Theory and Algorithms for Periodic Functions of Matrices, with Applications

Aprahamian, Mary

2016

MIMS EPrint: **2016.28**

Manchester Institute for Mathematical Sciences
School of Mathematics

The University of Manchester

Reports available from: http://eprints.maths.manchester.ac.uk/
And by contacting: The MIMS Secretary
School of Mathematics
The University of Manchester
Manchester, M13 9PL, UK

ISSN 1749-9097

Theory and Algorithms for Periodic Functions of Matrices, with Applications

2016

Mary Aprahamian
School of Mathematics
The University of Manchester

Contents

List of Tables

List of Figures

Abstract

Mary Aprahamian
Doctor of Philosophy
Theory and Algorithms for Periodic Functions of Matrices, with Applications

Theoretical aspects of periodic functions of matrices and issues arising from the multivalued nature of their inverse functions are studied. Several algorithms for computing periodic and multivalued functions of matrices are developed.

We illustrate the use of matrix functions in the analysis of complex networks—an application that has recently been of very high interest. The relative importance of nodes in the whole network can be expressed via functions of the adjacency matrix. There are two functions which have proven popular in practice. The first one is the exponential, which has the advantage of being parameter-free. The second one is the resolvent function, which can be the more computationally efficient, but it depends on a parameter. We give a prescription for selecting this parameter aiming to match the rankings of the exponential counterpart.

We define a new matrix function, the matrix unwinding function, corresponding to the scalar unwinding number of Corless, Hare, and Jeffrey introduced in 1996. The matrix unwinding function is shown to be an important tool for deriving identities involving the matrix logarithm and fractional matrix powers. We propose an algorithm for computing the matrix unwinding function based on the Schur–Parlett method with a special reordering. The matrix unwinding function is shown to be useful for computing the matrix exponential using an idea of argument reduction.

We study theoretical and computational aspects of matrix inverse trigonometric and inverse hyperbolic functions. Conditions for existence are given and principal values are defined and shown to be unique primary matrix functions. We derive various functional identities, with care taken to specify choices of signs and branches. An important tool for the derivations is the matrix unwinding function. We derive a new algorithm employing a Schur decomposition and variable-degree Padé approximation for computing the principal inverse cosine (acos). It is shown how it can also be used to compute the matrix asin, acosh, and asinh. In numerical experiments the algorithm is found to behave in a forward stable fashion.

Finally, we consider argument reduction in computing the sine and cosine, and their hyperbolic counterparts. New algorithms for these functions are given, which use the matrix unwinding function with multiple angle algorithms for the sine and cosine. An argument reduction algorithm for computing general periodic functions of matrices is presented. Numerical experiments illustrate the computational saving that can accrue from applying argument reduction.

Declaration

No portion of the work referred to in the thesis has been submitted in support of an application for another degree or qualification of this or any other university or other institute of learning.

Publications

- The material in Chapter 2 is based on the paper

  [13] Mary Aprahamian, Desmond J. Higham, and Nicholas J. Higham. Matching exponential-based and resolvent-based centrality measures. *Journal of Complex Networks*. Advance Access Published June 29, 2015.

- The material in Chapter 3 is based on the paper

  [14] Mary Aprahamian and Nicholas J. Higham. The matrix unwinding function, with an application to computing the matrix exponential. *SIAM J. Matrix Anal. Appl. 35 (1): 88–109, 2014.*

- The material in Chapter 4 is based on the paper

  [15] Mary Aprahamian and Nicholas J. Higham. Matrix inverse trigonometric and inverse hyperbolic functions: Theory and algorithms. MIMS EPrint 2016.4, January 2016.

- The material in Chapter 5 is partially based on the paper [14].

Acknowledgements

I would like to extend my most sincere gratitude to my supervisor, Prof. Nicholas J. Higham, without whose expert guidance and support this thesis would not have been possible. In the future, I can only hope to adhere to the standard of writing he has guided me to and the meticulous attention to detail he has always encouraged.

I would also like to thank Prof. Françoise Tisseur for nurturing a supportive and engaging research environment in the Manchester Numerical Linear Algebra group; it has been my privilege to be a part of it. I am extremely grateful to Dr Stefan Güttel, for sharing his knowledge and enthusiasm for the field in the many discussions we have had. I thank my colleagues and office mates Bahar, Nataša, Sam, Leo and Ramaseshan, for many interesting discussions and distractions.

I thank Prof. Desmond J. Higham for the useful discussions and collaboration, which has lead to the publication associated with Chapter 2 of this thesis. I thank Prof. Bruno Iannazzo for his insightful comments on an earlier version of the publication associated with Chapter 3.

Finally, I thank my family for all their support and encouragement.

CHAPTER 1

Introduction

The interest in matrix functions as an object of research began in 1858 with a study of the square roots of 2×2 and 3×3 matrices in Cayley's "A Memoir on the Theory of Matrices" [38]. The following hundred years were very fruitful in the development of the theory of matrix functions. Some noteworthy results include the definition of the matrix exponential using power series by Laguerre [101] in 1867 and in 1888 by Peano [126]. A definition of functions of matrices with distinct eigenvalues using interpolating polynomials was given by Sylvester [139] in 1883, and later refined and generalized by Buchheim [32], [33]. An early appearance of transcendental functions of matrices was in 1892 in Metzeler's work [111]. He defined the exponential, logarithm, sine and inverse sine functions via their power series. Since then periodic functions of matrices and their multivalued inverses have been subjects of extensive research, both from theoretical and computational points of view. An informative survey of the history of these and other matrix functions can be found in Higham's book [80, Sec. 1.10].

Nowadays matrix functions are an integral part of the solutions of many problems in applied mathematics. Some examples include the matrix exponential, sine and cosine, which arise naturally in linear matrix differential equations [71]. The matrix sign function [80, Chap. 5] arises in control theory [102, Chap. 22], [105] through its relations with Sylvester equations and has recently found application in lattice quantum chromodynamics [67]. Matrix roots, exponential and logarithm of matrices with particular structure, such as stochastic matrices, arise in Markov

models. We direct the reader to [80, Chap. 2] for many more applications of matrix functions.

One application area we consider in particular in this thesis is network science [56], [116], where matrix functions are a valuable tool for analysing the properties of complex networks from a range of areas—biochemical networks [50], protein–protein interaction networks [149], social and economic networks [65], [145], to name a few. Other problems we refer to include convection–diffusion equations [91], open quantum systems [30] and wave equations [19, Chap. 10], [69, Sec. 5.5], [71].

The thesis is organized as follows.

In the remaining sections of Chapter 1 we give some useful background material on matrix functions.

