115406	4	504	7070	919	79.60
`usroads`	126146	8	77	626	0	6.30
`ny2010`	350167	5	329	3914	15	111.39
`roadNet-PA`	1087562	8	106	827	0	84.46

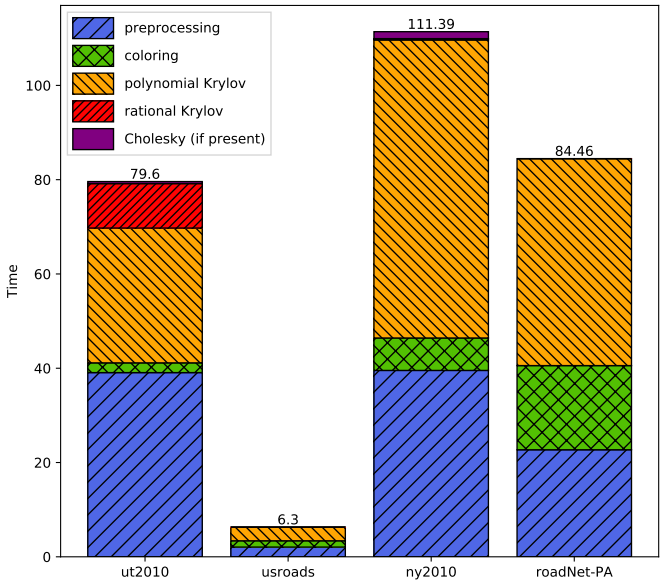


Figure 8.5 – Breakdown of the execution time of the probing method for the test matrices in Table 8.5.

Table 8.6 – Results for the probing method applied to test matrices with small-world sparsity structure, using relative tolerance $\varepsilon = 10^{-2}$.

test matrix	n	d	colors	poly iter	time (s)
`cond-mat-2005`	36458	3	1221	3883	13.809
`loc-Brightkite`	56739	3	3946	18765	90.564
`com-Amazon`	334863	3	625	1285	47.458

Table 8.7 – Results for Hutch++ applied to large test matrices, with relative tolerance $\varepsilon = 10^{-2}$ and failure probability $\delta = 10^{-2}$. The parameters N_r and N_H are defined in Section 7.1.

test matrix	n	N_r	$N_r + N_H$	time (s)
`ny2010`	350167	3	10	0.490
`usroads`	126146	3	14	0.190
`ny2010`	350167	3	10	0.487
`roadNet-PA`	1087562	3	8	1.126
`cond-mat-2005`	36458	3	34	0.448
`loc-Brightkite`	56739	3	42	4.576
`com-Amazon`	334863	3	11	1.171

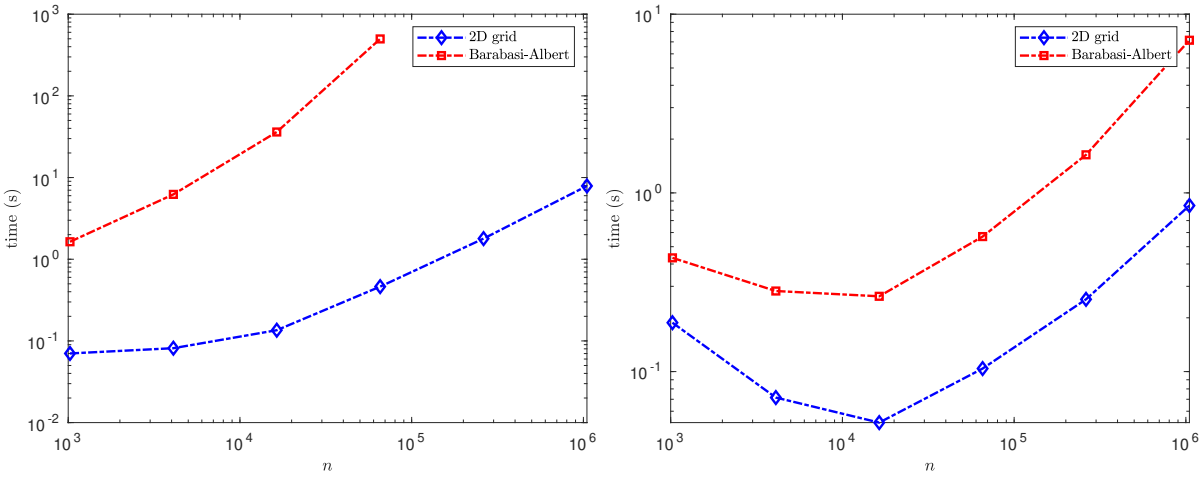


Figure 8.6 – Execution times for the probing method (left, $\varepsilon = 10^{-4}$) and the adaptive Hutch++ algorithm (right, $\varepsilon = 10^{-2}$) on the graph Laplacian of a 2D regular grid and a Barabasi-Albert random graph, as a function of the number of nodes n.

$\varepsilon = 10^{-2}$. The results are obtained as an average of 100 runs of the algorithm. We can observe that the stochastic trace estimator works well for both large-world and small-world graphs, in contrast to the probing method.

8.4.6 Algorithm scaling

To investigate how the complexity of the algorithms scales with the matrix size, we compare the scaling of the probing method and the stochastic trace estimator on two different test problems with increasing dimension. The first one is the graph Laplacian of a 2D regular square grid, and the second one is the graph Laplacian of a Barabasi-Albert random graph, generated using the `pref` function of the CONTEST MATLAB package [146]. For the probing method on the 2D grid, we use the optimal distance-d coloring with $\lceil \frac{1}{2}(d+1)^2 \rceil$ colors described in [70]. For both test problems, we use a relative tolerance $\varepsilon = 10^{-4}$ for the probing method, and a relative tolerance $\varepsilon = 10^{-2}$ and failure probability $\delta = 10^{-2}$ for adaptive Hutch++, averaging over 100 runs. The results are summarized in Figure 8.6, for graphs with a number of nodes from $n = 2^{10}$ to $n = 2^{20}$. As expected, the probing method is much more efficient in the case of the 2D grid, since the number of colors used in the distance-d colorings remains constant as n increases. On the other hand, for the Barabasi-Albert random graph, which has a small-world structure, the number of colors used in a distance-d coloring increases with the number of nodes, and hence the scaling for the probing method is significantly worse. The adaptive Hutch++ algorithm also has a better performance for the 2D grid, but the scaling in the problem size is good for

both graph categories, since the number of vectors used in the trace approximation does not increase with the matrix dimension. However, the stochastic approach is only viable with a loose tolerance ε for this kind of problem since low-rank approximation is not particularly effective, as discussed in Section 8.4.4. The initial decrease in the execution time for Hutch++ as n increases is caused by the fact that the adaptive algorithm uses a larger number of vectors for the graphs with fewer nodes.

Chapter 9

Estimation of gaps in the spectrum

In this chapter we consider the problem of locating gaps in the spectrum of an $n \times n$ real symmetric matrix A, i.e., finding intervals that do not contain any eigenvalue of A. A variant of this problem that is more commonly encountered in the literature is the following: given an integer k with $1 \leq k < n$, find $\mu \in \mathbb{R}$ such that exactly k of the eigenvalues of A (counted with their multiplicities) are strictly below μ. This problem can be equivalently restated as finding μ such that the spectral projector P_μ onto the subspace spanned by the eigenvectors of A associated with eigenvalues strictly smaller than μ satisfies $\mathrm{rank}(P_\mu) = k$; since for a projector the rank is equal to the trace, we can also write this requirement as $\mathrm{tr}(P_\mu) = k$. If we denote the eigenvalues of A by λ_i and we label them in non-decreasing order, this problem has a solution only if $\lambda_k < \lambda_{k+1}$ (non-degeneracy assumption): we will always assume this condition to be satisfied. In this case, any μ lying in the open interval $(\lambda_k, \lambda_{k+1})$ is a solution. Instead of looking for the gap for a specific value of k, the approach that we present here aims to find all gaps in the spectrum with width above a certain threshold, and in addition we obtain a rough estimate of the number of eigenvalues below each gap. In practical cases where it is of interest to find μ in the gap between λ_k and λ_{k+1}, the gap $(\lambda_k, \lambda_{k+1})$ is usually relatively large, so we expect to be able to find it by looking for all gaps above a certain width.

A major application area where this kind of problem is encountered is electronic structure computations. For example, in Kohn-Sham Density Functional Theory [108] it is required to determine either the eigenvectors of a symmetric matrix A (the discretized Hamiltonian) associated with eigenvalues below the *Fermi level* (or *chemical potential*, usually denoted by μ), corresponding to the so-called *occupied states* of the system, or equivalently the corresponding spectral projector P_μ. In the case of insulators at zero electronic temperature, the discrete Hamiltonian A exhibits a gap separating the first k eigenvalues from the rest of the spectrum, where k is the number of electrons, and locating this gap is equivalent to estimating the Fermi level μ. In linear scaling electronic structure computations the spectral projector is approximated either by polynomials or by rational functions that act as filters, i.e., approximate the step function $h_\mu(\lambda)$ that takes the value 1 for $\lambda < \mu$ and 0 for $\lambda > \mu$ (see, e.g., [14]). Clearly, in order to use this approach it is first necessary to estimate the Fermi level μ, which is a nontrivial task.

In the literature, a few approaches have been proposed to estimate values of μ lying within the target gap. It is well known that for any real symmetric matrix $A - \mu I$ there exists a unit lower triangular matrix L and a block diagonal matrix D with 1×1 and 2×2 blocks such that $A - \mu I = LDL^T$; see for example [29, Chapter 1.3.4]. Since A and D are congruent, they have the same inertia, i.e., the same number of positive, negative, and possibly zero eigenvalues (the latter can occur only when μ happens to be an eigenvalue of A, an extremely unlikely occurrence in practice). Hence, if A is not too large, an LDL^T factorization of $A - \mu I$ can be used to count the number of eigenvalues smaller than μ. If this number is less than k (larger than k) then μ is increased (respectively, decreased) and the procedure is repeated until the correct value is found. When A is sparse, one can make use of symmetric row and column permutations to reduce the fill-in in L (and therefore the arithmetic and storage costs) while preserving sparsity, see [139, Chapter 7]. A major drawback of this approach is its cost, and especially the fact that having computed the LDL^T factorization of $A - \mu I$ for a certain μ is

of no help in computing the factorization for a different value of μ, hence using matrix factorizations in a trial-and-error fashion can be prohibitively expensive, even assuming that a factorization can be computed at all.

Different techniques have been used in the context of linear scaling methods. In these methods, an initial guess $\mu_0 \approx \mu$ is used in constructing a polynomial approximation to the projector P_{μ_0} by an iterative process known as *purification*, essentially a Newton-type method. The value of μ is adjusted in the course of the iterative process until the prescribed value k of the trace is obtained; hence, the determination of μ and the approximation of the projector P_μ are interweaved. The main cost in this approach is the need to repeatedly solve $n \times n$ linear systems with variable coefficient matrix at each iteration. We refer to [123, 124] for details on these adaptive procedures.

Yet another approach is the one that combines stochastic trace estimators with the Lanczos algorithm to compute an approximation of $\operatorname{tr}(P_\mu)$. An advantage of this approach is that we can exploit the shift-invariance property of Krylov subspaces (see, e.g., [73]) to efficiently approximate the trace of the spectral projectors P_μ associated with many different values of μ at the same time. The combination of Hutchinson's estimator and the Lanczos algorithm has also appeared in literature on related problems that exploit the trace of spectral projectors, such as the estimation of the number of eigenvalues of a matrix A contained in a given interval $[a, b]$ or the estimation of spectral densities, see for instance [55, 109]. In the context of spectral density approximation, a similar approach is given by the kernel polynomial method, which was introduced in [140, 154] and uses a polynomial expansion based on Chebyshev polynomials.

Here we focus on the approach that combines Hutchinson's trace estimator and the Lanczos algorithm to estimate the trace of the spectral projectors P_μ. When using this kind of approach, it is essential to have a criterion to choose the parameters of the algorithm, namely the number of random vectors $\boldsymbol{x}_i$ for Hutchinson's estimator and the number of Lanczos iterations, in order to minimize the computational cost and guarantee the correctness of the results. However, although there exist theoretical results on the convergence speed of this approach, past literature did not give particular attention to establishing how the parameters should be selected. The main purpose of this chapter is to answer this question for the problem of finding gaps in the spectrum of a matrix. By changing the point of view from finding μ such that the number of eigenvalues below μ is exactly k to finding all the gaps in the spectrum with width above a certain threshold, we are able to provide a rigorous analysis that allows us to choose the input parameters to ensure that all the target gaps are found, and at the same time minimize the computational cost. Our analysis will determine that for this problem the most efficient choice is to use a single quadratic form $\boldsymbol{x}^T P_\mu \boldsymbol{x}$ to approximate $\operatorname{tr}(P_\mu)$, but in order to reach this conclusion we first have to analyze the more general algorithm that uses Hutchinson's stochastic trace estimator.

9.1 Algorithm overview

We start by introducing some notation to establish the framework for our algorithm. Given $\mu \in \mathbb{R}$, let $h_\mu : \mathbb{R} \to \mathbb{R}$ denote the Heaviside function

$$h_\mu(x) = \begin{cases} 1 & \text{for } x < \mu, \\ 1/2 & \text{for } x = \mu, \\ 0 & \text{for } x > \mu. \end{cases}$$

The step function h_μ is closely related to the sign function, indeed we have $h_\mu(x) = -\frac{1}{2}\operatorname{sign}(x-\mu) + \frac{1}{2}$. We denote by λ_i the eigenvalues of A in non-decreasing order, and by $\boldsymbol{u}_i \in \mathbb{R}^n$ a normalized eigenvector associated to λ_i. Assuming that μ does not coincide with any of the eigenvalues of the matrix A, we can define the spectral projector

$$P_\mu := h_\mu(A) = \sum_{i\,:\,\lambda_i < \mu} \boldsymbol{u}_i \boldsymbol{u}_i^T.$$

In the following, we are always going to assume that $\mu \neq \lambda_i$ for all $i = 1, \dots, n$. Since the eigenvalues of P_μ are all either 0 or 1, we have

$$\operatorname{tr}(P_\mu) = \operatorname{rank}(P_\mu) = \#\{i \,:\, \lambda_i < \mu\} =: \mathrm{n_e}(\mu),$$

where $\mathrm{n_e}(\mu)$ denotes the number of eigenvalues of A that are strictly smaller than μ. The problem of finding gaps in the spectrum of A can be formulated as finding intervals $[a_j, b_j]$ such that $\mathrm{tr}(P_\mu)$ is constant for $\mu \in [a_j, b_j]$. We are going to determine gaps in the spectrum of A by approximating $\mathrm{tr}(P_\mu)$ for several different values of μ, using Hutchinson's trace estimator combined with the Lanczos method for the approximation of quadratic forms. We first focus on the approximation of $\mathrm{tr}(P_\mu)$ for a fixed value of μ.

Given s random vectors $\{\boldsymbol{x}_i\}_{i=1}^s \subset \mathbb{R}^n$ with i.i.d. unit normal entries $\mathcal{N}(0,1)$, using Hutchinson's estimator we can approximate $\mathrm{tr}(P_\mu)$ with

$$\mathrm{tr}_s^{\mathrm{H}}(P_\mu) = \frac{1}{s}\sum_{i=1}^{s} \boldsymbol{x}_i^T P_\mu \boldsymbol{x}_i. \tag{9.1.1}$$

Recall that in order to attain an absolute accuracy ε on $\mathrm{tr}(P_\mu)$, Hutchinson's trace estimator requires $O(\|P_\mu\|_F^2/\varepsilon^2)$ sample vectors (see Proposition 7.1.1). The $O(1/\varepsilon^2)$ dependence on ε makes it very expensive to obtain highly accurate approximations, but it is not an issue for a setting such as ours, where an absolute accuracy $\varepsilon = \frac{1}{2}$ is enough to exactly determine $\mathrm{tr}(P_\mu)$, which is an integer. In contrast, the dependence on $\|P_\mu\|_F^2$ will make it quite difficult to accurately estimate the number of eigenvalues below μ, especially when μ is near the center of the spectrum, since $\|P_\mu\|_F^2 = \mathrm{n_e}(\mu)$ can be quite large. To overcome this issue, we will have to slightly change our perspective in order to develop an efficient algorithm. This topic will be discussed in greater detail in Section 9.2.2.

Remark 9.1.1. Since $\mathrm{rank}(P_\mu) = \mathrm{n_e}(\mu)$ and all the eigenvalues of P_μ are either 0 or 1, for this setting it usually does not make much sense to use the improved stochastic trace estimators such as Hutch++ which employ a low-rank matrix approximation. Indeed, there is no decay in the eigenvalues of P_μ and hence there is no advantage in computing a low-rank approximation of P_μ, unless $\mathrm{n_e}(\mu)$ is very small compared to the matrix size.

Remark 9.1.2. When the matrix A is sparse, we could alternatively approximate the trace of the matrix function $P_\mu = h_\mu(A)$ with a probing method (see Section 7.2). Recall that by Theorem 7.2.1 the probing approach has a faster convergence when the function $h_\mu(A)$ is well-approximated by polynomials on the spectrum of A, i.e., it converges similarly to the Lanczos method for the computation of $h_\mu(A)\boldsymbol{x}$. We will see in Section 9.2.3 that this convergence rate is heavily dependent on the value of μ, so we prefer to use a stochastic trace estimator in order to eliminate this dependence.

Within Hutchinson's estimator we have to compute the quadratic forms $\boldsymbol{x}_i^T P_\mu \boldsymbol{x}_i$ for $i = 1, \ldots, s$, which can be done efficiently by using the Lanczos method. Let us consider a single quadratic form $\boldsymbol{x}^T h_\mu(A)\boldsymbol{x}$. Denoting by V_{m+1} the orthonormal basis of $\mathcal{K}_{m+1}(A, \boldsymbol{x})$ constructed by the Lanczos algorithm, and by $\underline{T}_m \in \mathbb{R}^{(m+1)\times m}$ the tridiagonal matrix that satisfies the Arnoldi relation

$$AV_m = V_{m+1}\underline{T}_m,$$

we can approximate the quadratic form $\boldsymbol{x}^T P_\mu \boldsymbol{x}$ with

$$\psi_m := \|\boldsymbol{x}\|_2^2 \boldsymbol{e}_1^T h_\mu(T_m)\boldsymbol{e}_1, \tag{9.1.2}$$

recalling that $V_m^T \boldsymbol{x} = \|\boldsymbol{x}\|_2 \boldsymbol{e}_1$.

9.1.1 Simultaneous computation for different μ

The approach described above can be easily modified in order to compute approximations to $\mathrm{tr}(P_\mu)$ simultaneously for several μ, for a cost that is only slightly higher than the cost of the approximation for a single μ. The key observation is that for a fixed matrix A and vector $\boldsymbol{x}$, we can extract approximations to the quadratic forms $\boldsymbol{x}^T f_j(A)\boldsymbol{x}$ for several different functions f_j from a single Krylov subspace $\mathcal{K}_m(A, \boldsymbol{x})$. Since the cost of computing the approximation (9.1.2) is dominated by the cost of constructing the orthonormal basis V_m via the Lanczos algorithm, the additional operations required to compute approximations for different functions represent only a minor part of the overall cost.