In Chapter 2 we consider a particular complex networks problem which illustrates the use of matrix functions in this area. The relative importance of nodes in the network can be expressed via functions of the adjacency matrix. Two functions have been particularly popular in practice. The first one is the exponential, which has the advantage of being parameter free. The second is the resolvent function, which can be the more computationally efficient, especially for large directed networks, and has the benefit of generalizing naturally to time-dependent network sequences, but it depends on a parameter. We give a prescription for selecting this parameter aiming to match the rankings of the exponential counterpart. For our new choice of parameter the resolvent can be very ill conditioned, but we demonstrate that it can nevertheless reliably be used for ranking.

In Chapter 3 we introduce a new matrix function corresponding to the scalar unwinding number of Corless, Hare, and Jeffrey [94]. This matrix unwinding function, $\mathcal{U}$, is shown to be an important tool for deriving identities involving the matrix logarithm and fractional matrix powers. We use it to reveal, for example, the precise relation between $\log A^\alpha$ and $\alpha \log A$. Results showing the close connection between the unwinding function and the matrix sign function are given. We propose an algorithm for computing the unwinding function based on the Schur–Parlett method with a special reordering.

In Chapter 4 we study theoretical and computational aspects of matrix inverse

trigonometric and inverse hyperbolic functions. Conditions for existence are given and principal values are defined and shown to be unique primary matrix functions. We derive various functional identities, with care taken to specify choices of signs and branches. While some scalar identities, such as $\cos(-z) = \cos(z)$, $\sin(-z) = -\sin(z)$, and the sine and cosine addition and subtraction formulas translate directly to the matrix case, the derivation of identities for complex matrices are generally not straightforward. Some of the new results we present include explicit relations between the functions and their inverses, i.e., we give formulas describing $f^{-1}(f(A))$ for f^{-1} the principal inverse cosine (acos), inverse sine (asin), inverse hyperbolic cosine (acosh), and inverse hyperbolic sine (asinh). Important tools we use are the matrix unwinding function and the matrix sign function. We derive a new inverse scaling and squaring type algorithm, which employs a Schur decomposition and variable-degree Padé approximation for computing acos. We also show how it can be used to compute asin, acosh, and asinh. In numerical experiments the algorithm is demonstrated to behave in a forward stable way and to be preferable to computing these functions via the logarithm.

In Chapter 5 we consider argument reduction in computing the exponential, sine and cosine, and their hyperbolic counterparts. Weshow that matrix argument reduction using the function $\mathrm{mod}(A) = A - 2\pi i\,\mathcal{U}(A)$, which has spectrum with imaginary parts in the interval $(-\pi, \pi]$ and for which $\mathrm{e}^A = \mathrm{e}^{\mathrm{mod}(A)}$, can offer significant savings in the computation of the exponential by scaling and squaring algorithms. New algorithms are given, which use the matrix unwinding function with multiple angle algorithms for computing sine and cosine. A generalized argument reduction algorithm for computing periodic functions of matrices is presented. It uses the relation $f(A) = f(A - p\mathcal{U}_f(A))$ for a complex period p and the generalized matrix unwinding function $\mathcal{U}_f(A)$ all of whose eigenvalues are integers. The algorithm computes f at its reduced argument $A - p\mathcal{U}_f(A)$, which may be more economical. An algorithm for $\mathcal{U}_f(A)$ based on the Schur–Parlett method with a special reordering and blocking is presented. Numerical experiments illustrate the computational saving that can accrue from applying argument reduction.

Conclusions and remarks on future work are given in Chapter 6.

Figure 1.1 illustrates the main contributions of the thesis and the connections between them. A variant of this thesis, organised in Alternative thesis format was submitted to The University of Manchester for the degree of Doctor of Philosophy.

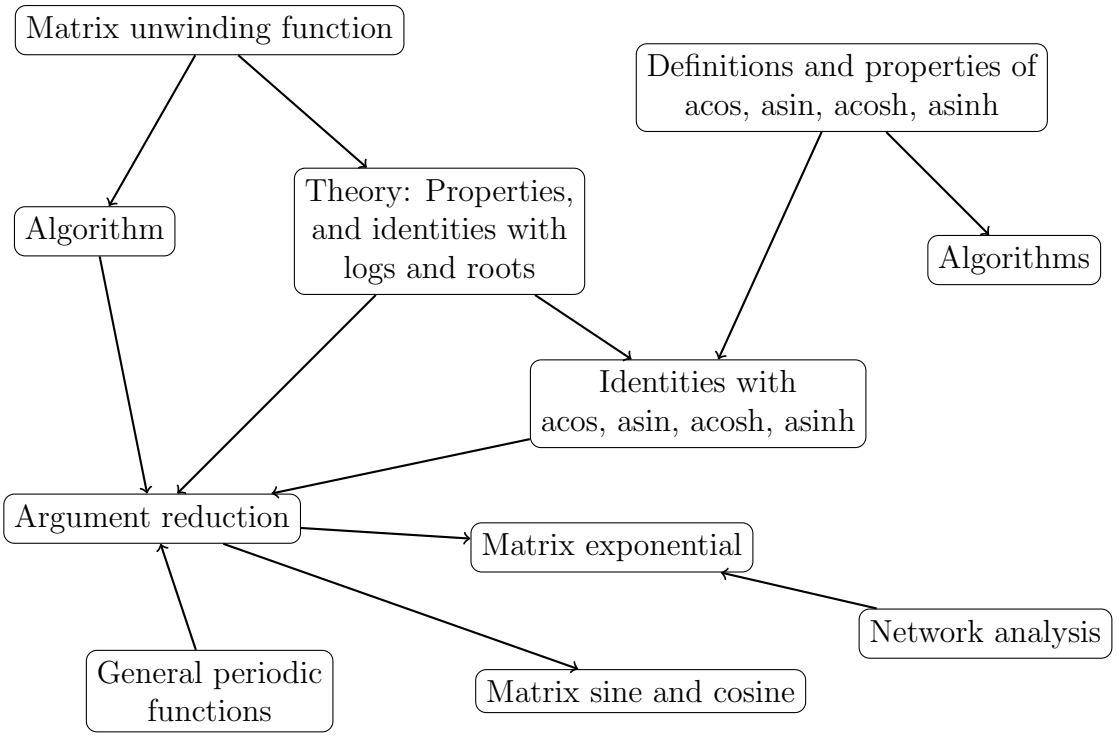


Figure 1.1: Research contributions of the thesis and the connections between them.

We now introduce the most important concepts and give some preliminary results.

1.1 Definitions and properties of matrix functions

The concept of *matrix functions* offers a straightforward generalization of a mapping $f : \mathbb{C} \to \mathbb{C}$ to the domain of complex square matrices $\mathbb{C}^{n\times n}$, so that a matrix $A \in \mathbb{C}^{n\times n}$ is mapped to a complex matrix of the same dimension as A. There are different rigorous definitions which formalize this intuitive interpretation of matrix functions. We focus on three definitions, which have been discussed in more detail in [80, Sec. 1.2].