More precisely, let us consider N_f different functions f_j, $j = 1, \ldots, N_f$. For a fixed Krylov subspace dimension m, we can approximate each quadratic form $\boldsymbol{x}^T f_j(A)\boldsymbol{x}$ with

$$\psi_{m,j} := \|\boldsymbol{x}\|_2^2 \boldsymbol{e}_1^T f_j(T_m)\boldsymbol{e}_1, \qquad j = 1, \ldots, N_f.$$

The quantities $\psi_{m,j}$ can be computed efficiently in the following way. Obtain first an eigenvalue decomposition $T_m = U_m^T D_m U_m$ and compute $f_j(D_m)$ for $j = 1, \ldots, N_f$ by applying the functions f_j to the diagonal entries of D_m. The Lanczos approximations to the quadratic forms can now be computed with the identity

$$\psi_{m,j} = \boldsymbol{w}_m^T f_j(D_m)\boldsymbol{w}_m, \qquad j = 1, \ldots, N_f,$$

where $\boldsymbol{w}_m := \|\boldsymbol{x}\|_2 U_m \boldsymbol{e}_1$ is the same for all j and needs to be computed only once. If we assume that the small matrix function $f(T_m)$ is computed via an eigenvalue decomposition also in the case of a single function f, the only additional operations that we have to do when we have N_f different functions are the computation of the diagonal matrix functions $f_j(D_m)$ and the computation of the final approximations $\psi_{m,j} = \boldsymbol{w}_m^T f_j(D_m)\boldsymbol{w}_m$ for all j, for a total cost of $O(mN_f)$. This additional cost is negligible compared to the $O(m\,\mathtt{nnz}(A))$ cost for the matrix-vector products with A in the construction of the Krylov subspace $\mathcal{K}_m(A, \boldsymbol{x})$, unless the number of functions N_f is extremely large.

In total, the cost of computing approximations to $\boldsymbol{x}^T f_j(A)\boldsymbol{x}$ for $j = 1, \ldots, N_f$ with the Lanczos algorithm adds up to $O(m\,\mathtt{nnz}(A) + nm + m^3 + mN_f)$, where the $O(nm)$ term is the cost of orthogonalization in the Lanczos algorithm using the three-term recurrence; if full reorthogonalization is used instead to increase stability, the orthogonalization cost would increase to $O(nm^2)$. The $O(m^3)$ term corresponds to the computation of eigenvalues and eigenvectors of the tridiagonal matrix T_m, and the $O(mN_f)$ term corresponds to the computation of $\psi_{m,j}$ for all j. Note that the convergence rate of the approximation of the quadratic form $\boldsymbol{x}^T f_j(A)\boldsymbol{x}$ can vary significantly depending on the function f_j, so this approach may construct approximations with considerably different errors for different functions, since we are using the same number of iterations m for all functions. We will focus on analyzing this aspect in Section 9.2.3.

Remark 9.1.3. A similar approach could be used to approximate quadratic forms with several different functions using the same rational Krylov subspace $\mathcal{Q}_m(A, \boldsymbol{x}, \boldsymbol{\xi}_m)$. However, the convergence of a rational Krylov method is heavily dependent on the poles $\boldsymbol{\xi}_m$, which should be chosen depending on the function. In particular, in the case of $f_j = h_{\mu_j}$ for different parameters μ_j, a sequence of poles that produces an accurate approximation for a certain μ_j will give poor approximations for other parameters, especially if they are far from μ_j. For this reason, the Lanczos method is more suited for this scenario. A rational Krylov subspace method could be useful to compute a more accurate approximation for the quadratic form $\boldsymbol{x}^T h_\mu(A)\boldsymbol{x}$ associated with a specific parameter μ, especially in situations in which the Lanczos method converges very slowly.

Returning now to the approximation of $\mathrm{tr}(P_\mu)$ with Hutchinson's estimator, let us consider N_f different parameters $\{\mu_j\}_{j=1}^{N_f} \subset \mathbb{R}$ and define the functions $f_j = h_{\mu_j}$. Provided that we use the same random sample vectors $\{\boldsymbol{x}_i\}_{i=1}^s$ for the approximation of $\mathrm{tr}(P_{\mu_j})$ via Hutchinson's estimator for all j, we can use the same Krylov subspace $\mathcal{K}_m(A, \boldsymbol{x}_i)$ to extract approximations to $\boldsymbol{x}_i^T P_{\mu_j}\boldsymbol{x}_i$ for all μ_j at the same time, with essentially the same cost of approximating a quadratic form for a single parameter μ. This feature of this approach represents a considerable computational advantage over other methods, such as those based on the LDL^T factorization, which are only able to test a single candidate μ at a time.

A high-level algorithmic description of the procedure outlined in this section is given in Algorithm 9.1. This algorithm returns an approximation to $\mathrm{tr}(P_{\mu_j})$ for all $j = 1, \ldots, N_f$. The level of accuracy of these approximations, as well as whether they can be exploited to detect gaps in the spectrum of A, are going to be thoroughly investigated in the sections that follow. We are also going to identify criteria for selecting the input parameters of the method, namely the number of Hutchinson sample vectors s and Lanczos iterations m. The product of this analysis will be an algorithm that, given as input a failure probability δ and a relative gap width θ, aims to find all the gaps in the spectrum that are larger than θ; each one of these gaps is found with probability at least $1 - \delta$. In addition, all the gaps found by the algorithm are certified, in the sense that the probability of finding an eigenvalue of A within a gap is smaller than δ.

Algorithm 9.1 Eigenvalue gap finder – Prototype

Input: Symmetric matrix $A \in \mathbb{R}^{n\times n}$, number of Hutchinson samples s, Krylov subspace dimension m, vector of test parameters $\boldsymbol{\mu} = [\mu_1, \dots, \mu_{N_f}]$
Output: $\mathtt{tr}_j \approx \operatorname{tr}(P_{\mu_j})$ for $j = 1, \dots, N_f$

for $i = 1, \dots, s$ **do**
  Draw a vector $\boldsymbol{x}_i \in \mathbb{R}^n$ with random i.i.d. Gaussian $\mathcal{N}(0,1)$ entries
  Construct an orthonormal basis $V_m \in \mathbb{R}^{n\times m}$ of the Krylov space $\mathcal{K}_m(A, \boldsymbol{x}_i)$ with the Lanczos algorithm and compute the associated tridiagonal matrix $\underline{T}_m \in \mathbb{R}^{(m+1)\times m}$
  Compute the eigenvalue decomposition $T_m = U_m^T D_m U_m$ and $\boldsymbol{w}_m = \|\boldsymbol{x}_i\|_2 U_m \boldsymbol{e}_1$
  for $j = 1, \dots, N_f$ **do**
    Compute $\psi_{m,j}^{(i)} = \boldsymbol{w}_m^T h_{\mu_j}(D_m)\boldsymbol{w}_m$
  end for
end for
for $j = 1, \dots, N_f$ **do**
  Compute the trace approximation $\mathtt{tr}_j = s^{-1}\sum_{i=1}^s \psi_{m,j}^{(i)}$
end for

9.2 Algorithm analysis

In this section we examine some general properties of the trace approximations computed by Algorithm 9.1, and we analyze in detail the errors that occur in the Hutchinson and Lanczos parts of the approximation.

9.2.1 General properties

A simple but essential observation is that the approximations to $\operatorname{tr}(P_\mu)$ computed by Algorithm 9.1 are increasing in μ. Let us start by looking at the Hutchinson approximation. Denoting by $(\lambda_j, \boldsymbol{u}_j)_{j=1}^n$ the normalized eigenpairs of A, so that $A = \sum_{j=1}^n \lambda_j \boldsymbol{u}_j \boldsymbol{u}_j^T$, we can write the trace approximation obtained with Hutchinson's estimator as

$$\operatorname{tr}_s^{\mathrm{H}}(P_\mu) = \frac{1}{s}\sum_{i=1}^s \boldsymbol{x}_i^T P_\mu \boldsymbol{x}_i = \frac{1}{s}\sum_{i=1}^s \boldsymbol{x}_i^T \Big(\sum_{j\,:\,\lambda_j<\mu} \boldsymbol{u}_j \boldsymbol{u}_j^T\Big)\boldsymbol{x}_i = \frac{1}{s}\sum_{i=1}^s \sum_{j\,:\,\lambda_j<\mu} |\boldsymbol{u}_j^T \boldsymbol{x}_i|^2.$$

This shows that the Hutchinson trace approximation is monotonic in μ. In addition, the above identity also highlights the fact that Hutchinson's approximation of $\operatorname{tr}(P_\mu)$ is a piecewise constant function, with jumps when μ coincides with an eigenvalue of A. The height of the jump at λ_k is given by $\frac{1}{s}\sum_{i=1}^s |\boldsymbol{u}_k^T \boldsymbol{x}_i|^2$.

Let us now consider the Lanczos approximation of the quadratic form $\boldsymbol{x}_i^T P_\mu \boldsymbol{x}_i$. Letting $T_m^{(i)}$ be the projection of A onto $\mathcal{K}_m(A, \boldsymbol{x}_i)$ computed by the Lanczos algorithm, and denoting by $(\lambda_j^{(i)}, \boldsymbol{u}_j^{(i)})_{j=1}^m$ the normalized eigenpairs of $T_m^{(i)}$, we have

$$\boldsymbol{x}_i^T P_\mu \boldsymbol{x}_i \approx \psi_{m,j}^{(i)} := \|\boldsymbol{x}_i\|_2^2 \boldsymbol{e}_1^T h_\mu(T_m^{(i)})\boldsymbol{e}_1 = \|\boldsymbol{x}_i\|_2^2 \sum_{j\,:\,\lambda_j^{(i)}<\mu} |\boldsymbol{e}_1^T \boldsymbol{u}_j^{(i)}|^2.$$

Again, this is an increasing function of μ, with jumps whenever μ coincides with an eigenvalue of $T_m^{(i)}$. It follows that the approximation of $\operatorname{tr}(P_\mu)$ computed by Algorithm 9.1 using Hutchinson's estimator and the Lanczos method is an increasing function of μ, with jumps when μ is equal to an eigenvalue of $T_m^{(i)}$ for some i.

Example 9.2.1. To illustrate the behavior of the approximation of $\operatorname{tr}(P_\mu)$ computed by Algorithm 9.1 on a simple example, we consider a symmetric matrix A of size $n = 600$, with eigenvalues contained in the interval $[0, 60]$ and three gaps given by the intervals $[20, 21]$, $[30, 32]$ and $[40, 44]$. In Figure 9.1 we compare the approximation from Algorithm 9.1 with $m = 50$ Lanczos iterations, Hutchinson's

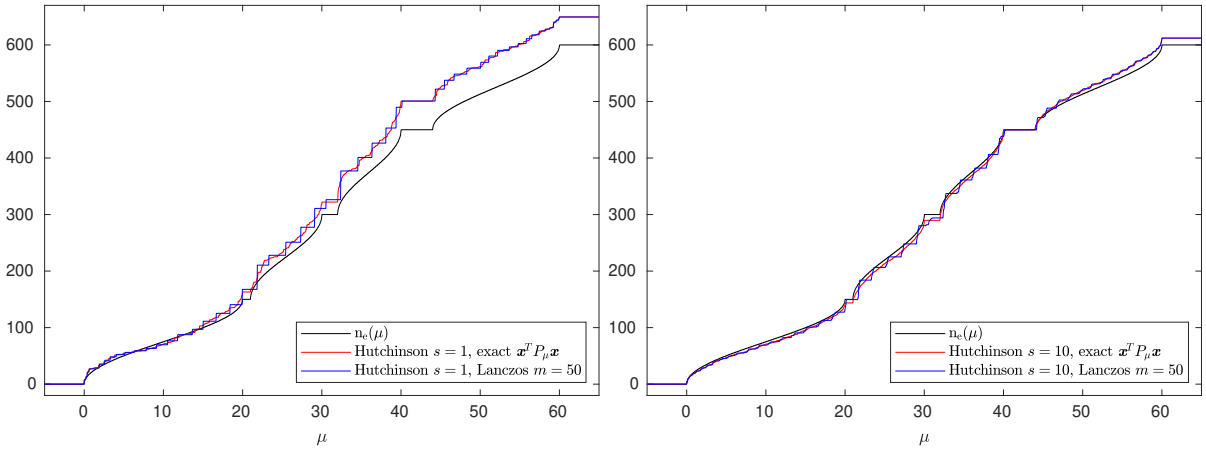


Figure 9.1 – Approximations to $\mathrm{tr}(P_\mu)$ obtained with Algorithm 9.1 compared with exact $\mathrm{tr}(P_\mu)$ and Hutchinson's approximation $\mathrm{tr}_s^{\mathrm{H}}(P_\mu)$ with exact quadratic forms, for several different μ. Left: $s = 1$, $m = 50$. Right: $s = 10$, $m = 50$.

estimator $\mathrm{tr}_s^{\mathrm{H}}(P_\mu)$ with exact quadratic forms, and the exact trace $\mathrm{tr}(P_\mu)$, which coincides with $\mathrm{n_e}(\mu)$, for several different values of μ. The number of random vectors is either $s = 1$ or $s = 10$. Figure 9.1 confirms the behavior described above. In particular, we see that Hutchinson's estimator with exact quadratic forms has a jump whenever μ coincides with an eigenvalue of A, so it correctly detects all the gaps in the spectrum, although the value of the approximation of $\mathrm{tr}(P_\mu)$ can be quite off, especially with $s = 1$. On the other hand, Hutchinson's estimator combined with the Lanczos algorithm for computing quadratic forms, which is the approximation that we are actually able to compute, has jumps when μ corresponds to an eigenvalue of one of the projected matrices $T_m^{(i)}$. We see that the Lanczos approximation is able to detect the largest gap quite well, but the other two gaps do not stand out from the other steps in the staircase-like curve. With $s = 10$, the Lanczos curve looks like a staircase with somewhat jagged steps, because the projected matrices $T_m^{(i)}$ have different eigenvalues for different random vectors $\boldsymbol{x}_i$, and the curve is obtained by averaging the approximations $\psi_{m,j}^{(i)}$ associated with each $T_m^{(i)}$.

9.2.2 Trace estimator analysis

In this section we focus on analyzing the approximation of $\mathrm{tr}(P_\mu)$ using Hutchinson's estimator. The error from the stochastic trace estimator can be bounded with Proposition 7.1.1. Since all the eigenvalues of P_μ are either 0 or 1, we have $\|P_\mu\|_2 = 1$ and $\|P_\mu\|_F^2 = \mathrm{n_e}(\mu)$. Therefore, in order to have $\mathbb{P}\big(|\mathrm{tr}(P_\mu) - \mathrm{tr}_p^H(P_\mu)| \geq \varepsilon\big) \leq \delta$ with Gaussian vectors, we would need to take

$$s \geq \frac{4}{\varepsilon^2}\big(\mathrm{n_e}(\mu) + \varepsilon\big)\log(2/\delta). \tag{9.2.1}$$

Since the exact $\mathrm{tr}(P_\mu)$ only takes integer values, in order to compute it correctly it is sufficient to have a final absolute accuracy smaller than $\frac{1}{2}$ on the trace approximation; if we also account for the error in the Lanczos approximation, it is enough to take $\varepsilon = \frac{1}{4}$. However, a key issue with this approach is the fact that the required number of sample vectors s grows proportionally to the number $\mathrm{n_e}(\mu)$ of eigenvalues below the level μ. This means that in a situation in which a gap is not close to the extrema of the spectrum, such as if $\mathrm{n_e}(\mu) = n/c$ for some moderate constant $c > 1$, we would have to take $s = O(n)$, which is too expensive to be practical.

Remark 9.2.1. In alternative to Gaussian random vectors, we can also use Rademacher vectors, which have random ± 1 entries. As we mentioned in Chapter 7, the convergence rate of Hutchinson's estimator with Rademacher random vectors depends on $\|B - \mathrm{diag}(B)\|_F^2$ instead of $\|B\|_F^2$, so it can be faster than

with Gaussian vectors in certain scenarios, such as when the entries of B decay away from the diagonal. Here we prefer to employ Gaussian vectors because of the rotational invariance of their distribution, which will prove useful in the following analysis.

In order to circumvent the growth of the number of sample vectors with $\mathrm{n_e}(\mu)$, we slightly change our point of view. Instead of considering the approximation of $\mathrm{tr}(P_\mu)$, let us take two different values $\mu < \mu'$, and consider the difference $\mathrm{tr}(P_\mu) - \mathrm{tr}(P_{\mu'})$. If we use the same sample vectors in Hutchinson's estimator for both $\mathrm{tr}(P_\mu)$ and $\mathrm{tr}(P_{\mu'})$, the difference $\mathrm{tr}_s^{\mathrm{H}}(P_\mu) - \mathrm{tr}_s^{\mathrm{H}}(P_{\mu'})$ coincides with the Hutchinson approximation of $\mathrm{tr}(P_\mu - P_{\mu'})$. Since we have $\|P_\mu - P_{\mu'}\|_F^2 = \mathrm{n_e}(\mu') - \mathrm{n_e}(\mu)$, this means that Hutchinson's estimator can easily approximate the differences in the traces of projectors when μ and μ' are close enough that there are only a few eigenvalues of A in the interval $[\mu, \mu']$, even if the exact values of $\mathrm{tr}(P_\mu)$ and $\mathrm{tr}(P_{\mu'})$ are quite far from the approximations computed by the stochastic estimator. In particular, the heights of the jumps in $\mathrm{tr}_s^{\mathrm{H}}(P_\mu)$ that occur when μ coincides with an eigenvalue of A should be approximated quite accurately even with a small number of sample vectors.

Let us analyze this phenomenon in more detail. Recall that

$$\mathrm{tr}_s^{\mathrm{H}}(P_\mu) = \frac{1}{s} \sum_{i=1}^{s} \sum_{j \,:\, \lambda_j < \mu} |\boldsymbol{u}_j^T \boldsymbol{x}_i|^2,$$

so if we take μ and μ' such that $\lambda_{k-1} < \mu < \lambda_k < \mu' < \lambda_{k+1}$ for some k, we have

$$\mathrm{tr}_s^{\mathrm{H}}(P_{\mu'}) - \mathrm{tr}_s^{\mathrm{H}}(P_\mu) = \frac{1}{s} \sum_{i=1}^{s} |\boldsymbol{u}_k^T \boldsymbol{x}_i|^2.$$

Since the random vectors $\boldsymbol{x}_i$ are independent of the eigenvector $\boldsymbol{u}_k$, we have that $\{\boldsymbol{u}_k^T \boldsymbol{x}_i\}_{i=1}^s$ are i.i.d. $\mathcal{N}(0,1)$ random variables, due to the rotational invariance of Gaussian vectors. As a consequence, the random variable $y_s = \sum_{i=1}^s |\boldsymbol{u}_k^T \boldsymbol{x}_i|^2$ has a χ^2 distribution with s degrees of freedom, i.e., it is the sum of the squares of s independent unit normal random variables.

We have $\mathbb{E}[y_s] = s$, which means that the expected height of each jump in $\mathrm{tr}_s^{\mathrm{H}}(P_\mu)$ is equal to 1. Intuitively, the height of a jump in $\mathrm{tr}_s^{\mathrm{H}}(P_\mu)$ is small when the vectors $\boldsymbol{x}_i$ are all approximately orthogonal to the eigenvector $\boldsymbol{u}_k$, an event that has a small chance of occurring. Since small jumps in $\mathrm{tr}_s^{\mathrm{H}}(P_\mu)$ are more difficult to detect when approximating the quadratic forms in Hutchinson's estimator via the Lanczos method, it will be useful to have an upper bound on the probability of small jumps. The lemma that follows provides us with such a bound. In the following, we say that the jump in $\mathrm{tr}_s^{\mathrm{H}}(P_\mu)$ at the k-th eigenvalue of A is *ε-small* if its height is smaller than ε, i.e., if $y_s \leq s\varepsilon$.

Lemma 9.2.2. Given $\delta \in (0,1)$ and $\varepsilon \in (0,1)$, if

$$s \geq \frac{2\log(1/\delta)}{\log(1/\varepsilon) + \varepsilon - 1},$$

then we have

$$\mathbb{P}(y_s \leq s\varepsilon) \leq \delta,$$

i.e., the jump at λ_k is ε-small with probability at most δ.

Proof. To prove this, we use a Chernoff bound on the lower tail of the probability distribution of y_s. For any $\lambda > 0$, we have

$$\mathbb{P}(y_s \leq s\varepsilon) = \mathbb{P}(e^{-\lambda y_s} \geq e^{-\lambda s\varepsilon}) \leq \frac{\mathbb{E}[e^{-\lambda y_s}]}{e^{-\lambda s\varepsilon}}, \tag{9.2.2}$$

where the last inequality follows from Markov's inequality. Since y_s is a χ^2 random variable with s degrees of freedom, we can write $y_s = X_1^2 + \cdots + X_s^2$, where $\{X_j\}_{j=1}^s$ are i.i.d. $\mathcal{N}(0,1)$ random variables. We have

$$\mathbb{E}[e^{-\lambda X_j^2}] = \frac{1}{\sqrt{2\pi}} \int_{\mathbb{R}} e^{-\lambda z^2} e^{-\frac{1}{2}z^2} \,\mathrm{d}z = \frac{1}{\sqrt{2\pi}} \int_{\mathbb{R}} \frac{1}{\sqrt{1+2\lambda}} e^{-\frac{1}{2}y^2} \,\mathrm{d}y = \frac{1}{\sqrt{1+2\lambda}},$$

where we used the change of variables $y = z\sqrt{1+2\lambda}$. It follows that

$$\mathbb{E}[e^{-\lambda y_s}] = \mathbb{E}\left[\prod_{j=1}^{s} e^{-\lambda X_j^2}\right] = \mathbb{E}[e^{-\lambda X_1^2}]^s = (1+2\lambda)^{-s/2},$$

and by combining this identity with (9.2.2) we obtain

$$\mathbb{P}(y_s \leq s\varepsilon) \leq e^{\lambda s\varepsilon}(1+2\lambda)^{-s/2} \qquad \forall \lambda > 0. \tag{9.2.3}$$

Taking $\lambda = \frac{1}{2\varepsilon} - \frac{1}{2}$, we get

$$\mathbb{P}(y_s \leq s\varepsilon) \leq \left(\varepsilon e^{1-\varepsilon}\right)^{s/2}.$$

With some simple algebraic manipulations, we conclude that $\mathbb{P}(y_s \leq s\varepsilon) \leq \delta$ provided that we take

$$s \geq \frac{2\log(1/\delta)}{\log(1/\varepsilon) + \varepsilon - 1}.$$ □

Note that $\mathbb{P}(y_s \leq s\varepsilon)$ is the probability that a jump in $\mathrm{tr}_s^{\mathrm{H}}(P_\mu)$ corresponding to a certain fixed eigenvalue of A is ε-small. Observe that the heights of different jumps in $\mathrm{tr}_s^{\mathrm{H}}(P_\mu)$ are independent, because the quantities $\{\boldsymbol{u}_k^T \boldsymbol{x}_i\}_{i,k}$ are all independent due to the rotational invariance of the $\boldsymbol{x}_i$ and the orthogonality of the $\boldsymbol{u}_k$, so Lemma 9.2.2 can also be used to obtain conditions on s that bound the probability of having ε-small jumps associated with several different eigenvalues of A. The number of sample vectors s required to have an ε-small jump probability lower than δ reduces as ε decreases. At the same time, the number of Lanczos iterations required to approximate the quadratic forms with accuracy of order ε (and thus manage to detect the jumps in $\mathrm{tr}_s^{\mathrm{H}}(P_\mu)$) increases as ε becomes small. In Section 9.2.3 we obtain some bounds on the Lanczos error, which we will later compare with the ε-small jump probability bound from Lemma 9.2.2 in order to determine the best choice of parameters for the algorithm.

9.2.3 Lanczos approximation analysis

In this section we discuss the error in the Lanczos approximation of a quadratic form $\boldsymbol{x}^T h_\mu(A)\boldsymbol{x}$. Recall that we denote by

$$E_k(f, \mathcal{S}) = \min_{p \in \Pi_k} \max_{z \in \mathcal{S}} |f(z) - p(z)|$$

the best uniform norm error in the approximation of a continuous function f on the compact set $\mathcal{S} \subset \mathbb{R}$ with polynomials of degree up to k. If ψ_m denotes the Lanczos approximation to $\boldsymbol{x}^T f(A)\boldsymbol{x}$ from the Krylov subspace $\mathcal{K}_m(A, \boldsymbol{x})$, we have

$$|\boldsymbol{x}^T f(A)\boldsymbol{x} - \psi_m| \leq 2\|\boldsymbol{x}\|_2^2 E_{2m-1}(f, \Lambda(A) \cup \Lambda(T_m)), \tag{9.2.4}$$

see Proposition 2.5.12. Since the function $f = h_\mu$ is discontinuous at μ, we have that $E_{2m-1}(h_\mu, [a,b])$ is always at least $\frac{1}{2}$ if $\mu \in [a,b]$, so we cannot derive meaningful error bounds if we simply consider an interval $[a,b]$ that contains the whole spectrum of A. On the other hand, we can obtain bounds on the polynomial approximation error of h_μ on two disjoint intervals, one to the left and one to the right of μ. We start by stating a result for the polynomial approximation of the sign function on the union of two symmetric intervals $[-b,-a] \cup [a,b]$.

Proposition 9.2.3 ([65, Theorem 1])**.** Let $0 < a < b$. We have

$$\lim_{m\to\infty} \sqrt{m}\left(\frac{b+a}{b-a}\right)^m E_{2m+1}(\mathrm{sign}, [-b,-a] \cup [a,b]) = \frac{b-a}{\sqrt{\pi ab}}.$$

The statement of Proposition 9.2.3 implies that there exists a constant $C > 0$ such that

$$E_{2m+1}(\mathrm{sign}, [-b,-a] \cup [a,b]) \leq \frac{C}{\sqrt{m}}\left(\frac{b-a}{b+a}\right)^m,$$

and for m large enough we can take

$$C = \frac{b-a}{\sqrt{\pi ab}} + 1.$$

In a situation in which the spectrum of A is not symmetric with respect to the origin, using the bound in Proposition 9.2.3 to bound the error of the Lanczos method for the sign function can yield overly pessimistic results, since it would require us to consider intervals that are much wider than $\Lambda(A)$. In such a case, it would be more convenient to consider the polynomial approximation of the sign function on a nonsymmetric set. For instance, the asymptotics of $E_{2k+1}(\text{sign}, [-a,-1] \cup [1,b])$ for $a \neq b$ have been studied in [66], where the authors generalize Proposition 9.2.3 to [66, Theorem 1.1]. The result in [66] could be used to obtained more refined error bounds for the Lanczos approximation error, but the complexity of the statement, which involves Green's functions and integral representations, makes it significantly more difficult to employ in our setting.

We can easily turn Proposition 9.2.3 into a bound for the step function h_μ for any $\mu \in \mathbb{R}$. Let us define

$$b_1 := \lambda_1 - \mu, \qquad a_1 := \lambda_{\text{n}_\text{e}(\mu)} - \mu, \qquad a_2 := \lambda_{\text{n}_\text{e}(\mu)+1} - \mu, \qquad b_2 := \lambda_n - \mu,$$

and let

$$a := \min\{|a_1|, |a_2|\} \qquad \text{and} \qquad b := \max\{|b_1|, |b_2|\}.$$

Note that $\Lambda(A) \subset [\lambda_1, \lambda_{\text{n}_\text{e}(\mu)}] \cup [\lambda_{\text{n}_\text{e}(\mu)+1}, \lambda_n]$ and recall that $h_\mu(x) = -\frac{1}{2}\,\text{sign}(x-\mu) + \frac{1}{2}$, so we have

$$\begin{aligned} E_{2m+1}(h_\mu, [\lambda_1, \lambda_{\text{n}_\text{e}(\mu)}] \cup [\lambda_{\text{n}_\text{e}(\mu)+1}, \lambda_n]) &= E_{2m+1}(h_0, [b_1, a_1] \cup [a_2, b_2]) \\ &= \frac{1}{2} E_{2m+1}(\text{sign}, [b_1, a_1] \cup [a_2, b_2]) \\ &\leq \frac{1}{2} E_{2m+1}(\text{sign}, [-b, -a] \cup [a, b]) \\ &\leq \frac{C}{2\sqrt{m}} \left(\frac{b-a}{b+a}\right)^m, \end{aligned}$$

with $C = 1 + (b-a)/\sqrt{\pi ab}$, where we used Proposition 9.2.3 for the last inequality. Denoting the relative spectral gap associated to μ by $\theta := a/b$, we obtain

$$E_{2m+1}(h_\mu, [\lambda_1, \lambda_{\text{n}_\text{e}(\mu)}] \cup [\lambda_{\text{n}_\text{e}(\mu)+1}, \lambda_n]) \leq \frac{C}{2\sqrt{m}} \left(\frac{1-\theta}{1+\theta}\right)^m. \tag{9.2.5}$$

We can combine this result with (9.2.4) to bound the error in the Lanczos approximation of quadratic forms with $P_\mu = h_\mu(A)$, provided that no eigenvalue of T_m lies within the gap $(\lambda_{\text{n}_\text{e}(\mu)}, \lambda_{\text{n}_\text{e}(\mu)+1})$. Unfortunately, this cannot be excluded a priori; however, since there must be at least one eigenvalue of A in the open interval between two consecutive Ritz values [104, Section 4], there can be at most one eigenvalue of T_m in the interval $(\lambda_{\text{n}_\text{e}(\mu)}, \lambda_{\text{n}_\text{e}(\mu)+1})$. This fact can be proved using the theory of orthogonal polynomials, see for instance [83, Theorem 1.21], using the fact that the Ritz values are the zeros of an orthogonal polynomial associated with a Gauss quadrature rule, see, e.g., [87, Theorem 6.2]. In the analysis that follows, we assume for simplicity that there are no Ritz values in the gap $(\lambda_{\text{n}_\text{e}(\mu)}, \lambda_{\text{n}_\text{e}(\mu)+1})$. We discuss how we can modify our approach to deal with this issue in Remark 9.2.6.

Remark 9.2.4. In the literature, several techniques have been used to obtain approximations to the sign matrix function that circumvent the problem of Ritz values getting arbitrarily close to zero [150], for instance by writing $\text{sign}(z) = z/(z^2)^{-1/2}$ and constructing an approximation of $(A^2)^{-1/2}\boldsymbol{x}$ using the Krylov subspace $\mathcal{K}_m(A^2, \boldsymbol{x})$, which ensures that the eigenvalues of the projected matrix are bounded away from 0, since A^2 is positive definite as long as A is nonsingular. Alternatively, one can construct an approximation using harmonic Ritz values [127], which are always outside of the interval between the largest negative and the smallest positive eigenvalue of A, so they avoid the discontinuity of the sign function. Since in this work our goal is to approximate $\boldsymbol{x}^T h_\mu(A)\boldsymbol{x}$ for several μ, we opted to use the standard Lanczos approximation (9.1.2), for which it is very cheap and straightforward to compute approximations for many different functions.

Combining (9.2.4) with (9.2.5), we obtain the following bound on the Lanczos approximation error.

Proposition 9.2.5. Let ψ_m be the Lanczos approximation (9.1.2) to $\boldsymbol{x}^T h_\mu(A)\boldsymbol{x}$ from the Krylov subspace $\mathcal{K}_m(A, \boldsymbol{x})$, and assume that there is no eigenvalue of T_m in the interval $(\lambda_{\mathrm{n_e}(\mu)}, \lambda_{\mathrm{n_e}(\mu)+1})$. We have

$$|\boldsymbol{x}^T h_\mu(A)\boldsymbol{x} - \psi_m| \leq \frac{C\|\boldsymbol{x}\|_2^2}{\sqrt{m-1}} \left(\frac{1-\theta}{1+\theta}\right)^{m-1}, \tag{9.2.6}$$

where C and θ are defined above.

Note that if we take a different value of μ in the interval $(\lambda_{\mathrm{n_e}(\mu)}, \lambda_{\mathrm{n_e}(\mu)+1})$, the error $|\boldsymbol{x}^T h_\mu(A)\boldsymbol{x} - \psi_m|$ remains the same, because the function h_μ takes the same values on the eigenvalues of A and T_m and hence the matrix functions $h_\mu(A)$ and $h_\mu(T_m)$ do not change. On the other hand, the bound in Proposition 9.2.5 depends on θ, which in turn depends on μ. It is quite easy to see that the largest value of θ is obtained when μ is precisely in the middle of the gap, i.e. $\mu = \frac{1}{2}(\lambda_{\mathrm{n_e}(\mu)} + \lambda_{\mathrm{n_e}(\mu)+1})$. This value of μ gives us the sharpest bound in (9.2.6).

The convergence predicted by Proposition 9.2.5 is at least exponential with rate $(1-\theta)/(1+\theta)$, with in addition an $O(1/\sqrt{m})$ factor which is independent of the gap width. In order to have an error $|\boldsymbol{x}^T h_\mu(A)\boldsymbol{x} - \psi_m| \leq \varepsilon$, the number of Lanczos iterations m should satisfy

$$m + \frac{\log(m-1)}{2\log\big((1+\theta)/(1-\theta)\big)} \geq 1 + \frac{\log\big(C\|\boldsymbol{x}\|_2^2/\varepsilon\big)}{\log\big((1+\theta)/(1-\theta)\big)}. \tag{9.2.7}$$

Assuming that the relative gap θ is large enough, we do not lose much by discarding the $\log(m-1)$ term, and we can take

$$m \geq 1 + \frac{\log\big(C\|\boldsymbol{x}\|_2^2/\varepsilon\big)}{\log\big((1+\theta)/(1-\theta)\big)}. \tag{9.2.8}$$

On the other hand, in a situation where the spectral gap is very small, the Lanczos method effectively converges like $O(1/\sqrt{m})$. Note that if $\boldsymbol{x}$ is a Gaussian random vector we have $\mathbb{E}[\|\boldsymbol{x}\|_2^2] = n$, so with a very small gap for which $\theta \approx 0$ we would need to take

$$m \gtrsim \frac{C^2 n^2}{\varepsilon^2},$$

which is definitely unfeasible. This means that when using the Lanczos method we can only hope to approximate $\boldsymbol{x}^T h_\mu(A)\boldsymbol{x}$ accurately when μ is inside a relatively large gap in the spectrum of A, and we should instead expect to have quite a large error whenever μ is close to an eigenvalue of A.

A crucial requirement for using the bound in Proposition 9.2.5 is the knowledge of the relative gap width θ. Since the gaps in the spectrum are not known to us in advance, we cannot use this result to check if the error of the Lanczos method is below a certain tolerance ε. On the other hand, Proposition 9.2.5 can be useful to choose the number m of Lanczos iterations at the start of the algorithm. Indeed, if our goal is to find all the gaps in the spectrum of A with relative width at least θ, for any given ε we can select m using (9.2.8) to ensure that the error $|\boldsymbol{x}^T h_\mu(A)\boldsymbol{x} - \psi_m| \leq \varepsilon$ whenever μ is inside a gap with width at least θ. However, we are still unable to determine for which values of μ this condition is satisfied. The final component that we need to make the algorithm work is an a posteriori bound on the error, which will allow us to detect when $|\boldsymbol{x}^T h_\mu(A)\boldsymbol{x} - \psi_m| \leq \varepsilon$, without knowing the gaps in the spectrum in advance. For this purpose we can use the a posteriori error bound from Section 3.2.1, which we adapt to the step function h_μ in Section 9.2.4.

Remark 9.2.6. If there is an eigenvalue of T_m inside the gap $(\lambda_{\mathrm{n_e}(\mu)}, \lambda_{\mathrm{n_e}(\mu)+1})$, the results above do not hold. However, if the gap in the spectrum of A has relative width θ, then there must be a corresponding gap in $\Lambda(A) \cup \Lambda(T_m)$ with a relative width at least $\theta/2$, since there is at most one Ritz value contained in the interval $(\lambda_{\mathrm{n_e}(\mu)}, \lambda_{\mathrm{n_e}(\mu)+1})$. In other words, there is an interval of relative width $\theta/2$ where there are no eigenvalues of A or T_m, and hence when μ is inside that interval the Lanczos approximation error satisfies the bound (9.2.6) with θ replaced by $\theta/2$.