Any $A \in \mathbb{C}^{n\times n}$ can be expressed in the *Jordan canonical form*

$$Z^{-1}AZ = J = \operatorname{diag}(J_1, J_2, \dots, J_p), \tag{1.1a}$$

$$J_k = J_k(\lambda_k) = \begin{bmatrix} \lambda_k & 1 & & \\ & \lambda_k & \ddots & \\ & & \ddots & 1 \\ & & & \lambda_k \end{bmatrix} \in \mathbb{C}^{m_k\times m_k}, \tag{1.1b}$$

where Z is nonsingular and $m_1 + m_2 + \cdots + m_p = n$. Denote by $\lambda_1, \lambda_2, \dots, \lambda_s$ the distinct eigenvalues of A and let n_i be the order of the largest Jordan block containing λ_i (also referred to as the index of λ_i). Suppose the function f is *defined on the spectrum of* A, i.e., all the derivatives $f^{(j)}(\lambda_i)$, $j = 0 : n_i - 1$, $i = 1 : s$ exist. The matrix function $f(A)$ is defined as [80, Def. 1.2]

$$f(A) = Zf(J)Z^{-1} = Z\operatorname{diag}(f(J_k))Z^{-1}, \tag{1.2}$$

where

$$f(J_k) := \begin{bmatrix} f(\lambda_k) & f'(\lambda_k) & \dots & \frac{f^{(m_k-1)}(\lambda_k)}{(m_k-1)!} \\ & f(\lambda_k) & \ddots & \vdots \\ & & \ddots & f'(\lambda_k) \\ & & & f(\lambda_k) \end{bmatrix}. \tag{1.3}$$

This definition of $f(A)$ calls for a few important remarks to be made. The Jordan matrix J is unique up to the ordering of the blocks J_i, however, the transformation matrix is not unique. The above definition of $f(A)$ is in fact independent of the particular Jordan canonical form used.

If A is diagonalizable, the Jordan canonical form reduces to the eigendecomposition $A = ZDZ^{-1}$, where $D = \operatorname{diag}(\lambda_i)$ and the columns of Z are the eigenvectors of A. We can write $f(A)$ as

$$f(A) = Zf(D)Z^{-1} = Z\operatorname{diag}(f(\lambda_i))Z^{-1}.$$

If f is a multivalued function defined via multiple branches in its natural domain, it is necessary that the same branch of f is taken for all eigenvalues which appear in the same Jordan block of A. Moreover, the same branch of f must be chosen for all equal eigenvalues of A even if they appear in different Jordan blocks. This requirement yields *primary matrix functions*, otherwise *nonprimary matrix*

functions are obtained. Although nonprimary matrix functions may be of interest in some applications, we restrict our studies to the much more popular class of primary matrix functions; additional detail on nonprimary matrix functions can be found in [80, Sec. 1.4].

To illustrate the distinction between primary and nonprimary matrix functions we give the following example. Let

$$J = \begin{bmatrix} -1 & 1 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & -1 \end{bmatrix}$$

and f be the square root function $f(x) = x^{1/2}$, with its principal branch characterized by $(-1)^{1/2} = i$. One of the primary square roots of J is

$$J^{1/2} = \begin{bmatrix} i & -i/2 & 0 & 0 \\ 0 & i & 0 & 0 \\ 0 & 0 & i & 0 \\ 0 & 0 & 0 & i \end{bmatrix}.$$

In addition, taking different branches of the square roots functions in each block of the Jordan form gives rise to nonprimary square roots, e.g.,

$$\begin{bmatrix} i & -i/2 & 0 & 0 \\ 0 & i & 0 & 0 \\ 0 & 0 & -i & 0 \\ 0 & 0 & 0 & -i \end{bmatrix}, \quad \begin{bmatrix} i & -i/2 & 0 & 0 \\ 0 & i & 0 & 0 \\ 0 & 0 & -i & 0 \\ 0 & 0 & 0 & i \end{bmatrix}, \quad \begin{bmatrix} -i & i/2 & 0 & 0 \\ 0 & -i & 0 & 0 \\ 0 & 0 & i & 0 \\ 0 & 0 & 0 & -i \end{bmatrix}.$$

The second definition of matrix functions we consider involves interpolating polynomials. Let ψ denote the minimal polynomial of $A \in \mathbb{C}^{n\times n}$, i.e., the unique monic polynomial of lowest degree such that $\psi(A) = 0$. Assuming again that f is defined on the spectrum of A, the function $f(A)$ can be defined as $f(A) := p(A)$, where p is the polynomial of degree less than $\sum_{i=1}^{s} n_i = \deg \psi$ that satisfies the interpolation conditions

$$p^{(j)}(\lambda_i) = f^{(j)}(\lambda_i), \quad j = 0 : n_i - 1, \quad i = 1 : s. \tag{1.4}$$

As before, $\lambda_1, \ldots, \lambda_s$ are the distinct eigenvalues of A and n_i is the size of the largest Jordan block in which λ_i appears. The polynomial p is unique and it is

known as the *Hermite interpolating polynomial* [80, Def. 1.4]. It is important to note that p is a polynomial whose coefficients depend on A.

The third and last definition of a matrix function we consider is a generalization of the *Cauchy integral formula* [80, Def. 1.11]. For $A \in \mathbb{C}^{n\times n}$ and a function f analytic on and inside a closed contour Γ that encloses the spectrum of A,

$$f(A) := \frac{1}{2\pi i}\int_{\Gamma} f(z)(zI - A)^{-1}\,dz. \tag{1.5}$$

This definition leads to elegant proofs of some theoretical results.

It can be shown that the three definitions (1.2), (1.4) and (1.5) are equivalent, modulo the analyticity requirement of the Cauchy integral formula, for a proof see [80, Thm. 1.12].

The following fundamental properties of matrix functions will be used repeatedly throughout this thesis. They can be found in [80, Thm. 1.13].

Theorem 1.1. *Let $A \in \mathbb{C}^{n\times n}$ and let f be defined on the spectrum of A. Then*

(i) *$f(A)$ commutes with A.*

(ii) *$f(A^T) = f(A)^T$.*

(iii) *$f(XAX^{-1}) = Xf(A)X^{-1}$ for any nonsingular $X \in \mathbb{C}^{n\times n}$.*

(iv) *the eigenvalues of $f(A)$ are $f(\lambda_i)$, where λ_i are the eigenvalues of A.*

(v) *if $X \in \mathbb{C}^{n\times n}$ commutes with A then X commutes with $f(A)$.*

(vi) *if $A = (A_{ij})$ is a block triangular matrix, then $F = f(A)$ is block triangular with the same block structure as A, and $F_{ii} = f(A_{ii})$.*

1.2 Fréchet derivatives and condition numbers

It is important to understand the sensitivity of matrix functions to perturbations in the data. Often the input matrices, especially those arising from applications, are inexact or have embedded uncertainties. Even if the data is exact, computations are subject to rounding errors, which may also be viewed as perturbations. Sensitivity is measured by condition numbers, therefore for each matrix function it

is instructive to consider the magnitude of the condition numbers and provide algorithms for computing them. Condition numbers can be expressed via the norms of the Fréchet derivatives, so we consider their properties too.