9.2.4 An a posteriori error bound for the Lanczos algorithm

In this section we adapt the a posteriori error bound given in Theorem 3.2.3 for the approximation of $\boldsymbol{x}^T f(A)\boldsymbol{x}$ with a rational Krylov method to the case of the Lanczos approximation of $f = h_\mu$. The proof in the case of $f = h_\mu$ requires some caution since the function h_μ is discontinuous at μ, so in the discussion that follows we briefly retrace the proof of Theorem 3.2.3 and adapt it to the current setting. If Γ is a simple closed curve that surrounds the eigenvalues of A below μ, we have

$$P_\mu = h_\mu(A) = \frac{1}{2\pi i}\int_\Gamma (zI - A)^{-1}\,\mathrm{d}z.$$

If the eigenvalues of T_m that are smaller than μ also lie in the interior of Γ, we have

$$h_\mu(A)\boldsymbol{x} - \|\boldsymbol{x}\|_2 V_m h_\mu(T_m)\boldsymbol{e}_1 = \frac{1}{2\pi i}\int_\Gamma (zI - A)^{-1}\boldsymbol{r}_m(z)\,\mathrm{d}z,$$

where

$$\boldsymbol{r}_m(z) = \boldsymbol{x} - (zI - A)\boldsymbol{x}_m(z), \qquad \text{with} \qquad \boldsymbol{x}_m(z) = \|\boldsymbol{x}\|_2 V_m (zI - T_m)^{-1}\boldsymbol{e}_1,$$

that is, $\boldsymbol{r}_m(z)$ is the residual of a shifted linear system after m iterations of the Lanczos method. With the same algebraic manipulations used in Section 3.2.1, we obtain

$$\boldsymbol{x}^T h_\mu(A)\boldsymbol{x} - \|\boldsymbol{x}\|_2^2 \boldsymbol{e}_1^T h_\mu(T_m)\boldsymbol{e}_1 = \frac{\|\boldsymbol{x}\|_2^2}{2\pi i}\int_\Gamma \varphi_m(z)^2 \boldsymbol{v}_{m+1}^T (zI - A)^{-1}\boldsymbol{v}_{m+1}\,\mathrm{d}z,$$

with

$$\varphi_m(z) = t_{m+1,m}\boldsymbol{e}_m^T (zI - T_m)^{-1}\boldsymbol{e}_1,$$

where $t_{m+1,m}$ denotes the element in position $(m+1, m)$ of the matrix $\underline{T}_m$. Using an eigenvalue decomposition $T_m = U_m D_m U_m^T$, with U_m orthogonal and $D_m = \mathrm{diag}(\theta_1, \ldots, \theta_m)$, we can define

$$\big[\alpha_1, \ldots, \alpha_m\big] := t_{m+1,m}\boldsymbol{e}_m^T U_m, \qquad \big[\beta_1, \ldots, \beta_m\big]^T := U_m^T \boldsymbol{e}_1$$

and

$$\gamma_j := \sum_{k\,:\,k\neq j} \alpha_k\beta_k \frac{1}{\theta_j - \theta_k}, \qquad j = 1, \ldots, m.$$

With these definitions, the function $\varphi_m(z)^2$ can be rewritten in the form

$$\varphi_m(z)^2 = \sum_{j=1}^m \alpha_j^2\beta_j^2 \frac{1}{(z-\theta_j)^2} + 2\sum_{j=1}^m \alpha_j\beta_j\gamma_j \frac{1}{z-\theta_j}.$$

Using the residue theorem, with the same procedure used in Section 3.2.1 we obtain

$$\boldsymbol{x}^T h_\mu(A)\boldsymbol{x} - \|\boldsymbol{x}\|_2^2 \boldsymbol{e}_1^T h_\mu(T_m)\boldsymbol{e}_1 = \|\boldsymbol{x}\|_2^2 \boldsymbol{v}_{m+1}^T g_m(A)\boldsymbol{v}_{m+1}, \tag{9.2.9}$$

with

$$g_m(z) := \begin{cases} \displaystyle\sum_{j\,:\,\theta_j > \mu} \alpha_j^2\beta_j^2 \frac{1}{(z-\theta_j)^2} + 2\alpha_j\beta_j\gamma_j \frac{1}{z-\theta_j} & \text{if } z < \mu, \\ \displaystyle -\sum_{j\,:\,\theta_j < \mu} \alpha_j^2\beta_j^2 \frac{1}{(z-\theta_j)^2} - 2\alpha_j\beta_j\gamma_j \frac{1}{z-\theta_j} & \text{if } z > \mu. \end{cases} \tag{9.2.10}$$

We can use (9.2.9) to obtain the following upper bound for the Lanczos approximation error, which corresponds to the upper bound in Theorem 3.2.3.

Proposition 9.2.7. Let ψ_m be the Lanczos approximation (9.1.2) to $\boldsymbol{x}^T h_\mu(A)\boldsymbol{x}$ from the Krylov subspace $\mathcal{K}_m(A, \boldsymbol{x})$. We have

$$|\boldsymbol{x}^T h_\mu(A)\boldsymbol{x} - \psi_m| \leq \|\boldsymbol{x}\|_2^2 \max_{z\in\Lambda(A)} |g_m(z)| \leq \|\boldsymbol{x}\|_2^2 \max_{z\in[\lambda_{\min},\lambda_{\max}]} |g_m(z)|,$$

where g_m is defined in (9.2.10).

In contrast with what happens when the function f is continuous over the whole spectral interval of A, in this case the function $g_m(z)$ is discontinuous at μ and the lower bound from Theorem 3.2.3 no longer holds. Indeed, if we denote by $(\lambda_k, \boldsymbol{u}_k)$ the normalized eigenpairs of the matrix A, we can write

$$\boldsymbol{v}_{m+1}^T g_m(A)\boldsymbol{v}_{m+1} = \sum_{k=1}^{m} g_m(\lambda_k)(\boldsymbol{u}_k^T\boldsymbol{v}_{m+1})^2, \qquad \text{with} \qquad \sum_{k=1}^{m}(\boldsymbol{u}_k^T\boldsymbol{v}_{m+1})^2 = 1.$$

This expression shows that the quadratic form $\boldsymbol{v}_{m+1}^T g_m(A)\boldsymbol{v}_{m+1}$ can potentially take any value in the convex hull of the set $\{g_m(\lambda_k)\}_{k=1}^m$, depending on the coefficients $(\boldsymbol{u}_k^T\boldsymbol{v}_{m+1})^2$. In the case that $g_m(\lambda_k)$ takes both positive and negative values, it is possible to have $\boldsymbol{v}_{m+1}^T g_m(A)\boldsymbol{v}_{m+1} = 0$, so the lower bound

$$|\boldsymbol{x}^T h_\mu(A)\boldsymbol{x} - \|\boldsymbol{x}\|_2^2 \boldsymbol{e}_1^T h_\mu(T_m)\boldsymbol{e}_1| \geq \|\boldsymbol{x}\|_2^2 \min_{z\in[\lambda_{\min},\lambda_{\max}]} |g_m(z)|$$

may not hold. Note that this argument is not in contradiction with the statement of Theorem 3.2.3 when the function g_m is continuous over the whole spectral interval of A, because if $g_m(\lambda_k)$ takes both negative and positive values there must be a point $z \in [\lambda_{\min}, \lambda_{\max}]$ such that $g_m(z) = 0$ by continuity of g_m.

Example 9.2.2. In this example we compare the error of the Lanczos approximation of $\boldsymbol{x}^T h_\mu(A)\boldsymbol{x}$ with the a priori error bound from Proposition 9.2.5, the a posteriori error bound form Proposition 9.2.7, and the simple error estimate $|\boldsymbol{x}^T h_\mu(A)\boldsymbol{x} - \psi_m| \approx |\psi_m - \psi_{m+1}|$. We consider a symmetric matrix A of size $n = 1000$ with $\Lambda(A) \subset [0, 40]$, and gaps in the intervals $[10, 11]$ and $[20, 25]$. We plot the convergence of the Lanczos method in Figure 9.2, where on the left plot $\mu = 22$ is contained in the larger gap, and on the right plot $\mu = 10.6$ is inside the smaller gap. In the legend, the "central μ" a priori bound refers to the bound obtained by taking μ in the middle of the gap, as mentioned below Proposition 9.2.5.

Note that the convergence of the Lanczos method is much slower when μ is inside the smaller gap, and in both cases the convergence behavior is irregular and oscillating. The "central μ" is the most accurate between the two a priori bounds, but it does not capture the correct convergence rate for $\mu = 10.6$, which is further from the middle of the spectrum; in this case, a better a priori bound would be obtained by using a convergence result with nonsymmetric intervals, such as [66, Theorem 1.1]. The a posteriori bound is also quite irregular, but it is able to capture the convergence rate of the Lanczos approximation correctly in both cases, although it is off by a constant factor. Surprisingly, the error estimate $|\boldsymbol{x}^T h_\mu(A)\boldsymbol{x} - \psi_m| \approx |\psi_m - \psi_{m+1}|$ is very accurate, and it seems to almost never underestimate the error, in contrast with the experiments in Section 3.4. This behavior can be explained in the following way: the error estimate based on consecutive differences satisfies the triangle inequality

$$|\psi_m - \psi_{m+1}| \geq \big|\, |\boldsymbol{x}^T h_\mu(A)\boldsymbol{x} - \psi_m| - |\boldsymbol{x}^T h_\mu(A)\boldsymbol{x} - \psi_{m+1}| \,\big|,$$

and due to the oscillations in the convergence of the Lanczos approximation, the two right-hand side terms $|\boldsymbol{x}^T h_\mu(A)\boldsymbol{x} - \psi_m|$ and $|\boldsymbol{x}^T h_\mu(A)\boldsymbol{x} - \psi_{m+1}|$ often have significantly different moduli, so we have

$$|\,|\psi - \psi_m| - |\psi - \psi_{m+1}|\,| \geq c \max\{|\psi - \psi_m|, |\psi - \psi_{m+1}|\}, \qquad \psi := \boldsymbol{x}^T h_\mu(A)\boldsymbol{x}, \tag{9.2.11}$$

for a constant c that is close to 1. For instance, if $|\psi - \psi_m| > |\psi - \psi_{m+1}|$, we can take

$$c = 1 - \frac{|\psi - \psi_{m+1}|}{|\psi - \psi_m|}.$$

The inequality (9.2.11) explains why the consecutive difference estimate is so often an upper bound when the Lanczos method has oscillating convergence, as in the current setting.

9.3 Final algorithm

In this section we put together all the theoretical results obtained in Section 9.2 to improve Algorithm 9.1 and obtain an efficient algorithm that also provides some guarantees on its output.

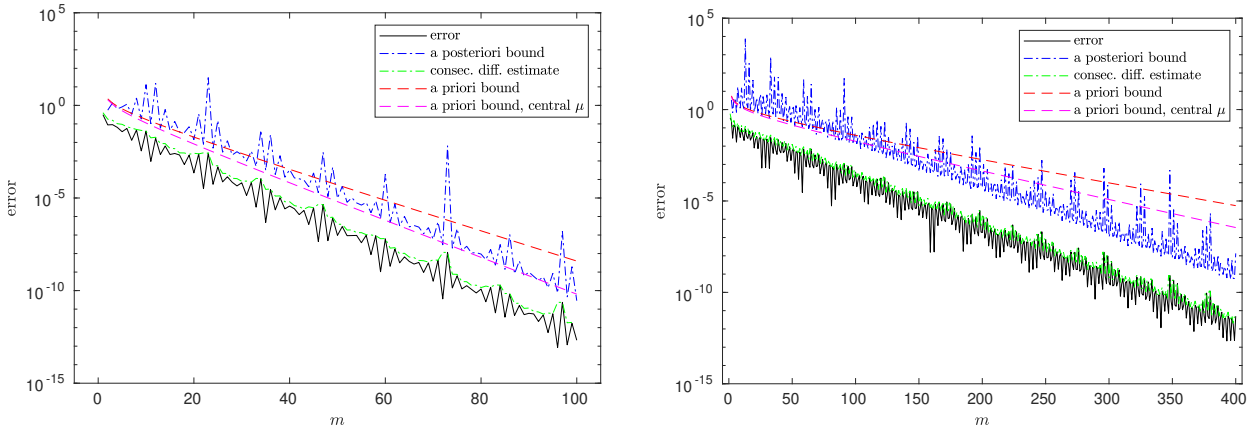


Figure 9.2 – Approximation of $\boldsymbol{x}^T h_\mu(A)\boldsymbol{x}$ with the Lanczos algorithm, and comparison with the a priori error bound from Proposition 9.2.5, the a posteriori error bound from Proposition 9.2.7 and the consecutive difference estimate. The spectrum of A has gaps in the intervals $[10, 11]$ and $[20, 25]$. Left: $\mu = 22$. Right: $\mu = 10.6$.

9.3.1 Choosing the input parameters

First of all, we need a criterion to decide the number s of Hutchinson sample vectors and the number m of Lanczos iterations. For this purpose, given a failure probability δ and a target relative gap width θ, we set the goal to find all gaps in the spectrum of A with relative width greater than or equal to θ, with probability $1 - \delta$.

How should we choose s and m so that all these gaps are found? Let us consider a candidate gap $[\mu_1, \mu_2] \subset \mathbb{R}$ and fix a parameter $\varepsilon > 0$, and say that we take m such that the Lanczos approximation error for each quadratic form $\boldsymbol{x}_i^T P_\mu \boldsymbol{x}_i$ is smaller than $\varepsilon/2$ for all μ inside the gap, so that $\operatorname{tr}_s^{\mathrm{H}}(P_\mu)$ is contained in an interval of width smaller than ε for all $\mu \in [\mu_1, \mu_2]$. Then there are two possible scenarios: either there are no eigenvalues of A in the interval $[\mu_1, \mu_2]$, or all the jumps in $\operatorname{tr}_s^{\mathrm{H}}(P_\mu)$ associated to eigenvalues inside that interval are ε-small (more precisely, the sum of the heights of these jumps is smaller than ε). This second event has a small probability of occurring, which can be bounded using Lemma 9.2.2, so we can conclude that $[\mu_1, \mu_2]$ is a gap in the spectrum of A with high probability.

The probability that all jumps in $\operatorname{tr}_s^{\mathrm{H}}(P_\mu)$ associated to eigenvalues in $[\mu_1, \mu_2]$ are ε-small is smaller than the probability of a single jump being ε-small, so we can impose

$$\mathbb{P}(y_s \leq s\varepsilon) \leq \delta,$$

where y_s is a χ^2 distribution with s degrees of freedom, see Section 9.2.2. By Lemma 9.2.2, this condition is satisfied provided that

$$s \geq \frac{2\log(1/\delta)}{\log(1/\varepsilon) + \varepsilon - 1}.$$

If the relative width of $[\mu_1, \mu_2]$ is at least θ, by Proposition 9.2.5 we have

$$\left| \operatorname{tr}_s^{\mathrm{H}}(P_\mu) - \frac{1}{s}\sum_{i=1}^{s} \psi_m^{(i)} \right| \leq \frac{C\sum_{i=1}^{s} \|\boldsymbol{x}_i\|_2^2}{s\sqrt{m-1}} \left(\frac{1-\theta}{1+\theta}\right)^{m-1},$$

where we denote by $\psi_m^{(i)}$ the approximation of $\boldsymbol{x}_i^T P_\mu \boldsymbol{x}_i$ after m Lanczos iterations. By (9.2.8), the above bound is smaller than $\varepsilon/2$ if we take

$$m \geq 1 + \frac{\log\left(2C\sum_{i=1}^{s} \|\boldsymbol{x}_i\|_2^2/s\varepsilon\right)}{\log\left((1+\theta)/(1-\theta)\right)}, \qquad C = \frac{1-\theta}{\sqrt{\pi\theta}} + 1. \tag{9.3.1}$$

Note that although the a priori bound in Proposition 9.2.5 allows us to say that the Lanczos error is smaller than $\varepsilon/2$ when μ is inside a gap wider than θ, we cannot use it to determine for which values

of μ this actually happens. To detect the gaps in practice, we are going to use either the a posteriori bound from Proposition 9.2.7 or the error estimate based on consecutive differences, and assume that it is at least as accurate as the a priori bound, i.e., that it is also smaller than $\varepsilon/2$ inside a gap with relative width θ. This assumption is usually satisfied, as shown in Example 9.2.2, at least in the central portion of each gap.

We have obtained bounds on s and m that depend on δ, θ and ε, where the latter is a parameter that can be freely chosen. In order to obtain an algorithm that is as efficient as possible, we should aim to minimize the term that dominates the computational cost of the algorithm, which is the $O(sm\,\texttt{nnz}(A))$ term corresponding to the sm matrix vector products with A. We have

$$sm \geq \frac{2\log(2/\delta)}{\log(1/\varepsilon)+\varepsilon-1} \cdot \left(1 + \frac{\log\left(2C\sum_{i=1}^{s}\|\boldsymbol{x}_i\|_2^2/s\varepsilon\right)}{\log\left((1+\theta)/(1-\theta)\right)}\right),$$

and ε should be chosen in order to minimize the right hand side. With some simple computations one can show that the function

$$f(\varepsilon) = \frac{1}{\log(1/\varepsilon)+\varepsilon-1} \cdot \left(1 + \frac{\log\left(2Cn/\varepsilon\right)}{\log\left((1+\theta)/(1-\theta)\right)}\right)$$

is increasing for $\varepsilon \in (0,1)$ if $2Cn \geq 1$, a condition that for any fixed θ is satisfied when n is sufficiently large. Recalling that $\mathbb{E}[\|\boldsymbol{x}_i\|^2] = n$ for all i, this implies that we should take ε as small as possible, and since s must be an integer, in practice the most efficient choice is to take ε so that the probability of a jump being ε-small is lower than δ already when $s = 1$, i.e.,

$$\varepsilon \leq e^{-1}\delta^2. \tag{9.3.2}$$

In other words, we should simply use a single quadratic form $\boldsymbol{x}^T P_\mu \boldsymbol{x}$ to estimate $\operatorname{tr}(P_\mu)$, instead of using the more general stochastic trace estimator with s different random vectors.

9.3.2 Detecting gaps in the spectrum

To detect the gaps in the spectrum, we use an a posteriori error bound to determine when the error of the Lanczos approximation is smaller than $\varepsilon/2$. We focus on the case $s = 1$, with a single random Gaussian vector $\boldsymbol{x} \in \mathbb{R}^n$, since we concluded that it is the most efficient choice in Section 9.3.1. More precisely, for each μ we perform m iterations of the Lanczos method to obtain $\psi_m(\mu) \approx \boldsymbol{x}^T P_\mu \boldsymbol{x}$ with the approximation (9.1.2), and we use either Proposition 9.2.7 or the estimate based on consecutive differences to obtain

$$|\boldsymbol{x}^T P_\mu \boldsymbol{x} - \psi_m(\mu)| \leq \texttt{bound}_m(\mu),$$

where either

$$\texttt{bound}_m(\mu) = \|\boldsymbol{x}\|_2^2 \max_{z\in[\lambda_{\min},\lambda_{\max}]} |g_m(z)| \qquad \text{with } g_m(z) \text{ as defined in (9.2.10)},$$

or $\texttt{bound}_m(\mu) = c|\psi_m(\mu) - \psi_{m+1}(\mu)|$, with a safety factor $c \geq 1$. As shown in Example 9.2.2, the consecutive difference error estimate is usually more accurate than the a posteriori error bound from Proposition 9.2.7, and very often we can also expect it to be an actual upper bound due to the oscillation in the convergence of the Lanczos algorithm. In the discussion that follows, we are going to assume that $\texttt{bound}_m(\mu)$ is always an upper bound for the error $|\boldsymbol{x}^T P_\mu \boldsymbol{x} - \psi_m(\mu)|$; since there is no guarantee that this is true for the consecutive difference estimate, we will discuss some ways to make it more robust in Example 9.3.1. For all μ, letting

$$\begin{aligned} \mathtt{q}_m^{\mathrm{U}}(\mu) &:= \psi_m(\mu) + \texttt{bound}_m(\mu), \\ \mathtt{q}_m^{\mathrm{L}}(\mu) &:= \psi_m(\mu) - \texttt{bound}_m(\mu), \end{aligned}$$

we have $\boldsymbol{x}^T P_\mu \boldsymbol{x} \in [\mathtt{q}_m^{\mathrm{L}}(\mu), \mathtt{q}_m^{\mathrm{U}}(\mu)]$.

Recalling that the quadratic form $\boldsymbol{x}^T P_\mu \boldsymbol{x}$ is an increasing function of μ, we can refine the upper and lower bounds $\mathfrak{q}_m^{\mathrm{U}}(\mu)$ and $\mathfrak{q}_m^{\mathrm{L}}(\mu)$. Indeed, we can also take the upper and lower bounds to be increasing in μ: specifically, for any $\mu < \mu'$ we must have

$$\boldsymbol{x}^T P_\mu \boldsymbol{x} \leq \boldsymbol{x}^T P_{\mu'} \boldsymbol{x} \leq \mathfrak{q}_m^{\mathrm{U}}(\mu') \qquad \text{and} \qquad \boldsymbol{x}^T P_{\mu'} \boldsymbol{x} \geq \boldsymbol{x}^T P_\mu \boldsymbol{x} \geq \mathfrak{q}_m^{\mathrm{L}}(\mu),$$

so we can replace the bounds $\mathfrak{q}_m^{\mathrm{U}}(\mu)$ and $\mathfrak{q}_m^{\mathrm{L}}(\mu)$ with, respectively,

$$\widehat{\mathfrak{q}}_m^{\mathrm{U}}(\mu) := \min_{\mu' \geq \mu} \mathfrak{q}_m^{\mathrm{U}}(\mu') \qquad \text{and} \qquad \widehat{\mathfrak{q}}_m^{\mathrm{L}}(\mu) := \max_{\mu' \leq \mu} \mathfrak{q}_m^{\mathrm{L}}(\mu'). \tag{9.3.3}$$

Note that the bounds $\widehat{\mathfrak{q}}_m^{\mathrm{U}}(\mu)$ and $\widehat{\mathfrak{q}}_m^{\mathrm{L}}(\mu)$ are by construction increasing functions of μ. We can further improve the bounds by taking the best ones over several values of m. Indeed, given a set of positive integers $\mathcal{M} \subset \mathbb{N}$, we can define

$$\widehat{\mathfrak{q}}_{\mathcal{M}}^{\mathrm{U}}(\mu) := \min_{m \in \mathcal{M}} \widehat{\mathfrak{q}}_m^{\mathrm{U}}(\mu) \qquad \text{and} \qquad \widehat{\mathfrak{q}}_{\mathcal{M}}^{\mathrm{L}}(\mu) := \max_{m \in \mathcal{M}} \widehat{\mathfrak{q}}_m^{\mathrm{L}}(\mu), \tag{9.3.4}$$

and we still have $\boldsymbol{x}^T P_\mu \boldsymbol{x} \in [\widehat{\mathfrak{q}}_{\mathcal{M}}^{\mathrm{L}}(\mu), \widehat{\mathfrak{q}}_{\mathcal{M}}^{\mathrm{U}}(\mu)]$. Using a set $\mathcal{M}$ with more than a single value of m can be useful to improve the bounds in the case when there is an eigenvalue of T_m inside one of the gaps. In such a situation, by considering bounds for $\mathcal{M} = \{m-d, \dots, m-1, m\}$ there is a higher chance that one of the matrices T_{m-j}, $j = 0, \dots, d$ has no eigenvalues in the gap, so the bounds $\widehat{\mathfrak{q}}_{\mathcal{M}}^{\mathrm{U}}(\mu)$ and $\widehat{\mathfrak{q}}_{\mathcal{M}}^{\mathrm{L}}(\mu)$ are likely going to be better than $\widehat{\mathfrak{q}}_m^{\mathrm{U}}(\mu)$ and $\widehat{\mathfrak{q}}_m^{\mathrm{L}}(\mu)$.

After incorporating these improvements, we can conclude that if ε is chosen according to (9.3.2) and for some $\mu_1 < \mu_2$ we have $|\widehat{\mathfrak{q}}_{\mathcal{M}}^{\mathrm{U}}(\mu_2) - \widehat{\mathfrak{q}}_{\mathcal{M}}^{\mathrm{L}}(\mu_1)| \leq \varepsilon$, then there are no eigenvalues of A in the interval $[\mu_1, \mu_2]$ with probability at least $1 - \delta$.

9.3.3 Algorithm summary

The analysis in the previous sections leads to the following algorithm, whose pseudocode is given in Algorithm 9.2. Given an input failure probability δ and target relative gap width θ, we first compute ε according to (9.3.2) such that the probability of having an ε-small jump at an eigenvalue in Hutchinson's trace estimator is less than δ when using a single random vector (i.e., with $s = 1$). We then use Proposition 9.2.5 to select the number m of Lanczos iterations that ensures that the error of the Lanczos approximation to the quadratic form $\boldsymbol{x}^T h_\mu(A) \boldsymbol{x}$ is smaller than $\varepsilon/2$ whenever μ is inside a gap with relative width at least θ. As an alternative, we could fix the number of Lanczos iterations m a priori, but by doing so we would lose the guarantee that all the gaps with width larger than θ are located. Then, we run m iterations of the Lanczos method to compute the approximation (9.1.2) and the upper and lower a posteriori bounds $\widehat{\mathfrak{q}}_{\mathcal{M}}^{\mathrm{U}}(\mu)$ and $\widehat{\mathfrak{q}}_{\mathcal{M}}^{\mathrm{L}}(\mu)$ defined in (9.3.4), for all input values of μ and a set $\mathcal{M}$ of integers smaller than or equal to m, such as $\{m-d, \dots, m-1, m\}$, with, e.g., $d = 2$. All the intervals $[\mu_1, \mu_2]$ such that $|\widehat{\mathfrak{q}}_{\mathcal{M}}^{\mathrm{U}}(\mu_2) - \widehat{\mathfrak{q}}_{\mathcal{M}}^{\mathrm{L}}(\mu_1)| \leq \varepsilon$ are hence gaps in the spectrum of A with probability at least $1 - \delta$.

Note that m is selected so that the a priori error bound on the approximation of $\boldsymbol{x}^T h_\mu(A) \boldsymbol{x}$ from Proposition 9.2.5 is smaller than $\varepsilon/2$ whenever μ is inside a gap with relative width at least θ, so, assuming that the a posteriori error bound or estimate is at least as accurate as the a priori one, we expect that all gaps of relative width θ or larger are detected by Algorithm 9.2. Numerical evidence shows that this assumption is generally satisfied when μ is not too close to the extrema of the gap and ε is not too large. We also mention that, although Algorithm 9.2 cannot explicitly find the gap between the k-th and the $(k+1)$-th eigenvalue for a given k, it still computes estimates for $\mathrm{tr}(P_{\mu_j})$ for all $\mu_j \in \boldsymbol{\mu}$, so we know the approximate number of eigenvalues below each detected gap and thus we can roughly determine if one of the gaps found by the algorithm is likely to be the one between λ_k and λ_{k+1} (of course, we can only expect that Algorithm 9.2 will detect this gap if we assume that it is wide enough). If more accuracy on the number of eigenvalues below each gap is required, then one should use a larger number s of sample vectors for Hutchinson's trace estimator. Alternatively, if it is feasible to compute an LDL^T factorization, it can be used to compute the exact number of eigenvalues below a certain gap in order to verify if it corresponds to the searched gap between the eigenvalues λ_k and λ_{k+1}.

Algorithm 9.2 Eigenvalue gap finder

Input: Symmetric matrix $A \in \mathbb{R}^{n\times n}$, failure probability δ, target relative gap size θ, vector of test parameters $\boldsymbol{\mu} = [\mu_1, \dots, \mu_{N_f}]$

Output: $\mathtt{tr}_j \approx \operatorname{tr}(P_{\mu_j})$ for $j = 1, \dots, N_f$, intervals $[a_i, b_i]$ that contain no eigenvalues of A with probability $1 - \delta$ each

1: Compute ε according to (9.3.2), ensuring that the probability of a jump being smaller than ε with $s = 1$ Hutchinson vectors is smaller than δ.
2: Draw a random Gaussian vector $\boldsymbol{x} \in \mathbb{R}^n$.
3: Compute m according to (9.3.1), ensuring that the Lanczos approximation (9.1.2) to $\boldsymbol{x}^T P_\mu \boldsymbol{x}$ has an error smaller than $\varepsilon/2$ if μ is within a gap of relative width θ or larger.
4: Select a set $\mathcal{M}$ of integers smaller than or equal to m, such as $\mathcal{M} = \{m-4, \dots, m-1, m\}$.
5: Compute V_m and $\underline{T}_m$ by running m iterations of the Lanczos algorithm with the matrix A and the vector $\boldsymbol{x}$.
6: Compute the eigendecomposition $T_m = U_m^T D_m U_m$ and $\boldsymbol{w}_m = \|\boldsymbol{x}\|_2 U_m \boldsymbol{e}_1$.
7: **for** $j = 1, \dots, N_f$ **do**
8: Compute $\psi_{m,j} = \boldsymbol{w}_m^T h_{\mu_j}(D_m) \boldsymbol{w}_m$.
9: **for** $m' \in \mathcal{M}$ **do**
10: Compute the upper and lower bounds $\widehat{\mathtt{q}}^{\mathrm{U}}_{m'}(\mu_j)$ and $\widehat{\mathtt{q}}^{\mathrm{L}}_{m'}(\mu_j)$ using (9.3.3), where $\mathtt{bound}_{m'}(\mu_j)$ is obtained either with Proposition 9.2.7 or by running Lanczos for one additional iteration to compute $|\psi_{m',j} - \psi_{m'+1,j}|$.
11: **end for**
12: Compute the bounds $\widehat{\mathtt{q}}^{\mathrm{U}}_{\mathcal{M}}(\mu_j) = \min_{m'\in\mathcal{M}} \widehat{\mathtt{q}}^{\mathrm{U}}_{m'}(\mu_j)$ and $\widehat{\mathtt{q}}^{\mathrm{L}}_{\mathcal{M}}(\mu_j) = \max_{m'\in\mathcal{M}} \widehat{\mathtt{q}}^{\mathrm{L}}_{m'}(\mu_j)$.
13: **end for**
14: Compute $\widehat{\mathtt{q}}^{\mathrm{U}}_{\mathcal{M}}(\boldsymbol{\mu}) - \widehat{\mathtt{q}}^{\mathrm{L}}_{\mathcal{M}}(\boldsymbol{\mu})$ and determine the largest intervals $[a_i, b_i]$ such that $|\widehat{\mathtt{q}}^{\mathrm{U}}_{\mathcal{M}}(b_i) - \widehat{\mathtt{q}}^{\mathrm{L}}_{\mathcal{M}}(a_i)| \le \varepsilon$.

Remark 9.3.1. Recall that $\boldsymbol{x}^T P_\mu \boldsymbol{x}$ is a staircase-like piecewise constant function of μ, and in particular it must be constant in any interval that contains no eigenvalues of A. This means that given a candidate gap $[\mu_1, \mu_2]$, in addition to the condition $|\widehat{\mathtt{q}}^{\mathrm{U}}_{\mathcal{M}}(\mu_2) - \widehat{\mathtt{q}}^{\mathrm{L}}_{\mathcal{M}}(\mu_1)| \le \varepsilon$, we also need to check that $\widehat{\mathtt{q}}^{\mathrm{L}}_{\mathcal{M}}(\mu_2) < \widehat{\mathtt{q}}^{\mathrm{U}}_{\mathcal{M}}(\mu_1)$. If this additional condition is not satisfied, it is impossible for $\boldsymbol{x}^T P_\mu \boldsymbol{x}$ to be a constant function for $\mu \in [\mu_1, \mu_2]$, and hence we can conclude with certainty that there must be at least one eigenvalue of A in the interval $[\mu_1, \mu_2]$. This seems to cause a contradiction when the condition $|\widehat{\mathtt{q}}^{\mathrm{U}}_{\mathcal{M}}(\mu_2) - \widehat{\mathtt{q}}^{\mathrm{L}}_{\mathcal{M}}(\mu_1)| \le \varepsilon$ is verified but $\widehat{\mathtt{q}}^{\mathrm{L}}_{\mathcal{M}}(\mu_2) < \widehat{\mathtt{q}}^{\mathrm{U}}_{\mathcal{M}}(\mu_1)$ is not, since it would imply that all the eigenvalues of A contained in $[\mu_1, \mu_2]$ have an associated ε-small jump, which is a low-probability event. However, it simply means that this situation is very unlikely to happen in a practical scenario, and it would probably not be encountered if the algorithm is run on the same problem for a second time.

9.3.4 Computational cost

The most expensive operations in Algorithm 9.2 are the m matrix-vector products with A required to construct the Krylov basis in the Lanczos algorithm, which cost $O(m\,\mathtt{nnz}(A))$ for a sparse matrix A. The additional operations done for the computation of the quadratic forms $\boldsymbol{x}^T P_{\mu_j} \boldsymbol{x}$ with N_f different parameters μ_j add up to $O(nm + m^3 + mN_f)$, as discussed in Section 9.1.1. If we estimate the error with the consecutive difference estimate, we have to run the Lanczos algorithm for one additional iteration, and compute the differences $|\psi_{m',j} - \psi_{m'+1,j}|$ for all $m' \in \mathcal{M}$. If $|\mathcal{M}| = d$ and the elements of $\mathcal{M}$ are consecutive integers, this amounts to computing $d+1$ quadratic forms $\psi_{m',j}$ for all j, so the additional cost is $O(m^3 d + mN_f d)$.

On the other hand, if we use the a posteriori error bound from Proposition 9.2.7, we have to evaluate the coefficients α_j, β_j and γ_j for all μ_j, which can be done with $O(m^2)$ operations assuming that the eigenvalue decomposition of T_m is available, and compute the maximum of $|g_m(z)|$ over $[\lambda_{\min}, \lambda_{\max}]$, where g_m is defined in (9.2.10). The evaluation of $g_m(z)$ for all μ_j at a single point z requires $O(mN_f)$ operations, and if a discretization of $[\lambda_{\min}, \lambda_{\max}]$ with L points is used, we can approximately evaluate the bound in Proposition 9.2.7 for all μ_j with $O(m^2 + mN_f L)$ operations. If a set $\mathcal{M}$ with d elements

is used to compute the more robust bounds $\widehat{\mathtt{q}}^{\mathrm{U}}_{\mathcal{M}}(\mu_j)$ and $\widehat{\mathtt{q}}^{\mathrm{L}}_{\mathcal{M}}(\mu_j)$, the cost for the computation of the bounds is roughly multiplied by a factor d and becomes $O(m^3 d + mN_f Ld)$, where we have also included the cost of computing the eigenvalues and eigenvectors of $T_{m'}$ for $m' \in \mathcal{M}$. Note that given $\mathtt{q}^{\mathrm{U}}_m(\mu_j)$ and $\mathtt{q}^{\mathrm{L}}_m(\mu_j)$, the refined bounds $\widehat{\mathtt{q}}^{\mathrm{U}}_m(\mu_j)$ and $\widehat{\mathtt{q}}^{\mathrm{L}}_m(\mu_j)$ described in Section 9.3.2 can be computed for all μ_j with $O(N_f)$ operations. Finally, once $\widehat{\mathtt{q}}^{\mathrm{U}}_{\mathcal{M}}(\mu_j)$ and $\widehat{\mathtt{q}}^{\mathrm{L}}_{\mathcal{M}}(\mu_j)$ have been computed, the gaps can be located with a cost of $O(N_f)$. The total computational cost of Algorithm 9.2 is therefore $O(m\,\mathtt{nnz}(A) + nm + m^3 d + mN_f d)$ with the consecutive difference estimate, and $O(m\,\mathtt{nnz}(A) + nm + m^3 d + mN_f Ld)$ with the a posteriori bound from Proposition 9.2.7.

Note that all operations associated with different values of μ are completely independent, so the algorithm can be easily parallelized, for instance by assigning different slices of the spectral interval to different processors. This parallelization is not particularly helpful when the matrix size n is large enough, since the dominant part of the computation would still be the $O(m\,\mathtt{nnz}(A))$ term associated with matrix-vector products with A; however, parallelizing the computations associated with different parameters μ can be helpful for mitigating the $O(mN_f Ld)$ cost of computing the a posteriori error bounds from Proposition 9.2.7, which can become quite high if both N_f and L are large, see for instance the experiment in Section 9.4.3.

If our goal is to detect all gaps with relative width larger than θ with probability at least $1 - \delta$, we should take m according to (9.3.1) with $\varepsilon \leq e^{-1}\delta^2$, i.e.,

$$m \geq 1 + \frac{1 + \log\left(2C\|\boldsymbol{x}\|_2^2/\delta^2\right)}{\log\left((1+\theta)/(1-\theta)\right)}, \qquad C = \frac{1-\theta}{\sqrt{\pi\theta}} + 1, \tag{9.3.5}$$

where we recall that $\mathbb{E}[\|\boldsymbol{x}\|_2^2] = n$. By looking at the behavior of the right-hand side of (9.3.5) as $\theta \to 0$, using the fact that

$$\log\left(\frac{1+\theta}{1-\theta}\right) \approx \frac{1}{2\theta},$$

we conclude that for small θ we should take

$$m \approx \log\left(\frac{2\|\boldsymbol{x}\|_2^2}{\delta^2\sqrt{\pi\theta}}\right)\frac{1}{2\theta}.$$

Although the dependence of m on the failure probability δ and the matrix size n is not particularly severe since they appear inside a logarithm, the main drawback of this approach is the $O(1/\theta)$ dependence on the relative gap width θ, which causes Algorithm 9.2 to quickly become more expensive as the target gap width is reduced. This is an inherent limitation of the Lanczos method for the computation of $\boldsymbol{x}^T P_\mu \boldsymbol{x}$, and more precisely of the polynomial approximation of the sign function (see Proposition 9.2.3). As we mentioned in Section 9.2.3, Proposition 9.2.3 can be improved by considering two nonsymmetric intervals as in [66], and this would in turn lead to an improved estimate on the required number of Lanczos iterations m; however, in general we do not expect to find an asymptotic estimate that is better than $m = O(1/\theta)$. Indeed, for a gap located near the center of $\Lambda(A)$, the spectrum of A is contained in two intervals that are approximately symmetric with respect to the gap, and there would not be any significant advantage in using the improved result [66, Theorem 1.1].