Let $\mathcal{C}$ be an open subset of $\mathbb{C}^{n\times n}$. The *Fréchet derivative* of a matrix function $f : \mathcal{C} \to \mathbb{C}^{n\times n}$ at a point $X \in \mathcal{C}$ is a linear mapping L such that for all $E \in \mathbb{C}^{n\times n}$

$$f(X + E) - f(X) - L(X, E) = o(\|E\|). \tag{1.6}$$

The matrix E is referred to as the direction of the derivative. The *absolute and relative condition numbers* are defined as

$$\mathrm{cond}_{\mathrm{abs}}(f, X) := \lim_{\epsilon\to 0} \sup_{\|E\|\le\epsilon} \frac{\|f(X + E) - f(X)\|}{\epsilon}, \tag{1.7}$$

$$\mathrm{cond}_{\mathrm{rel}}(f, X) := \lim_{\epsilon\to 0} \sup_{\|E\|\le\epsilon\|X\|} \frac{\|f(X + E) - f(X)\|}{\epsilon\|f(X)\|}. \tag{1.8}$$

Note that $\mathrm{cond}_{\mathrm{abs}}$ and $\mathrm{cond}_{\mathrm{rel}}$ differ only by a constant factor,

$$\mathrm{cond}_{\mathrm{rel}}(f, X) = \mathrm{cond}_{\mathrm{abs}}(f, X)\frac{\|X\|}{\|f(X)\|}. \tag{1.9}$$

Usually only the relative condition number is of interest, however the absolute one is easier to work with.

If we set

$$\|L(X)\| := \max_{Z\neq 0} \frac{\|L(X, Z)\|}{\|Z\|}, \tag{1.10}$$

the absolute and relative condition numbers can be expressed as

$$\mathrm{cond}_{\mathrm{abs}}(f, X) = \|L(X)\|, \tag{1.11}$$

$$\mathrm{cond}_{\mathrm{rel}}(f, X) = \frac{\|L(X)\|\|X\|}{\|f(X)\|}. \tag{1.12}$$

For the proofs of (1.11) and (1.12) see [80, Thm. 3.1]. The definitions of the absolute and relative condition numbers (1.7) and (1.8), respectively, are special cases of results by Rice [133].

The Fréchet derivative does not always exist. The following result, the proof of which can be found in [80, Thm. 3.8], gives a necessary condition for existence. If we let f be $2n - 1$ times continuously differentiable on its domain $\mathcal{D}$, for X with spectrum in $\mathcal{D}$, the Fréchet derivative $L(X, E)$ exists and is continuous in X

and E. This, together with a result by Mathias [108] allows us to obtain the Fréchet derivative $L(X, E)$ explicitly as the $(1, 2)$ block of the following matrix function

$$f\left(\begin{bmatrix} X & E \\ 0 & X \end{bmatrix}\right) = \begin{bmatrix} f(X) & L(X, E) \\ 0 & f(X) \end{bmatrix}. \tag{1.13}$$

This is a very useful formula both in theory and in practice as it reduces the computation of the Fréchet derivative to the computation of a single matrix function. Note that the size of the matrix function is twice that of the original matrix X, so an obvious computational drawback of using this formula may appear if the dimension n is large.

1.3 Floating point computation

A *floating point number system* $F \subset \mathbb{R}$ is a subset of the real numbers whose elements can be written as

$$y = \pm m \times \beta^{e-t}, \tag{1.14}$$

where all four parameters m, β, e and t are integers, known as significand (also mantissa), base, exponent and precision, respectively. The exponent is in the range $e_{\min} \leq e \leq e_{\max}$ and the significand satisfies $0 \leq m \leq \beta^t - 1$. To ensure that the representation of each nonzero $y \in F$ is unique it is assumed that the significand satisfies $m \geq \beta^{t-1}$. The range of nonzero floating point numbers in F is $\beta^{e_{\min}-1} \leq |y| \leq \beta^{e_{\max}}(1 - \beta^{-t})$.

Denoting by $G \subset \mathbb{R}$ all real numbers of the form (1.14) with no restrictions on the exponent e and letting x be a real number, then $fl(x)$ denotes the element of G closest to x and the mapping $x \to fl(x)$ is known as rounding. Although fl is defined as a mapping onto G, here we are only interested in cases for which $fl(x) \in F$.

The most useful quantity associated with F is the *unit roundoff* $u = \frac{1}{2}\beta^{1-t}$. The following result, which appears in [78, Thm. 2.2], shows that every real number x in the range of F can be approximated by an element of F with relative error not larger than the unit roundoff.

Theorem 1.2. *If $x \in \mathbb{R}$ lies in the range of F, then*

$$fl(x) = x(1+\delta), \quad |\delta| < u.$$

Floating point arithmetic is a calculation which involves the elementary operations of addition, subtraction, multiplication and division of floating point numbers. These operations are known as *floating point operations* (flops). We will use the total number of flops required by an algorithm as a measure of its complexity. The standard model of floating point arithmetic with $x, y \in F$ is [78, Sec. 2.2, (2.4)]

$$fl(x \operatorname{op} y) = (x \operatorname{op} y)(1+\delta), \quad |\delta| \leq u, \quad \operatorname{op} = +, -, *, /.$$

It is assumed that the standard model also holds for the square root operation.

Unless stated otherwise, all computation in this thesis has been carried out in IEEE double precision arithmetic, as specified by the 2008 modification of the 754 standard [1]. The IEEE double arithmetic system is characterized by a base $\beta = 2$, precision $t = 53$ and range $[e_{\min}, e_{\max}] = [-1021, 1024]$. The unit roundoff is $u = 2^{-53} \approx 1.11 \times 10^{-16}$.

CHAPTER 2

Matching Exponential-Based and Resolvent-Based Centrality

2.1 Introduction

The term *centrality* refers to a real number associated with a node of a network that conveys information about its relative "importance." Centrality measures came to prominence in social network analysis [145], but have proved to be extremely useful tools across network science [56], [116]. A discussion of the intuitive interpretation of centrality measures in social networks in presented by Freeman [64]. A number of measures derived from the degree of the nodes have since emerged, the simplest and most popular one of them is using the total degree of a vertex as an index of importance. It was first introduced in 1974 in a paper by Nieminen [118]. This measure is considered most appropriate for scale-free networks for which only the immediate connection between any two nodes is significant, see for example [10] and [146]. Betweenness and closeness centrality measures are based on the number of shortest paths between two nodes and passing through a given node [63]. An important class of centrality measures is based on eigenvectors of the adjacency matrix. These measures are based on the idea that a node adjacent to an influential one must also be important. A variant of this idea is used by the Google's PageRank algorithm [120]. An important predecessor of PageRank is Kleinberg's HITS algorithm [100], based on the hubs and authorities scores in a network. The most

appropriate centrality measure for a given network is usually embedded in the formulation and characteristics of the particular model, the books [28], [54], [56], and [116] contain overviews of the most widely used models.