Example 9.3.1. In this example we show how the bounds described in Section 9.3.2 perform on the simple test problem from Example 9.2.1. The matrix A is 600×600, with $\Lambda(A) \subset [0, 60]$ and three gaps in the spectrum given by the intervals $[20, 21]$, $[30, 32]$ and $[40, 44]$. We use a single Gaussian vector $\boldsymbol{x} \in \mathbb{R}^n$ for Hutchinson's estimator and $m = 50$ Lanczos iterations. We compare the exact $\boldsymbol{x}^T P_\mu \boldsymbol{x}$ with the upper and lower bounds $\widehat{\mathtt{q}}^{\mathrm{U}}_{\mathcal{M}}(\mu)$ and $\widehat{\mathtt{q}}^{\mathrm{L}}_{\mathcal{M}}(\mu)$ obtained both with Proposition 9.2.7 and with the consecutive difference error estimate. In particular, for the latter we use the error estimate $\mathtt{bound}_m(\mu_j) = c|\psi_{m,j} - \psi_{m+1,j}|$, with a safety factor $c = 2$. In both cases, we take $\mathcal{M} = \{m - d + 1, \ldots, m\}$ with $d = 3$. Since $c|\psi_{m,j} - \psi_{m+1,j}|$ is not guaranteed to be an upper bound for $|\boldsymbol{x}^T P_{\mu_j} \boldsymbol{x}|$, the inequalities $\widehat{\mathtt{q}}^{\mathrm{L}}_{\mathcal{M}}(\mu) \leq \boldsymbol{x}^T P_\mu \boldsymbol{x} \leq \widehat{\mathtt{q}}^{\mathrm{U}}_{\mathcal{M}}(\mu)$ do not necessarily hold for all μ, especially after the two minimizations and maximizations in (9.3.3) and (9.3.4). To overcome this issue, we also propose to use a "safe" variant of the upper and lower estimates for $\boldsymbol{x}^T P_\mu \boldsymbol{x}$, defined as

$$\widehat{\mathtt{q}}^{\mathrm{U}\star}_{\mathcal{M}}(\mu) := \max_{m \in \mathcal{M}} \widehat{\mathtt{q}}^{\mathrm{U}}_m(\mu) \qquad \text{and} \qquad \widehat{\mathtt{q}}^{\mathrm{L}\star}_{\mathcal{M}}(\mu) := \min_{m \in \mathcal{M}} \widehat{\mathtt{q}}^{\mathrm{L}}_m(\mu), \tag{9.3.6}$$

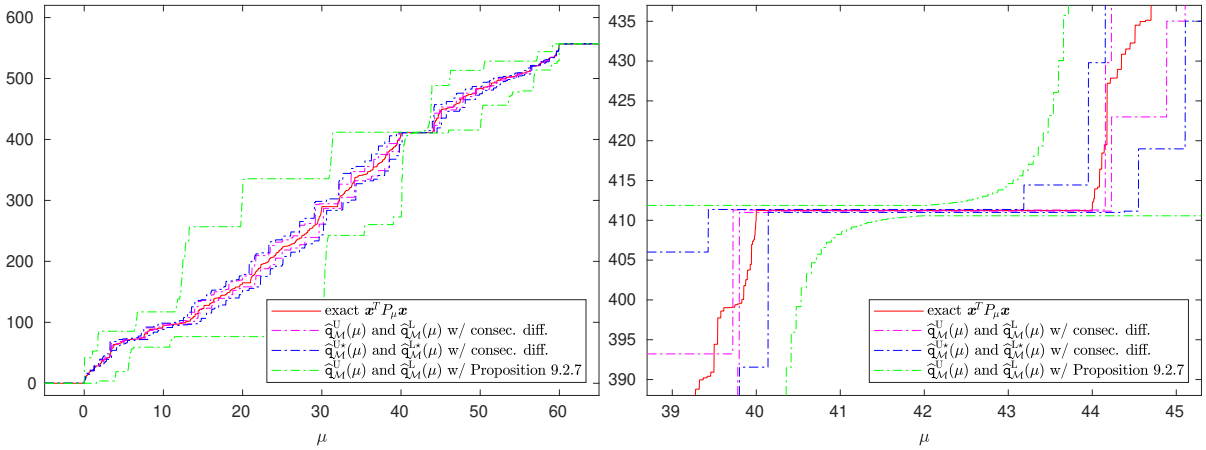


Figure 9.3 – Upper and lower bounds for $\boldsymbol{x}^T P_\mu \boldsymbol{x}$ obtained with the Lanczos algorithm with $m = 50$ iterations. The right plot is a close-up of the left plot around the larger gap $[40, 44]$.

which correspond to taking the worst of the bounds associated with $\mathcal{M}$ instead of the best ones. This variant significantly increases the chance that $\widehat{\mathtt{q}}^{\mathrm{U}\star}_{\mathcal{M}}(\mu)$ and $\widehat{\mathtt{q}}^{\mathrm{L}\star}_{\mathcal{M}}(\mu)$ are an upper and lower bound for $\boldsymbol{x}^T P_\mu \boldsymbol{x}$, respectively, although it reduces the overall accuracy of the estimates.

The bounds are shown in Figure 9.3, including the safe variants $\widehat{\mathtt{q}}^{\mathrm{U}\star}_{\mathcal{M}}(\mu)$ and $\widehat{\mathtt{q}}^{\mathrm{L}\star}_{\mathcal{M}}(\mu)$ of the bounds obtained by using consecutive differences. Notice that the upper and lower estimates $\widehat{\mathtt{q}}^{\mathrm{U}}_{\mathcal{M}}(\mu)$ and $\widehat{\mathtt{q}}^{\mathrm{L}}_{\mathcal{M}}(\mu)$ computed with consecutive differences are much closer to $\boldsymbol{x}^T P_\mu \boldsymbol{x}$ compared to the bounds obtained with Proposition 9.2.7, but they sometimes fail to satisfy $\widehat{\mathtt{q}}^{\mathrm{L}}_{\mathcal{M}}(\mu) \leq \boldsymbol{x}^T P_\mu \boldsymbol{x} \leq \widehat{\mathtt{q}}^{\mathrm{U}}_{\mathcal{M}}(\mu)$: see for instance the right plot in Figure 9.3, which is a close-up of the larger gap $[40, 44]$. On the other hand, the safe variants $\widehat{\mathtt{q}}^{\mathrm{U}\star}_{\mathcal{M}}(\mu)$ and $\widehat{\mathtt{q}}^{\mathrm{L}\star}_{\mathcal{M}}(\mu)$ are more robust, while still being more accurate than the a posteriori bounds from Proposition 9.2.7. To provide a quantitative comparison, for the plot in Figure 9.3 we used 4000 different shift parameters μ_j, and the inequality $\boldsymbol{x}^T P_{\mu_j} \boldsymbol{x} \leq \widehat{\mathtt{q}}^{\mathrm{U}}_{\mathcal{M}}(\mu_j)$ was not satisfied for 251 of them, while $\boldsymbol{x}^T P_{\mu_j} \boldsymbol{x} \leq \widehat{\mathtt{q}}^{\mathrm{U}\star}_{\mathcal{M}}(\mu_j)$ was not satisfied for only 35 values of j. Note that the fraction of values of μ for which the safe upper estimate $\widehat{\mathtt{q}}^{\mathrm{U}\star}_{\mathcal{M}}(\mu)$ fails to be an upper bound is smaller than 10^{-2}, which is the failure probability that we use for the experiments in Section 9.4.

9.4 Numerical experiments

In this section we run some experiments to investigate the performance of Algorithm 9.2. For all the experiments, we use Algorithm 9.2 with a failure probability $\delta = 10^{-2}$ and a number of Lanczos iterations m that is either fixed in advance or computed according to (9.3.5) in order to find all gaps with relative width larger than a certain target θ. When not stated differently, for the Lanczos algorithm we employ the error estimate $\mathtt{bound}_m(\mu_j) = c|\psi_{m,j} - \psi_{m,j+1}|$ with a safety factor $c = 2$, and we use the more robust upper and lower estimates $\widehat{\mathtt{q}}^{\mathrm{U}\star}(\mu)$ and $\widehat{\mathtt{q}}^{\mathrm{L}\star}(\mu)$ defined in (9.3.6), with $\mathcal{M} = \{m - d + 1, \ldots, m\}$ with $d = 3$. These estimates are cheaper to compute compared to the a posteriori error bound from Proposition 9.2.7, and they are also quite reliable, as shown in Example 9.3.1. Even though the estimate $c|\psi_{j,m} - \psi_{j,m+1}|$ does not have the same theoretical guarantees as the a posteriori error bound, we have seen in Example 9.2.2 that it is usually much closer to the actual error. All the experiments have been done using MATLAB R2021b on a laptop running Ubuntu 20.04, with 32 GB of RAM and an Intel Core i5-10300H CPU with clock rate 2.5 GHz. The code for reproducing the experiments in this section is available on Github at `https://github.com/simunec/phd-thesis-code`.

9.4.1 Scaling with gap width

We start by examining the behavior of Algorithm 9.2 with respect to the relative gap width θ. We consider a symmetric tridiagonal matrix A of size $n = 30000$ obtained as a sum $A = D + T$, where T is a

Table 9.1 – Performance of Algorithm 9.2 with $n = 30000$ and varying gap width θ.

θ	true gap	estim. gap	estim. $\mathrm{n_e}(\mu)$	Lanczos its.	time (s)	time `eig` (s)
0.1	[1001.80, 2633.87]	[1016.48, 2637.19]	19 637	112	0.114	2.985
0.05	[1001.65, 1856.42]	[999.77, 1856.64]	19 776	226	0.209	2.798
0.025	[1001.15, 1435.81]	[1001.61, 1434.55]	20 198	456	0.359	2.907
0.01	[1001.17, 1175.30]	[1001.61, 1174.65]	20 027	1156	1.025	2.645
0.005	[1000.69, 1085.83]	[1000.69, 1085.18]	19 862	2342	2.451	2.909
0.0025	[1003.69, 1042.75]	[1004.38, 1042.08]	19 709	4745	7.160	2.924

random symmetric tridiagonal matrix with $\mathcal{N}(0,1)$ entries and D is a diagonal matrix. The eigenvalues of D are chosen so that the spectrum of A has a gap with relative width approximately θ. Specifically, the eigenvalues of D are contained in the intervals $[1, 10^3] \cup [10^3 + x, 10^4]$, where x satisfies

$$\frac{x/2}{10^4 - 10^3 - x/2} = \theta,$$

so that the relative width of the gap $[10^3, 10^3 + x]$ is θ, and hence the spectrum of the perturbed matrix $A = D + T$ also has a gap with relative width approximately θ. We consider a matrix D with 20000 logspaced eigenvalues in $[1, 10^3]$ and 10000 logspaced eigenvalues in $[10^3 + x, 10^4]$.

We estimate this gap using Algorithm 9.2, with m computed according to (9.3.5), so that we can expect it to successfully find all gaps with relative width larger than θ; we use $N_f = 10000$ parameters μ, logarithmically spaced on the interval $[1, 10^4]$. For each θ, in Table 9.1 we show the true gap computed by diagonalizing A, the gap estimated by Algorithm 9.2 and the approximate number of eigenvalues below the gap, obtained by rounding the trace approximation $\boldsymbol{x}^T P_\mu \boldsymbol{x}$ for μ equal to the left endpoint of the estimated gap, as well as the number of Lanczos iterations m, the execution time of the algorithm and the time required for the computation of the eigenvalues of A. We see that the gap is found successfully for all θ, with the estimated gap being slightly larger than the true gap only for $\theta = 0.1$ and $\theta = 0.05$. The estimated number of eigenvalues below the gap roughly approximates the exact value 20000, but it is still relatively far from the correct number since we are using only one sample vector for Hutchinson's estimator. Note that the number of Lanczos iterations scales approximately as $O(1/\theta)$, but the execution time increases more than linearly in the number of Lanczos iterations m; this is likely due to the $O(m^3)$ cost for the computation of eigenvalues and eigenvectors of the projected matrix T_m becoming the dominant part of the computation when the subspace dimension m is large. On the other hand, the execution time for `eig` remains constant, so when the gap width θ becomes sufficiently small it would be more efficient to directly diagonalize the matrix instead of using Algorithm 9.2. Of course, this would not be the case if the matrix dimension n increases, as we show in the following experiment.

9.4.2 Scaling with matrix size

Let us now consider a sequence of matrices with increasing size and approximately constant gap width. We use a setup similar to Section 9.4.2, with fixed relative gap width $\theta = 0.01$ and increasing matrix size n. We take $A = D + T$, where T is symmetric tridiagonal with random $\mathcal{N}(0,1)$ entries and D is diagonal with $\Lambda(A) \subset [1, 10^3] \cup [10^3 + x, 10^4]$, with $n/2$ logspaced eigenvalues below the gap and $n/2$ logspaced eigenvalues above the gap. The value of x is chosen so that A has relative gap width approximately θ, as in Section 9.4.2.

We use Algorithm 9.2 to estimate the gap, with the number of Lanczos iterations m computed according to (9.3.5) and $N_f = 10000$ parameters μ, logarithmically spaced over the interval $[1, 10^4]$. The results for a matrix dimension that increases from $n = 5000$ to $n = 80000$ are shown in Table 9.2. In this case, the gap is found correctly for all matrix sizes, with none of the estimated gaps being larger than the true gap computed via a diagonalization. Note that the number of Lanczos iterations increases very slowly with the matrix dimension, in accordance with the discussion in Section 9.3.4. Hence, the execution time of the algorithm increases roughly linearly, since with a large matrix size n and moderate subspace dimension m the leading term in the computational cost is given by the m matrix-vector products with A in the construction of the Lanczos basis, which cost $O(mn)$ for a tridiagonal

Table 9.2 – Performance of Algorithm 9.2 with fixed gap width $\theta = 0.01$ and increasing matrix size n.

n	true gap	estim. gap	estim. $\mathrm{n_e}(\mu)$	Lanczos its.	time (s)	time `eig` (s)
5000	[1000.58, 1177.15]	[1000.69, 1176.81]	2377	1067	0.412	0.115
10 000	[1001.11, 1175.88]	[1001.61, 1175.73]	4956	1101	0.548	0.378
20 000	[1000.23, 1177.01]	[1000.69, 1176.81]	9835	1136	0.806	1.355
40 000	[1002.67, 1175.70]	[1003.46, 1174.65]	19 930	1171	1.236	4.760
80 000	[1001.39, 1176.18]	[1001.61, 1175.73]	39 874	1205	2.237	16.337

Table 9.3 – Performance of Algorithm 9.2 with fixed gap width $\theta = 0.01$ and increasing matrix size n. The number of Lanczos iterations is fixed to $m = 250$ and the failure probability is set to $\delta = 0.5$.

n	true gap	estim. gap	estim. $\mathrm{n_e}(\mu)$	Lanczos its.	time (s)	time `eig` (s)
5000	[1000.58, 1177.15]	[998.85, 1178.98]	2377	250	0.102	0.114
10 000	[1001.11, 1175.88]	[1003.46, 1170.33]	4956	250	0.123	0.386
20 000	[1000.23, 1177.01]	[1001.61, 1167.10]	9835	250	0.188	1.353
40 000	[1002.67, 1175.70]	[1002.54, 1161.73]	19 930	250	0.281	4.757
80 000	[1001.39, 1176.18]	[1005.31, 1175.73]	39 874	250	0.467	16.454

matrix. On the other hand, the time required for computing the eigenvalues of the tridiagonal matrix A scales like $O(n^2)$, so it quickly becomes more expensive than Algorithm 9.2. Note that if A was a sparse matrix that is not tridiagonal, the diagonalization time would scale like $O(n^3)$, making the difference between the two approaches even more evident.

It is interesting to observe that Algorithm 9.2 can be also used to cheaply obtain a rough estimate of the gap by taking a larger failure probability δ and fixing the number of Lanczos iterations m in advance instead of computing it with (9.3.5). For instance, we show in Table 9.3 the performance of Algorithm 9.2 with $\delta = 0.5$ and $m = 250$. We see that the gap in the spectrum is still located successfully, although the accuracy is generally lower and sometimes the estimated gap is larger than the true gap, for instance in the case $n = 5000$. Note that the estimated number of eigenvalues below the gap coincides with the corresponding number from Table 9.2, even if the number of Lanczos iterations is different: this happens because we used the same random vector $\boldsymbol{x}$ for Hutchinson's trace estimator, and in both cases we are approximating the same quadratic form $\boldsymbol{x}^T P_\mu \boldsymbol{x}$ (the values of μ in the two cases may be different, but as long as they are both inside the gap the corresponding values of $\boldsymbol{x}^T P_\mu \boldsymbol{x}$ coincide). In particular, this also confirms that the error in the estimated number of eigenvalues below the gap is essentially only due to the error in the stochastic trace estimator, and not to the error in the approximation of the quadratic form via the Lanczos algorithm.

9.4.3 A real-world example

We conclude the experiments in this chapter with an example using a matrix from the SuiteSparse Matrix Collection [51]. We take as A the symmetric matrix `GHS_indef/linverse`, which has size $n = 11999$ with 95977 nonzero entries. The spectrum of this matrix, shown in Figure 9.4, has three relatively large gaps, which we aim to locate using Algorithm 9.2. In this test we compare the robust consecutive difference estimates defined in (9.3.6) with the a posteriori bound from Proposition 9.2.7, both obtained using $m = 100$ Lanczos iterations and $d = 3$. We plot in Figure 9.5 the approximations to $\boldsymbol{x}^T P_\mu \boldsymbol{x}$ and the upper and lower bounds obtained with the two approaches, where $\boldsymbol{x}$ is the single random vector used in Hutchinson's estimator; we use the same vector $\boldsymbol{x}$ for the two variants of Algorithm 9.2. Note that the upper and lower bounds obtained by using Proposition 9.2.7 are much more conservative compared to those based on consecutive difference estimates, especially far from the gaps in the spectrum.

The gaps found by the two variants of Algorithm 9.2 and the exact gaps obtained by diagonalizing A are shown in Table 9.4, where we also include the execution times for all the methods. The gaps found by using the a posteriori bounds from Proposition 9.2.7 are significantly smaller than the ones obtained

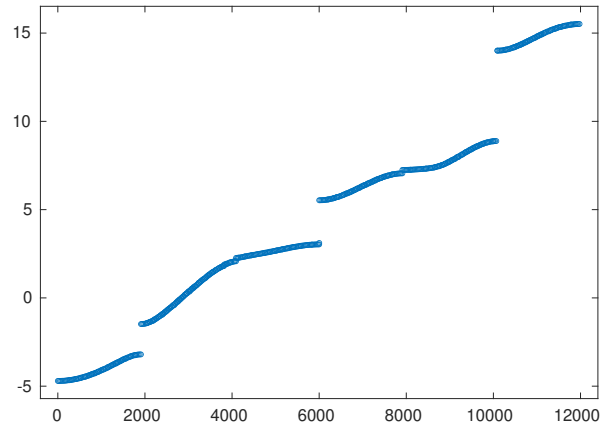


Figure 9.4 – Spectrum of the matrix `GHS_indef/linverse`, with three evident gaps.