A popular way to define centrality is to quantify the ability of a node to initiate walks around the network, a concept that leads naturally to the use of matrix functions. Using standard notation, we let $A = (a_{ij})$ denote the adjacency matrix for an unweighted network of n nodes, so that $a_{ij} = 1$ if there is an edge from i to j and $a_{ij} = 0$ otherwise. It follows that the number of walks of length k from node i to node j is given by $(A^k)_{ij}$; see, for example, [23]. It is interesting to note that the use of matrix powers to count the number of walks between any two nodes of a network was considered as early as 1949 in a work by Festinger [60]. Soon after and first suggested by Katz [96] in 1953, resolvent-based centrality measures emerged. They penalize long walks through multiplication by a fixed factor $\alpha \geq 0$ for each edge used. This leads to a power series of the form $\sum_{k=0}^{\infty} \alpha^k A^k$, where for $i \neq j$ the (i, j) element gives a weighted count of the number of walks of all lengths from i to j. This series converges to the resolvent $(I - \alpha A)^{-1}$ for any $\alpha \in [0, 1/\rho(A))$, where $\rho(A)$ denotes the spectral radius of A. The ith row sum of the resolvent therefore summarizes the ability of node i to initiate walks to all nodes in the network. Similarly, the (i, i) element of the resolvent gives a weighted count of *closed* walks, that is, walks that start and finish at node i, with a uniform unit shift. Since we are concerned with the comparative performance across nodes, this shift is not important.

A related centrality concept arises from the suggestion by Estrada and Rodríguez-Velázquez [58] to weight walks of length k by the factor $1/k!$, so that the resolvent is replaced by $\sum_{k=0}^{\infty} A^k/k!$, which is the matrix exponential function, e^A [80, Chap. 10]. The authors define the subgraph centrality of a node to be the weighted sum of all closed walks originating from it, which can be computed as the diagonal entry (i, i) of e^A. Some justification for this definition is given by Estrada, Hatano, and Benzi [57, Sec. III] using the metaphor of a network as a system of oscillators.

In this work we measure the importance of a node via a weighted sum of both the open and closed walks starting from it; that is, we use the *total subgraph*

communicability of a node, introduced by Benzi and Klymko [20], as its measure of centrality. The associated exponential-based centrality measure of node i is thus given by the ith element of the vector

$$\boldsymbol{c}_{\boldsymbol{e}}(A) = \mathrm{e}^{A}\,\mathbf{1}, \tag{2.1}$$

where $\mathbf{1} = [1, 1, \ldots, 1]^T$. Similarly, the resolvent-based centrality of node i is the ith element of the vector

$$\boldsymbol{c}_{\boldsymbol{\alpha}}(A) = (I - \alpha A)^{-1}\,\mathbf{1}\,. \tag{2.2}$$

We note that the combinatorial "weighted walk count" interpretation of the matrix resolvent and matrix exponential extends naturally to the case of nonnegative integer weights if we interpret a_{ij} as recording the number of distinct connections between node i to j. For example, in a road network, if there are two distinct roads connecting town A and town B and three distinct roads connecting town B and town C, then there are $2 \times 3 = 6$ distinct ways to get from town A to town C in two hops via town B. The adjacency matrix power A^k therefore continues to count walks in this generalized sense, and the centrality vectors $\boldsymbol{c}_{\boldsymbol{e}}$ in (2.1) and $\boldsymbol{c}_{\boldsymbol{\alpha}}$ in (2.2) have a clear meaning. We also point out that in the case where A is non-symmetric, computing these centrality measures on the transpose, A^T, quantifies the propensity of nodes to receive, rather than broadcast, information.

The motivation for our work is that there currently seems to be no agreed mechanism for selecting the Katz parameter α, and, as we will show in Section 2.4, centrality rankings can be strongly dependent on this value. In order to derive and judge an approach for choosing α, we make the assumption that exponential-based total communicability is the "gold standard" and thereby seek to match this measure as closely as possible. Therefore we select α in (2.2) to match closely the centralities in (2.1).

This chapter is organized as follows. In Section 2.2 we pursue this approach for selecting a Katz parameter α, both for directed and undirected networks, and propose a new choice of the parameter. In Section 2.2.2 we give an overview of some particular choices of α that have appeared in the literature. In Section 2.3 we show that our new choice of Katz parameter can lead to a very ill conditioned

resolvent and explain why the ill conditioning is innocuous. Numerical experiments that test the performance of the proposed new value of the Katz parameter for ranking nodes in real networks are presented in Section 2.4. In Section 2.5 we briefly explain why computing resolvent-based centrality measures may be more favorable than the exponential versions for very large and sparse networks and also for time-dependent networks.

2.2 Katz parameter

The exponential-based centrality measure penalizes longer walks more heavily than the resolvent-based one; for a walk of length k the coefficient in the exponential series is $1/k!$, compared with α^k in the resolvent series. The exponential-based centrality has been found to yield meaningful results for some particular problems, for example those arising from biochemical applications [55]. Furthermore, in social networks and in other human interactions direct acquaintanceship is typically more important, which can be successfully exploited via the exponential-based centrality analysis [56, Chap. 19]. As we explain in Section 2.5, resolvent-based centrality has the advantage of extending naturally to the case of time-dependent network sequences. The resolvent measure is also more flexible, since α can be tuned according to the requirements of the specific problem. This, however, requires good knowledge of the network, which may not always be readily available to the person constructing the model. It is therefore desirable to have a prescription for a Katz parameter that closely matches the node rankings produced by the exponential measure. This will provide a computational alternative to the matrix exponential function for obtaining reliable node rankings.

2.2.1 A new Katz parameter

We propose a new method for selecting the Katz parameter that aims to minimize the norm of the difference between the centrality vectors $\boldsymbol{c_e}$ in (2.1) and $\boldsymbol{c_\alpha}$ in (2.2). This approach naturally ensures that the centralities of the nodes with the highest scores are closely matched. Indeed, in many applications it is only the best ranked

nodes that are of practical interest. We would therefore like to find α that solves

$$\min_\alpha \operatorname{err}(\alpha) := \min_\alpha \|\boldsymbol{c}_{\boldsymbol{e}}(A) - \boldsymbol{c}_{\boldsymbol{\alpha}}(A)\|_2 \quad \text{subject to} \quad 0 \le \alpha < 1/\rho(A), \tag{2.3}$$

where the 2-norm is defined by $\|\boldsymbol{x}\|_2 = (\boldsymbol{x}^T\boldsymbol{x})^{1/2}$. Initially we will make no assumptions about the network except that A is a diagonalizable matrix, so that $A = VDV^{-1}$, where $D = \operatorname{diag}(\lambda_i)$ contains the eigenvalues of A and V is nonsingular. (In fact, our derivation can be modified to use the Jordan canonical form when A is not diagonalizable, and the same value of α is obtained.) Since the matrix A is nonnegative, the Perron–Frobenius theory [90, Thm. 8.4.4] applied to A^T tells us that $\rho(A)$ is an eigenvalue of A with an associated nonnegative left eigenvector $\boldsymbol{y}$: $\boldsymbol{y}^TA = \rho(A)\boldsymbol{y}^T$. Without loss of generality we can take $\lambda_1 = \rho(A)$ and the first row of V^{-1} to be $\boldsymbol{y}^T$. We have

$$\begin{aligned}
\operatorname{err}(\alpha)^2 &= \|V\left(\mathrm{e}^D - (I-\alpha D)^{-1}\right)V^{-1}\mathbf{1}\|_2^2 \\
&\le \|V\|_2^2\|\left(\mathrm{e}^D - (I-\alpha D)^{-1}\right)V^{-1}\mathbf{1}\|_2^2 \qquad (2.4)\\
&= \|V\|_2^2\sum_{i=1}^n \left|\mathrm{e}^{\lambda_i} - \frac{1}{1-\alpha\lambda_i}\right|^2 |w_i|^2, \qquad (2.5)
\end{aligned}$$

where $\boldsymbol{w} = V^{-1}\mathbf{1}$. Then

$$\min_\alpha \operatorname{err}(\alpha)^2 \le \operatorname{err}(\alpha_{\min})^2 \le \|V\|_2^2\sum_{i=2}^n \left|\mathrm{e}^{\lambda_i} - \frac{1}{1-\alpha_{\min}\lambda_i}\right|^2 |w_i|^2, \tag{2.6}$$

where $\alpha_{\min}$ is such that $\left(\mathrm{e}^{\lambda_1} - 1/(1-\alpha_{\min}\lambda_1)\right)^2 w_1^2 = 0$. But $w_1 = \boldsymbol{y}^T\mathbf{1} \ne 0$ as $\boldsymbol{y}$ is a nonzero vector with nonnegative entries, so

$$\alpha_{\min} = \frac{1-\mathrm{e}^{-\lambda_1}}{\lambda_1}. \tag{2.7}$$

The value of the upper bound (2.6) on the minimum is governed both by the distribution of the eigenvalues of A and the sums w_i of the elements of the left eigenvectors of A.

Clearly, we need $\lambda_1 = \rho(A) > 0$ for $\alpha_{\min}$ to be defined. For an undirected network, $\rho(A) = 0$ implies $A = 0$, so $\rho(A) > 0$ can be assumed. For a directed network, if $\lambda_1 = 0$ then all the eigenvalues of A are zero and e^A and $(1-\alpha A)^{-1}$ have the same eigenvalues for all α. It is therefore not possible to choose α based purely on considerations of the spectrum and so some other approach must be used.

For the special case of normal adjacency matrices, i.e., ones that satisfy $A^TA = AA^T$, and hence are diagonalizable by orthogonal matrices—in particular, symmetric matrices, corresponding to undirected networks—we can take V orthogonal, and (2.4) and the second inequality in (2.6) are then equalities. Some classes of directed networks are known to have normal adjacency matrices. For example, (unweighted) "ring" networks are such that for $i = 1 : n-1$ there is an edge from node i to node $i+1$ and an edge from node n to node 1, and for these it is always true that $A^TA = AA^T = I$.

The upper bound (2.6) on $\min_\alpha \mathrm{err}(\alpha)$ is attained for certain types of graphs. For example, for unweighted and undirected k-regular graphs, where each node has degree k, it is easy to see that $\mathbf{1}$ is always an eigenvector of the adjacency matrix [23, Chap. 3] and then from the orthogonality of the eigenvectors it follows that $w_i = 0$ for all $i \geq 2$ [49]. In general the upper bound (2.6) provides a good estimate for $\min_\alpha \mathrm{err}(\alpha)$ either if A is such that there is a relatively big separation $|\lambda_1 - \mathrm{Re}(\lambda_2)|$ between its two eigenvalues with largest real part, or if $|w_1|$ is significantly larger than $|w_i|$ for all $i > 1$. These cases are common in practice, as we see from the examples in Section 2.4.

Benzi and Klymko [21, Sec. 9] observe experimentally that for both undirected and directed networks the exponential and resolvent measures differ the most for values of the Katz parameter that satisfy $0 \leq \alpha \leq 0.9/\lambda_1$. Provided $\lambda_1 > \log 10 \approx 2.3026$, $\alpha_{\min}$ avoids this interval. We note that by [90, Thm. 8.1.22] λ_1 lies between the smallest and largest row sums of the adjacency matrix of the network, so in practice such a small value for λ_1 will rarely be observed.

Finally, we note that $\alpha_{\min}$ can be readily adapted to match the rankings obtained from the more general parametrized exponential centrality $\mathrm{e}^{\beta A}$ [56, Chap. 5.2]. The parameter $\beta > 0$ can be interpreted as an artificial inverse temperature and reflects the influence of stress factors external to the system. This corresponds to a homogeneous scalar weighting of all the edges in a network, so the largest eigenvalue of the scaled system becomes $\beta\lambda_1$. For the corresponding Katz parameter we have $\alpha_{\min} = (1 - \mathrm{e}^{-\beta\lambda_1})/(\beta\lambda_1)$.

2.2.2 Other Katz parameters

Many different choices for the Katz parameter in the resolvent-based centrality measure have appeared in the literature, some of them proving more popular than others. In his original paper Katz suggests that a value for α in the interval $[1/(2\lambda_1), 1/\lambda_1)$ should be suitable [96]. Some authors have in particular chosen the value

$$\alpha_{0.5} = \frac{1}{2\lambda_1}$$

to study similarity in texts [11, p. 4411]. This Katz parameter has also successfully been used in the context of supply chain management [25]. We will use this value in our comparison studies in Section 2.4.

Another favored choice for the Katz parameter is [20]

$$\alpha_{0.85} = \frac{0.85}{\lambda_1}.$$

It arises by analogy with the damping factor of Google's PageRank algorithm, usually set to 0.85 [103].

In many applications the induced node rankings have been found to be very strongly dependent on the choice of the Katz parameter, so either an α particular to the model has been computed [121] or rankings have been reported for many values of α [31]. In some cases, the Katz parameter has a particularly meaningful interpretation, such as in protein–protein interaction networks [150], where it is indicative of the balance between the influence of the neighbors and the difference in activity levels.