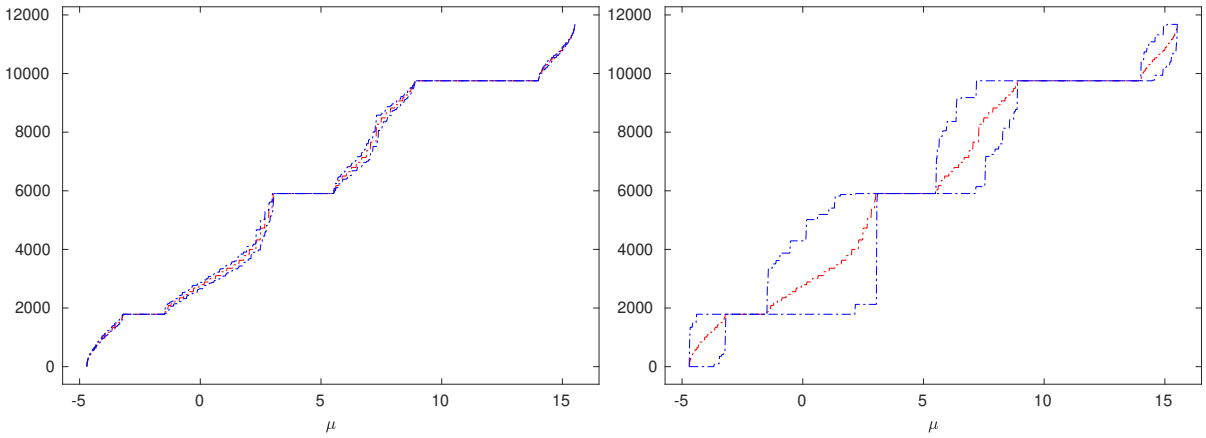


Figure 9.5 – Approximations to $\boldsymbol{x}^T P_\mu \boldsymbol{x}$, and upper and lower bounds obtained with Algorithm 9.2, for several different μ, with $m = 250$ Lanczos iterations. Left: Algorithm 9.2 with robust consecutive difference error estimate (9.3.6). Right: Algorithm 9.2 with a posteriori error bound from Proposition 9.2.7.

Table 9.4 – Performance of Algorithm 9.2 on the test matrix `GHS_indef/linverse`, using the robust consecutive difference estimate (9.3.6) and the a posteriori bound from Proposition 9.2.7. The number of Lanczos iterations is set to $m = 100$ and the failure probability to $\delta = 10^{-2}$.

method	first gap	second gap	third gap	time (s)
`eig`	[-3.201, -1.482]	[3.112, 5.531]	[8.886, 14.007]	18.826
Algorithm 9.2 w/ consec. diff.	[-3.185, -1.485]	[3.352, 5.518]	[8.979, 13.999]	0.030
Algorithm 9.2 w/ Proposition 9.2.7	[-2.963, -1.748]	[3.474, 5.154]	[8.999, 13.918]	0.375

by estimating the Lanczos error with consecutive differences, especially for the two smaller gaps; this is a consequence of the lower accuracy of the a posteriori bounds. Moreover, the execution time is quite higher with the a posteriori bounds, due to the $O(mN_fLd)$ term in the computation of the error bounds from Proposition 9.2.7; indeed, it turns out that this is the most expensive part of the computation in this setting, where we use $N_f = 1000$ different values of μ and a discretization with $L = 1000$ points to evaluate the maximum of $|g_m(z)|$ over the spectral interval of A. On the other hand, when the error in the Lanczos algorithm is estimated using consecutive differences, the portion of the computational cost that depends on N_f scales as $O(mN_fd)$, which is significantly less expensive when $L = 1000$ (see Section 9.3.4).

Chapter 10

Conclusions and outlook

In this thesis, we have explored the computation of the action of a matrix function $f(A)$ on a vector $\boldsymbol{b}$ using polynomial and rational Krylov subspace methods. Our analysis has covered both theoretical and computational perspectives, and we have additionally investigated the application of Krylov subspace methods to trace approximation problems encountered in various fields.

From a theoretical side, we have derived both a priori and a posteriori error bounds for the approximation of $f(A)\boldsymbol{b}$ and $\boldsymbol{b}^T f(A)\boldsymbol{b}$ using rational Krylov subspace methods. These error bounds, obtained by exploiting an integral representation of the function f, can be valuable as stopping criteria or for predicting the number of iterations required to achieve a desired accuracy. Moreover, we have introduced rational Krylov methods for the computation of generalized matrix functions, extending an approach based on the Golub-Kahan bidiagonalization and providing a convergence analysis that directly generalizes the rational Krylov theory for standard matrix functions. We have also shown how to exploit the quasiseparable structure of the projected matrices to construct the rational Krylov bases with a short-term recurrence.

From a computational standpoint, we have developed a novel low-memory method for computing $f(A)\boldsymbol{b}$ when A is Hermitian. This method combines an outer Lanczos iteration with an inner rational Krylov subspace for basis compression. A key feature of this method is the fact that the inner rational Krylov subspace is constructed using small projected matrices, and hence it does not involve any expensive operations with the matrix A. This approach essentially retains the same convergence properties of the outer Lanczos iteration, and it exhibits competitive performance when compared against existing low-memory Krylov subspace methods for $f(A)\boldsymbol{b}$.

We have also employed a rational Krylov method to solve fractional diffusion equations on graphs. In this setting, the matrix A is a singular graph Laplacian, and the function f is not analytic in a neighborhood of the spectrum of A, potentially leading to convergence issues in Krylov subspace methods. To address this problem, we have developed three different approaches to remove the singularity of the graph Laplacian: a rank-one modification, an explicit projection, and an implicit projection. Each of these techniques provides the same improvement in the convergence rate of Krylov subspace methods, and among them, the implicit projection is the cheapest one, with practically no additional cost compared to a standard implementation of a Krylov subspace method. The same technique can also be applied to improve convergence for other problems where a known eigenvalue of A is close to a singularity of f.

Next, we have applied Krylov subspace methods in combination with trace estimators to approximate $\operatorname{tr}(f(A))$ for two different problems: the computation of the von Neumann entropy and the location of spectral gaps. For the von Neumann entropy, we have analyzed the convergence of the probing method, providing both theoretical bounds and heuristic estimates for the error. By integrating this analysis with an a posteriori error bound for the Krylov subspace method, we have obtained an algorithm for computing the von Neumann entropy to a specified tolerance ε, employing either a probing method or an adaptive implementation of the stochastic trace estimator Hutch++.

For the identification of spectral gaps, we have analyzed an algorithm that approximates the traces of spectral projectors P_μ by combining Hutchinson's trace estimator and the Lanczos algorithm for

computing quadratic forms. Through a rigorous analysis of the algorithm's components, we have determined that the most efficient approach involves using a single random vector for Hutchinson's estimator. Additionally, we have established criteria for choosing the number of Lanczos iterations needed to ensure the detection of all spectral gaps wider than a certain threshold.

Several possible directions could be considered for future research. For instance, although the error bounds derived in Chapter 3 rely on an error expression that also holds for non-Hermitian matrices, the bounds themselves are stated only for the Hermitian case. The extension of these bounds to the non-Hermitian case remains an open challenge, with difficulties that arise due to the replacement of the spectrum with the field of values or other spectral sets [48]. It would be valuable to investigate whether similar approaches can also yield effective bounds in the non-Hermitian case, and to evaluate their numerical accuracy against the error. Another related topic with practical relevance is a numerical comparison of the various error bounds and estimates for polynomial and rational Krylov methods found in the literature, including those presented in this thesis. Such a study would provide useful insight for identifying which bounds are the most accurate under different conditions, such as the properties of the function f and the matrix A. Such a comparison should also take into account the potential difference in computational cost associated with the application of each bound, in order to evaluate the practical utility of the different bounds.

The low-memory Krylov subspace method presented in Chapter 5 can only be used for Hermitian A, as it exploits a low-rank structure of the projection of A onto the Krylov subspace that is only present when A is Hermitian. An extension of this approach to the general non-Hermitian setting appears challenging, due to the loss of this low-rank structure. However, such an extension may be feasible with suitable modifications to the algorithm, and could potentially have a large practical impact, even if it resulted in an algorithm with fewer theoretical guarantees compared to the Hermitian case.

Finally, the approach for identifying spectral gaps analyzed in Chapter 9 appears to be quite promising, and it should be tested on practical problems, such as those encountered in electronic structure computations, to evaluate its effectiveness compared to state-of-the-art techniques.

Bibliography

[1] J. Aarons and C.-K. Skylaris, *Electronic annealing Fermi operator expansion for DFT calculations on metallic systems*, J. Chem. Phys., 148 (2018), 074107.

[2] J. Alahmadi, M. Pranić, and L. Reichel, *Rational Gauss quadrature rules for the approximation of matrix functionals involving Stieltjes functions*, Numer. Math., 151 (2022), pp. 443–473.

[3] F. Andersson, M. Carlsson, and K.-M. Perfekt, *Operator-Lipschitz estimates for the singular value functional calculus*, Proc. Amer. Math. Soc., 144 (2016), pp. 1867–1875.

[4] F. Arrigo and M. Benzi, *Edge modification criteria for enhancing the communicability of digraphs*, SIAM J. Matrix Anal. Appl., 37 (2016), pp. 443–468.

[5] F. Arrigo, M. Benzi, and C. Fenu, *Computation of generalized matrix functions*, SIAM J. Matrix Anal. Appl., 37 (2016), pp. 836–860.

[6] J. L. Aurentz, A. P. Austin, M. Benzi, and V. Kalantzis, *Stable computation of generalized matrix functions via polynomial interpolation*, SIAM J. Matrix Anal. Appl., 40 (2019), pp. 210–234.

[7] B. Beckermann, *An error analysis for rational Galerkin projection applied to the Sylvester equation*, SIAM J. Numer. Anal., 49 (2011), pp. 2430–2450.

[8] B. Beckermann, J. Bisch, and R. Luce, *On the rational approximation of Markov functions, with applications to the computation of Markov functions of Toeplitz matrices*, Numer. Algorithms, 91 (2022), pp. 109–144.

[9] B. Beckermann, A. Cortinovis, D. Kressner, and M. Schweitzer, *Low-rank updates of matrix functions II: rational Krylov methods*, SIAM J. Numer. Anal., 59 (2021), pp. 1325–1347.

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

[11] I. Bengtsson and K. Zyczkowski, *Geometry of Quantum States: An Introduction to Quantum Entanglement*, Cambridge University Press, 2006.

[12] M. Benzi, *Localization in Matrix Computations: Theory and Applications*, in Exploiting Hidden Structure in Matrix Computations: Algorithms and Applications, vol. 2173 of Lecture Notes in Math., 2173, Fond. CIME/CIME Found. Subser., Springer, Cham, 2016, pp. 211–317.

[13] M. Benzi, D. Bertaccini, F. Durastante, and I. Simunec, *Non-local network dynamics via fractional graph Laplacians*, J. Complex Netw., 8 (2020), cnaa017.

[14] M. Benzi, P. Boito, and N. Razouk, *Decay properties of spectral projectors with applications to electronic structure*, SIAM Rev., 55 (2013), pp. 3–64.

[15] M. Benzi, E. Estrada, and C. Klymko, *Ranking hubs and authorities using matrix functions*, Linear Algebra Appl., 438 (2013), pp. 2447–2474.

[16] M. Benzi, P. Fika, and M. Mitrouli, *Graphs with absorption: numerical methods for the absorption inverse and the computation of centrality measures*, Linear Algebra Appl., 574 (2019), pp. 123–152.

[17] M. Benzi and G. H. Golub, *Bounds for the entries of matrix functions with applications to preconditioning*, BIT, 39 (1999), pp. 417–438.

[18] M. Benzi and R. Huang, *Some matrix properties preserved by generalized matrix functions*, Spec. Matrices, 7 (2019), pp. 27–37.

[19] M. Benzi and M. Rinelli, *Refined decay bounds on the entries of spectral projectors associated with sparse Hermitian matrices*, Linear Algebra Appl., 647 (2022), pp. 1–30.

[20] M. Benzi, M. Rinelli, and I. Simunec, *Spectral gap estimation for sparse symmetric matrices*, In preparation, 2024.

[21] ——, *Computation of the von Neumann entropy of large matrices via trace estimators and rational Krylov methods*, Numer. Math., 155 (2023), pp. 377–414.

[22] M. Benzi and V. Simoncini, *Decay bounds for functions of Hermitian matrices with banded or Kronecker structure*, SIAM J. Matrix Anal. Appl., 36 (2015), pp. 1263–1282.

[23] M. Benzi and I. Simunec, *Rational Krylov methods for fractional diffusion problems on graphs*, BIT, 62 (2022), pp. 357–385.

[24] C. Berg, *Stieltjes-Pick-Bernstein-Schoenberg and their connection to complete monotonicity*, Positive definite functions: From Schoenberg to space-time challenges, (2008), pp. 15–45.

[25] M. Berljafa, S. Elsworth, and S. Güttel, *A rational Krylov toolbox for MATLAB*, MIMS EPrint 2014.56, Manchester Institute for Mathematical Sciences, University of Manchester, Manchester, UK, (2014).

[26] M. Berljafa and S. Güttel, *Generalized rational Krylov decompositions with an application to rational approximation*, SIAM J. Matrix Anal. Appl., 36 (2015), pp. 894–916.

[27] ——, *Parallelization of the rational Arnoldi algorithm*, SIAM J. Sci. Comput., 39 (2017), pp. S197–S221.

[28] A. Berman and R. J. Plemmons, *Nonnegative Matrices in the Mathematical Sciences*, vol. 9 of Classics in Applied Mathematics, SIAM, Philadelphia, PA, 1994. Revised reprint of the 1979 original.

[29] Å. Björck, *Numerical Methods in Matrix Computations*, vol. 59 of Texts in Applied Mathematics, Springer, Cham, 2015.

[30] P. Bochev and R. B. Lehoucq, *On the finite element solution of the pure Neumann problem*, SIAM Rev., 47 (2005), pp. 50–66.

[31] A. Boriçi, *Fast methods for computing the Neuberger operator*, in Numerical Challenges in Lattice Quantum Chromodynamics, A. Frommer, T. Lippert, B. Medeke, and K. Schilling, eds., Berlin, Heidelberg, 2000, Springer Berlin Heidelberg, pp. 40–47.

[32] S. L. Braunstein, S. Ghosh, and S. Severini, *The Laplacian of a graph as a density matrix: a basic combinatorial approach to separability of mixed states*, Ann. Comb., 10 (2006), pp. 291–317.

[33] K. Burrage, N. Hale, and D. Kay, *An efficient implicit FEM scheme for fractional-in-space reaction-diffusion equations*, SIAM J. Sci. Comput., 34 (2012), pp. A2145–A2172.

[34] D. Camps, K. Meerbergen, and R. Vandebril, *An implicit filter for rational Krylov using core transformations*, Linear Algebra Appl., 561 (2019), pp. 113–140.

[35] A. Carpenter, A. Ruttan, and R. Varga, *Extended numerical computations on the "1/9" conjecture in rational approximation theory*, in Rational Approximation and Interpolation: Proceedings of the United Kingdom-United States Conference held at Tampa, Florida, December 12–16, 1983, Springer, 1984, pp. 383–411.

[36] A. A. Casulli, D. Kressner, and L. Robol, *Computing functions of symmetric hierarchically semiseparable matrices*, arXiv:2402.17369, 2024.

[37] A. A. Casulli and I. Simunec, *Computation of generalized matrix functions with rational Krylov methods*, Math. Comp., 92 (2023), pp. 749–777.

[38] A. A. Casulli and I. Simunec, *A low-memory Lanczos method with rational Krylov compression for matrix functions*, arXiv:2403.04390, 2024.

[39] A. Chapman and M. Mesbahi, *Advection on graphs*, in 2011 50th IEEE Conference on Decision and Control and European Control Conference, 2011, pp. 1461–1466.

[40] T. Chen, A. Greenbaum, C. Musco, and C. Musco, *Error bounds for Lanczos-based matrix function approximation*, SIAM J. Matrix Anal. Appl., 43 (2022), pp. 787–811.

[41] ——, *Low-memory Krylov subspace methods for optimal rational matrix function approximation*, SIAM J. Matrix Anal. Appl., 44 (2023), pp. 670–692.

[42] T. Chen and E. Hallman, *Krylov-aware stochastic trace estimation*, SIAM J. Matrix Anal. Appl., 44 (2023), pp. 1218–1244.

[43] H. Choi, J. He, H. Hu, and Y. Shi, *Fast computation of von Neumann entropy for large-scale graphs via quadratic approximations*, Linear Algebra Appl., 585 (2020), pp. 127–146.

[44] C. K. Chui and M. Hasson, *Degree of uniform approximation on disjoint intervals*, Pacific J. Math., 105 (1983), pp. 291–297.

[45] A. Cortinovis and D. Kressner, *On randomized trace estimates for indefinite matrices with an application to determinants*, Found. Comput. Math., 22 (2022), pp. 875–903.

[46] M. Cramer and J. Eisert, *Correlations, spectral gap and entanglement in harmonic quantum systems on generic lattices*, New J. Phys., 8 (2006), 71.

[47] M. Crouzeix, *Numerical range and functional calculus in Hilbert space*, J. Funct. Anal., 244 (2007), pp. 668–690.

[48] M. Crouzeix and A. Greenbaum, *Spectral sets: numerical range and beyond*, SIAM J. Matrix Anal. Appl., 40 (2019), pp. 1087–1101.

[49] M. Crouzeix and C. Palencia, *The numerical range is a* $(1+\sqrt{2})$*-spectral set*, SIAM J. Matrix Anal. Appl., 38 (2017), pp. 649–655.

[50] E. Cuthill and J. McKee, *Reducing the bandwidth of sparse symmetric matrices*, in ACM '69: Proceedings of the 1969 24th National Conference, New York, NY, USA, 1969, pp. 157–172.

[51] T. A. Davis and Y. Hu, *The University of Florida Sparse Matrix Collection*, ACM Trans. Math. Software, 38 (2011).

[52] M. De Domenico and J. Biamonte, *Spectral entropies as information-theoretic tools for complex network comparison*, Phys. Rev. X, 6 (2016), 041062.

[53] K. Deckers and A. Bultheel, *Rational Krylov sequences and orthogonal rational functions*, Technical Report TW 499, (2007).

[54] S. Demko, W. F. Moss, and P. W. Smith, *Decay rates for inverses of band matrices*, Math. Comp., 43 (1984), pp. 491–499.

[55] E. Di Napoli, E. Polizzi, and Y. Saad, *Efficient estimation of eigenvalue counts in an interval*, Numer. Linear Algebra Appl., 23 (2016), pp. 674–692.