Since λ_1 is bounded by any subordinate norm of the adjacency matrix, it is natural to propose a value for α that depends on such a norm. Foster et al. [61] suggest

$$\alpha_{\text{deg}} = \frac{1}{\|A\|_\infty + 1},$$

where the subscript relates to the fact that for networks with undirected and unweighted edges the ∞-norm is the largest node degree. The upper bound $\lambda_1 \leq \|A\|_\infty$ is known to be attained for several classes of networks. For k-regular

(undirected and unweighted) graphs it is always true that $\lambda_1 = \|A\|_\infty = k$. This includes complete graphs and rings.

In Section 2.4 we present a comparison between the node rankings obtained using the exponential-based centrality measure and its resolvent-based counterpart, computed with the Katz parameter taken as $\alpha_{\min}$ and the other options discussed in this section. But first we look more closely at the properties of $\alpha_{\min}$.

2.3 Conditioning

The resolvent-based centralities are the solution of a linear system, so it is of interest to know the conditioning of the coefficient matrix $I - \alpha A$ of that linear system, since this will influence the accuracy of the solution obtained in floating point arithmetic. Upper and lower bounds on the 2-norm condition number $\kappa_2(I - \alpha_{\min} A) = \|I - \alpha_{\min} A\|_2 \|(I - \alpha_{\min} A)^{-1}\|_2$ are given in the next result.

Lemma 2.1. *Let A be a nonnegative matrix and let λ_1 be an eigenvalue such that $\lambda_1 = \rho(A)$. Let $\alpha_{\min} = (1 - \mathrm{e}^{-\lambda_1})/\lambda_1$.*

(a) If A is diagonalizable, so that $A = VDV^{-1}$ with $D = \operatorname{diag}(\lambda_i)$ and V nonsingular, then

$$\kappa_2(I - \alpha_{\min} A) \le \kappa_2(V)^2 (2\mathrm{e}^{\lambda_1} - 1). \tag{2.8}$$

(b) If A has an eigenvalue with nonpositive real part then $\kappa_2(I - \alpha_{\min} A) \ge \mathrm{e}^{\lambda_1}$.

Proof. We have

$$\begin{aligned} \kappa_2(I - \alpha_{\min} A) &= \|V(I - \alpha_{\min} D)V^{-1}\|_2 \|V(I - \alpha_{\min} D)^{-1} V^{-1}\|_2 \\ &\le \kappa_2(V)^2 \kappa_2(I - \alpha_{\min} D). \end{aligned}$$

Now

$$\begin{aligned} \max_i |1 - \alpha_{\min}\lambda_i| &\le \max_i (1 + \alpha_{\min}|\lambda_i|) \\ &= 1 + \alpha_{\min}\lambda_1 \\ &= 1 + \left(\frac{1 - \mathrm{e}^{-\lambda_1}}{\lambda_1}\right)\lambda_1 \\ &= 2 - \mathrm{e}^{-\lambda_1}. \end{aligned}$$

Also, $\min_i |1-\alpha_{\min}\lambda_i| = |1-\alpha_{\min}\lambda_1| = \mathrm{e}^{-\lambda_1}$. Hence

$$\kappa_2(I-\alpha_{\min}D) \le \frac{2-\mathrm{e}^{-\lambda_1}}{\mathrm{e}^{-\lambda_1}} = 2\mathrm{e}^{\lambda_1} - 1.$$

Finally, if λ_k has nonpositive real part then $\|I-\alpha_{\min}A\|_2 \ge |1-\alpha_{\min}\lambda_k| \ge 1$, and $\|(I-\alpha_{\min}A)^{-1}\|_2 \ge |1-\alpha_{\min}\lambda_1|^{-1} = \mathrm{e}^{\lambda_1}$, which gives the lower bound. □

The condition in part (b) of the lemma is often satisfied in practice; indeed it is satisfied for the adjacency matrices of all the networks used in the experiments of Section 2.4, and more generally it is satisfied for any nonnegative A with zero diagonal.

The bounds in Lemma 2.1 are a cause for concern because they suggest that $I-\alpha_{\min}A$ is potentially very ill conditioned when either $\lambda_1 \gg 1$ or V is ill conditioned, the latter case corresponding to A being highly nonnormal. It is certainly possible that $\lambda_1 \gg 1$; indeed λ_1 is as large as 94 in our test problems in Section 2.4. Therefore $I-\alpha_{\min}A$ can be extremely ill conditioned, and in floating point arithmetic we can expect the computed centrality vector to have a large relative error. However, the ill conditioning is innocuous, as we now explain.

When we solve the linear system $(I-\alpha_{\min}A)\boldsymbol{x} = \mathbf{1}$, we are effectively carrying out an inverse iteration according to $(A-\alpha_{\min}^{-1}I)\boldsymbol{x} = -\alpha_{\min}^{-1}\mathbf{1}$, and for large λ_1, $\alpha_{\min}^{-1} = \lambda_1/(1-\mathrm{e}^{-\lambda_1})$ is a very good approximation to the eigenvalue λ_1 of A. Standard theory of inverse iteration [93, Sec. 6.3], [122, Sec. 4.3], [129, Sec. 2] shows that the error in the computed x will be almost parallel to x, that is, the inaccuracy is concentrated in its length and not its direction. Inverse iteration theory therefore tells us that while the computed centrality vector may be inaccurate, the relative sizes of the elements will be accurately determined. Since our interest in centralities is to assess the relative importance of nodes, we conclude that we can safely use $\alpha_{\min}$ in practice. We also observe that when $\alpha_{\min}^{-1}$ is a good approximation to λ_1, the vector of centrality scores x will be almost parallel to the nonnegative eigenvector corresponding to λ_1. In such cases the entries of this eigenvector give the relative importance of each node, and if only the relative importance of the nodes is required it is then not necessary to compute the centralities.

It is interesting to note that $I-\alpha_{\min}A$ is an M-matrix, since $\alpha_{\min} < 1/\lambda_1$ [22], but the above argument does not depend on any special properties of A.

2.4 Experiments with ranking

In our experiments we compare node rankings obtained from centrality vectors computed using the exponential-based and resolvent-based measures. Our computations are done in IEEE double precision arithmetic, which has unit roundoff $u \approx 1.1 \times 10^{-16}$.

For the resolvent measure we use each of the four choices for the Katz parameter suggested in Section 2.2: $\alpha_{\min}$, $\alpha_{0.5}$, $\alpha_{0.85}$, and $\alpha_{\deg}$. The first three depend on the largest eigenvalue λ_1 of the adjacency matrix A, which we compute using the MATLAB sparse eigensolver `eigs` with the vector of all ones as starting vector.