[56] R. Díaz Fuentes, M. Donatelli, C. Fenu, and G. Mantica, *Estimating the trace of matrix functions with application to complex networks*, Numer. Algorithms, 92 (2023), pp. 503–522.

[57] M. D. Domenico, V. Nicosia, A. Arenas, and V. Latora, *Structural reducibility of multilayer networks*, Nat. Commun., 6 (2015), 6864.

[58] T. A. Driscoll, N. Hale, and L. N. Trefethen, *Chebfun Guide*, Pafnuty Publications, 2014.

[59] V. Druskin and L. Knizhnerman, *Extended Krylov subspaces: approximation of the matrix square root and related functions*, SIAM J. Matrix Anal. Appl., 19 (1998), pp. 755–771.

[60] V. Druskin, C. Lieberman, and M. Zaslavsky, *On adaptive choice of shifts in rational Krylov subspace reduction of evolutionary problems*, SIAM J. Sci. Comput., 32 (2010), pp. 2485–2496.

[61] M. Eiermann and O. G. Ernst, *A restarted Krylov subspace method for the evaluation of matrix functions*, SIAM J. Numer. Anal., 44 (2006), pp. 2481–2504.

[62] M. Eiermann, O. G. Ernst, and S. Güttel, *Deflated restarting for matrix functions*, SIAM J. Matrix Anal. Appl., 32 (2011), pp. 621–641.

[63] S. W. Ellacott, *Computation of Faber series with application to numerical polynomial approximation in the complex plane*, Math. Comp., 40 (1983), pp. 575–587.

[64] E. N. Epperly, J. A. Tropp, and R. J. Webber, *XTrace: making the most of every sample in stochastic trace estimation*, SIAM J. Matrix Anal. Appl., 45 (2024), pp. 1–23.

[65] A. Eremenko and P. Yuditskii, *Uniform approximation of* $\operatorname{sgn}(x)$ *by polynomials and entire functions*, J. Anal. Math., 101 (2007), pp. 313–324.

[66] ——, *Polynomials of the best uniform approximation to* $\operatorname{sgn}(x)$ *on two intervals*, J. Anal. Math., 114 (2011), pp. 285–315.

[67] E. Estrada, E. Hameed, N. Hatano, and M. Langer, *Path Laplacian operators and superdiffusive processes on graphs. I. One-dimensional case*, Linear Algebra Appl., 523 (2017), pp. 307–334.

[68] E. Estrada, E. Hameed, M. Langer, and A. Puchalska, *Path Laplacian operators and superdiffusive processes on graphs. II. Two-dimensional lattice*, Linear Algebra Appl., 555 (2018), pp. 373–397.

[69] D. Fasino, *Rational Krylov matrices and QR steps on Hermitian diagonal-plus-semiseparable matrices*, Numer. Linear Algebra Appl., 12 (2005), pp. 743–754.

[70] G. Fertin, E. Godard, and A. Raspaud, *Acyclic and k-distance coloring of the grid*, Inform. Process. Lett., 87 (2003), pp. 51–58.

[71] A. Frommer, S. Güttel, and M. Schweitzer, *Convergence of restarted Krylov subspace methods for Stieltjes functions of matrices*, SIAM J. Matrix Anal. Appl., 35 (2014), pp. 1602–1624.

[72] ——, *Efficient and stable Arnoldi restarts for matrix functions based on quadrature*, SIAM J. Matrix Anal. Appl., 35 (2014), pp. 661–683.

[73] A. Frommer and P. Maass, *Fast CG-based methods for Tikhonov-Phillips regularization*, SIAM J. Sci. Comput., 20 (1999), pp. 1831–1850.

[74] A. Frommer, M. Rinelli, and M. Schweitzer, *Analysis of stochastic probing methods for estimating the trace of functions of sparse symmetric matrices*, to appear in Math. Comp., 2024.

[75] A. Frommer, C. Schimmel, and M. Schweitzer, *Bounds for the decay of the entries in inverses and Cauchy-Stieltjes functions of certain sparse, normal matrices*, Numer. Linear Algebra Appl., 25 (2018), e2131.

[76] ——, *Non-Toeplitz decay bounds for inverses of Hermitian positive definite tridiagonal matrices*, Electron. Trans. Numer. Anal., 48 (2018), pp. 362–372.

[77] ——, *Analysis of probing techniques for sparse approximation and trace estimation of decaying matrix functions*, SIAM J. Matrix Anal. Appl., 42 (2021), pp. 1290–1318.

[78] A. Frommer and M. Schweitzer, *Error bounds and estimates for Krylov subspace approximations of Stieltjes matrix functions*, BIT, 56 (2016), pp. 865–892.

[79] A. Frommer and V. Simoncini, *Matrix Functions*, in Model Order Reduction: Theory, Research Aspects and Applications, vol. 13 of Math. Ind., Springer, Berlin, 2008, pp. 275–303.

[80] ——, *Stopping criteria for rational matrix functions of Hermitian and symmetric matrices*, SIAM J. Sci. Comput., 30 (2008), pp. 1387–1412.

[81] R. E. Funderlic and R. J. Plemmons, *LU decomposition of M-matrices by elimination without pivoting*, Linear Algebra Appl., 41 (1981), pp. 99–110.

[82] A. S. Gambhir, A. Stathopoulos, and K. Orginos, *Deflation as a method of variance reduction for estimating the trace of a matrix inverse*, SIAM J. Sci. Comput., 39 (2017), pp. A532–A558.

[83] W. Gautschi, *Orthogonal Polynomials: Computation and Approximation*, Numerical Mathematics and Scientific Computation, Oxford University Press, New York, 2004. Oxford Science Publications.

[84] A. Ghavasieh and M. D. Domenico, *Generalized network density matrices for analysis of multiscale functional diversity*, Phys. Rev. E, 107 (2023), 044304.

[85] A. Ghavasieh and M. D. Domenico, *Statistical physics of network structure and information dynamics*, J. Phys.: Complex., 3 (2022), 011001.

[86] G. Golub and W. Kahan, *Calculating the singular values and pseudo-inverse of a matrix*, J. Soc. Indust. Appl. Math. Ser. B Numer. Anal., 2 (1965), pp. 205–224.

[87] G. H. Golub and G. Meurant, *Matrices, Moments and Quadrature with Applications*, Princeton Series in Applied Mathematics, Princeton University Press, Princeton, NJ, 2010.

[88] G. H. Golub and C. F. Van Loan, *Matrix Computations*, Johns Hopkins Studies in the Mathematical Sciences, Johns Hopkins University Press, Baltimore, MD, fourth ed., 2013.

[89] S. Güttel, *Rational Krylov Methods for Operator Functions*, PhD thesis, Technische Universität Bergakademie Freiberg, Germany, 2010. Dissertation available as MIMS Eprint 2017.39.

[90] ——, *Rational Krylov approximation of matrix functions: numerical methods and optimal pole selection*, GAMM-Mitt., 36 (2013), pp. 8–31.

[91] S. Güttel, D. Kressner, and K. Lund, *Limited-memory polynomial methods for large-scale matrix functions*, GAMM-Mitt., 43 (2020), e202000019.

[92] S. Güttel and M. Schweitzer, *A comparison of limited-memory Krylov methods for Stieltjes functions of Hermitian matrices*, SIAM J. Matrix Anal. Appl., 42 (2021), pp. 83–107.

[93] L. Han, F. Escolano, E. R. Hancock, and R. C. Wilson, *Graph characterizations from von Neumann entropy*, Pattern Recognit. Lett., 33 (2012), pp. 1958–1967.

[94] J. B. Hawkins and A. Ben-Israel, *On generalized matrix functions*, Linear Multilinear Algebra, 1 (1973), pp. 163–171.

[95] P. Henrici, *Applied and Computational Complex Analysis. Vol. 1*, Wiley Classics Library, John Wiley & Sons, Inc., New York, 1988. Reprint of the 1974 original.

[96] N. J. Higham, *Functions of Matrices: Theory and Computation*, SIAM, Philadelphia, PA, 2008.

[97] R. A. Horn and C. R. Johnson, *Topics in Matrix Analysis*, Cambridge University Press, Cambridge, 1994. Corrected reprint of the 1991 original.

[98] M. F. Hutchinson, *A stochastic estimator of the trace of the influence matrix for Laplacian smoothing splines*, Comm. Statist. Simulation Comput., 18 (1989), pp. 1059–1076.

[99] M. Ilić, F. Liu, I. Turner, and V. Anh, *Numerical approximation of a fractional-in-space diffusion equation. II. With nonhomogeneous boundary conditions*, Fract. Calc. Appl. Anal., 9 (2006), pp. 333–349.

[100] M. Ilić and I. Turner, *Approximating functions of a large sparse positive definite matrix using a spectral splitting method*, ANZIAM J., 46 (2004/05), pp. C472–C487.

[101] M. Ilić, I. Turner, and V. Anh, *A numerical solution using an adaptively preconditioned Lanczos method for a class of linear systems related with the fractional Poisson equation*, J. Appl. Math. Stoch. Anal., 2008. Art. ID 104525.

[102] D. Kressner, *Block algorithms for reordering standard and generalized Schur forms*, ACM Trans. Math. Software, 32 (2006), pp. 521–532.

[103] M. Kubale, *Graph Colorings*, Contemp. Math., American Mathematical Society, Providence, RI, 2004.

[104] A. B. J. Kuijlaars, *Convergence analysis of Krylov subspace iterations with methods from potential theory*, SIAM Rev., 48 (2006), pp. 3–40.

[105] L. D. Landau and E. M. Lifshitz, *Statistical Physics*, Pergamon Press, London, 1958.

[106] J. Liesen and Z. Strakoš, *Krylov Subspace Methods: Principles and Analysis*, Numerical Mathematics and Scientific Computation, Oxford University Press, Oxford, 2013.

[107] L. Lin, *Lecture Notes on Quantum Algorithms for Scientific Computation*, arXiv:2201.08309, 2022.

[108] L. Lin, J. Lu, and L. Ying, *Numerical methods for Kohn-Sham density functional theory*, Acta Numer., 28 (2019), pp. 405–539.

[109] L. Lin, Y. Saad, and C. Yang, *Approximating spectral densities of large matrices*, SIAM Rev., 58 (2016), pp. 34–65.

[110] T. Mach, M. S. Pranić, and R. Vandebril, *Computing approximate extended Krylov subspaces without explicit inversion*, Electron. Trans. Numer. Anal., 40 (2013), pp. 414–435.

[111] ——, *Computing approximate (block) rational Krylov subspaces without explicit inversion with extensions to symmetric matrices*, Electron. Trans. Numer. Anal., 43 (2014/15), pp. 100–124.

[112] G. Mantica, *Quantum dynamical entropy and an algorithm by Gene Golub*, Electr. Trans. Numer. Anal., 28 (2008), pp. 190–205.

[113] C. Martínez, M. Sanz, and J. Pastor, *A functional calculus and fractional powers for multivalued linear operators*, Osaka J. Math., 37 (2000), pp. 551–576.

[114] S. Massei and L. Robol, *Rational Krylov for Stieltjes matrix functions: convergence and pole selection*, BIT, 61 (2021), pp. 237–273.

[115] G. Meinardus, *Approximation of Functions: Theory and Numerical Methods*, Expanded translation of the German edition. Translated by Larry L. Schumaker. Springer Tracts in Natural Philosophy, Vol. 13, Springer-Verlag New York, Inc., New York, 1967.

[116] C. D. Meyer, *Matrix Analysis and Applied Linear Algebra*, SIAM, Philadelphia, PA, 2000.

[117] R. A. Meyer, C. Musco, C. Musco, and D. P. Woodruff, *Hutch++: Optimal stochastic trace estimation*, in Symposium on Simplicity in Algorithms (SOSA), SIAM, 2021, pp. 142–155.

[118] T. Michelitsch, A. P. Riascos, B. Collet, A. Nowakowski, and F. Nicolleau, *Fractional Dynamics on Networks and Lattices*, John Wiley & Sons, Ltd, 2019.

[119] I. Moret and P. Novati, *RD-rational approximations of the matrix exponential*, BIT, 44 (2004), pp. 595–615.

[120] ——, *Krylov subspace methods for functions of fractional differential operators*, Math. Comp., 88 (2019), pp. 293–312.

[121] C. Musco, C. Musco, and A. Sidford, *Stability of the Lanczos method for matrix function approximation*, in Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms, SIAM, Philadelphia, PA, 2018, pp. 1605–1624.

[122] Y. Nakatsukasa, O. Sète, and L. N. Trefethen, *The AAA algorithm for rational approximation*, SIAM J. Sci. Comput., 40 (2018), pp. A1494–A1522.

[123] A. M. N. Niklasson, *Density Matrix Methods in Linear Scaling Electronic Structure Theory*, in Linear-Scaling Techniques in Computational Chemistry and Physics: Methods and Applications, R. Zalesny, M. G. Papadopoulos, P. G. Mezey, and J. Leszczynski, eds., Springer Netherlands, Dordrecht, 2011, pp. 439–473.

[124] A. M. N. Niklasson, C. J. Tymczak, and M. Challacombe, *Trace resetting density matrix purification in O(N) self-consistent-field theory*, J. Chem. Phys., 118 (2003), pp. 8611–8620.

[125] V. Noferini, *A formula for the Fréchet derivative of a generalized matrix function*, SIAM J. Matrix Anal. Appl., 38 (2017), pp. 434–457.

[126] C. C. Paige, *Error analysis of the Lanczos algorithm for tridiagonalizing a symmetric matrix*, J. Inst. Math. Appl., 18 (1976), pp. 341–349.

[127] C. C. Paige, B. N. Parlett, and H. A. van der Vorst, *Approximate solutions and eigenvalue bounds from Krylov subspaces*, Numer. Linear Algebra Appl., 2 (1995), pp. 115–133.

[128] D. Palitta, S. Pozza, and V. Simoncini, *The short-term rational Lanczos method and applications*, SIAM J. Sci. Comput., 44 (2022), pp. A2843–A2870.

[129] D. Persson, A. Cortinovis, and D. Kressner, *Improved variants of the Hutch++ algorithm for trace estimation*, SIAM J. Matrix Anal. Appl., 43 (2022), pp. 1162–1185.

[130] M. S. Pranić and L. Reichel, *Rational Gauss quadrature*, SIAM J. Numer. Anal., 52 (2014), pp. 832–851.

[131] A. P. Riascos and J. L. Mateos, *Fractional dynamics on networks: Emergence of anomalous diffusion and Lévy flights*, Phys. Rev. E, 90 (2014), 032809.

[132] F. Roosta-Khorasani and U. Ascher, *Improved bounds on sample size for implicit matrix trace estimators*, Found. Comput. Math., 15 (2015), pp. 1187–1212.

[133] A. Ruhe, *Rational Krylov sequence methods for eigenvalue computation*, Linear Algebra Appl., 58 (1984), pp. 391–405.

[134] Y. Saad, *Iterative Methods for Sparse Linear Systems*, SIAM, Philadelphia, PA, second ed., 2003.

[135] R. L. Schilling, R. Song, and Z. Vondraček, *Bernstein Functions: Theory and Applications*, vol. 37 of De Gruyter Studies in Mathematics, Walter de Gruyter & Co., Berlin, second ed., 2012.

[136] C. Schimmel, *Bounds for the Decay in Matrix Functions and its Exploitation in Matrix Computations*, PhD thesis, Bergische Universität Wuppertal, Wuppertal, Germany, 2019.

[137] H. R. Schwarz, *Tridiagonalization of a symmetric band matrix*, Numer. Math., 12 (1968), pp. 231–241.

[138] M. Schweitzer, S. Güttel, and A. Frommer, *FUNM_QUAD: An implementation of a stable, quadrature based restarted Arnoldi method for matrix functions*, Technical Report, IMACM Preprint 14/04, 2014.

[139] J. Scott and M. Tůma, *Algorithms for Sparse Linear Systems*, Nečas Center Series, Birkhäuser/Springer, Cham, 2023.

[140] R. Silver and H. Röder, *Densities of states of mega-dimensional hamiltonian matrices*, Int. J. Mod. Phys. C, 5 (1994), pp. 735–753.

[141] V. Simoncini, *Restarted Full Orthogonalization Method for shifted linear systems*, BIT, 43 (2003), pp. 459–466.

[142] V. Simoncini and D. B. Szyld, *Recent computational developments in Krylov subspace methods for linear systems*, Numer. Linear Algebra Appl., 14 (2007), pp. 1–59.

[143] I. Simunec, *Error bounds for the approximation of matrix functions with rational Krylov methods*, Numer. Linear Algebra Appl., 2024, e2571.

[144] G. W. Stewart, *The efficient generation of random orthogonal matrices with an application to condition estimators*, SIAM J. Numer. Anal., 17 (1980), pp. 403–409.

[145] J. M. Tang and Y. Saad, *A probing method for computing the diagonal of a matrix inverse*, Numer. Linear Algebra Appl., 19 (2012), pp. 485–501.

[146] A. Taylor and D. J. Higham, *CONTEST: A controllable test matrix toolbox for MATLAB*, ACM Trans. Math. Softw., 35 (2009).

[147] L. N. Trefethen, *Approximation Theory and Approximation Practice*, SIAM, Philadelphia, PA, 2013.

[148] M. Van Barel, D. Fasino, L. Gemignani, and N. Mastronardi, *Orthogonal rational functions and structured matrices*, SIAM J. Matrix Anal. Appl., 26 (2005), pp. 810–829.

[149] N. Van Buggenhout, M. Van Barel, and R. Vandebril, *Biorthogonal rational Krylov subspace methods*, Electron. Trans. Numer. Anal., 51 (2019), pp. 451–468.

[150] J. van den Eshof, A. Frommer, T. Lippert, K. Schilling, and H. van der Vorst, *Numerical methods for the QCDd overlap operator. I. Sign-function and error bounds*, Comput. Phys. Commun., 146 (2002), pp. 203–224.

[151] J. van den Eshof and M. Hochbruck, *Preconditioning Lanczos approximations to the matrix exponential*, SIAM J. Sci. Comput., 27 (2006), pp. 1438–1457.

[152] R. Vandebril, M. Van Barel, and N. Mastronardi, *Matrix Computations and Semiseparable Matrices: Linear Systems. Vol. 1*, Johns Hopkins University Press, Baltimore, MD, 2008.

[153] J. von Neumann, *Mathematical Foundations of Quantum Mechanics*, Princeton University Press, Princeton, 1955. Translated by Robert T. Beyer.

[154] L.-W. Wang, *Calculating the density of states and optical-absorption spectra of large quantum systems by the plane-wave moments method*, Phys. Rev. B, 49 (1994), p. 10154.

[155] A. Wehrl, *General properties of entropy*, Rev. Mod. Phys., 50 (1978), pp. 221–260.

[156] T. P. Wihler, B. Bessire, and A. Stefanov, *Computing the entropy of a large matrix*, J. Phys. A, 47 (2014), 245201.