Although we state the value of the relative error

$$\mathrm{err}_{\mathrm{rel}}(\alpha) := \|\boldsymbol{c_e}(A) - \boldsymbol{c_\alpha}(A)\|/\|\boldsymbol{c_e}(A)\|$$

for every choice of α, the conclusions of our tests are based on correlation coefficients between the rankings arising from $\boldsymbol{c_e}$ and $\boldsymbol{c_\alpha}$. We compute three types of correlation coefficients: Kendall's τ [97], Spearman's ρ [138], and Pearson's r [127]; see [131, Chap. 14] for a summary of them. The first is a popular statistic used to measure the association between rankings of objects by counting the numbers of concordant and discordant pairs of elements. Similarly Spearman's ρ is a nonparametric statistic indicative of whether the relation between two sets of elements can be expressed as a monotonic function. Spearman's ρ is better suited to lists with repeated values, and hence equal ranks. We also report the values of Pearson's r statistic. It serves as a test for linear dependence which, while not of immediate interest when comparing rankings, can still provide some indication as to how the different centrality measures relate. All three statistics take real values in the interval $[-1, 1]$, where 1 indicates perfect agreement and -1 indicates perfect disagreement between the objects.

In practice, only the top ranked nodes may need to be identified and there are several ways of reflecting this in the reported correlation coefficients. One of them is to apply a weighting to the test statistics, so that a disagreement of the best ranked nodes results in a lower than usual correlation coefficient. For example, Langville and Meyer suggest a weighted version of Spearman's ρ [104]. Another

alternative, also described in [104], is to tune the statistics to take into account that the compared lists are only partial. We will use the standard forms of the statistics to compute the correlation coefficients between the rankings obtained from full centrality vectors, and also from the top ranked $k\%$ of the nodes.

As a representative notation, we will use $\tau_{0.05}(\boldsymbol{c_e}(A), \boldsymbol{c_{\alpha_{\min}}}(A))$ to mean Kendall's correlation coefficient between the exponential- and resolvent-based rankings of the top 5% of the nodes of network A, obtained using the Katz parameter $\alpha_{\min}$.

To check the reliability of the centralities computed in floating point arithmetic we compute the quantity

$$\mathrm{err}_{\mathrm{dir}} = \left(1 - \frac{\widehat{\boldsymbol{x}}^{\boldsymbol{T}}\boldsymbol{x_q}}{\|\widehat{\boldsymbol{x}}\|_2\|\boldsymbol{x_q}\|_2}\right)^{1/2} \equiv (1-\cos\theta)^{1/2},$$

where θ is the angle between the computed $\widehat{\boldsymbol{x}}$ and a reference vector $\boldsymbol{x_q}$ computed in quadruple precision. We compute $\boldsymbol{x_q}$ using the Advanpix Multiprecision Computing Toolbox for MATLAB [3], which has very efficient IEEE 754-2008-compliant quadruple precision arithmetic. We actually compute $\mathrm{err}_{\mathrm{dir}}$ from the alternative formula

$$\mathrm{err}_{\mathrm{dir}} = \frac{1}{\sqrt{2}}\left\| \frac{\widehat{\boldsymbol{x}}}{\|\widehat{\boldsymbol{x}}\|_2} - \frac{\boldsymbol{x_q}}{\|\boldsymbol{x_q}\|_2}\right\|_2,$$

which is more accurately evaluated in floating point arithmetic. A value $\mathrm{err}_{\mathrm{dir}}$ of order u indicates that the computed and reference solutions are parallel to working precision. For each network we also compute the 1-norm condition number $\kappa_1(I-\alpha A)$ for each α, or, for the three largest networks, an estimate of the condition number computed using the MATLAB function `condest`, which implements the algorithm of [87].

We will use five examples of real networks available in the literature and one which is new and consists of recorded communication on the social networking platform Twitter. Table 2.1 summarizes the basic features for each network, including the spectral radius λ_1 and the eigenvalue with next largest real part, λ_2. We also give the condition number $\kappa_2(V)$ of the eigenvector matrix of A. For the undirected networks $\kappa_2(V) = 1$. For the largest network Strathclyde MUFC we compute the condition number of the rectangular matrix of eigenvectors corresponding to the 100 eigenvalues with largest real parts. Figures 2.1–2.5 show the sparsity patterns

Table 2.1: Basic characteristics of test networks. "Sparsity" denotes the proportion of nonzeros.

Name	Nodes	Edges	Sparsity	Directed	Weighted
Karate	34	78	1.3e-2	No	No
p53	133	558	3.2e-2	Yes	No
Minnesota	2642	3303	9.4e-4	No	Yes
ca-CondMat	23133	93497	3.5e-4	No	No
ca-AstroPh	18772	198110	1.1e-3	No	No
Strathclyde MUFC	148918	193032	8.7e-6	Yes	Yes

Name	λ_1	λ_2	$\kappa_2(V)$
Karate	6.7257	4.9771	1
p53	5.4032	$2.0696 + 0.2998i$	$\geq$ 1e16
Minnesota	3.2324	3.2319	1
ca-CondMat	37.9541	30.6438	1
ca-AstroPh	94.4415	75.5007	1
Strathclyde MUFC	41.1511	34.2307	5.7e2

and/or eigenvalue distributions, as appropriate. Sparsity plots for networks ca-CondMat and ca-AstroPh are omitted as they lack a distinctive visual pattern, at least in the node orderings provided.

The correlation coefficients penalize heavily variations in the rankings, even though the centrality scores of some nodes may be very close together, and in this case the rankings may change greatly with small variations in α. Such sensitivity of the ordering can arise for networks whose adjacency matrices have a very clustered spectrum or very ill conditioned eigenvectors. We give two such examples, the networks p53 and Minnesota.

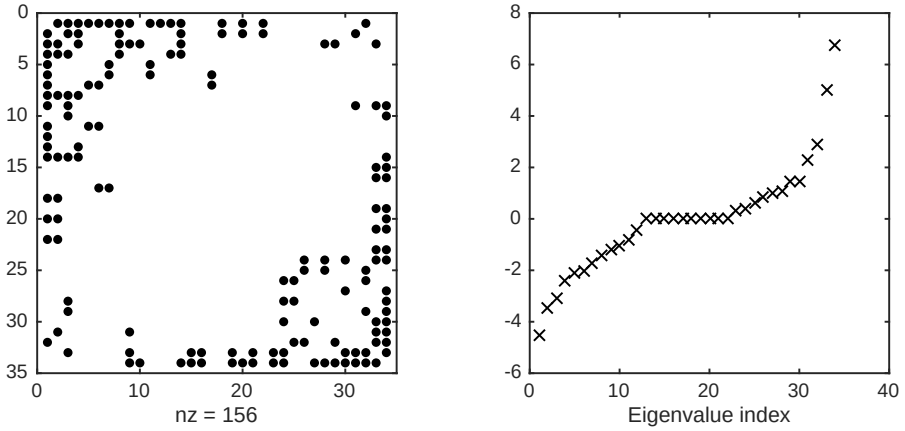


Figure 2.1: Sparsity and eigenvalue distribution plots for Zachary's Karate Club network.

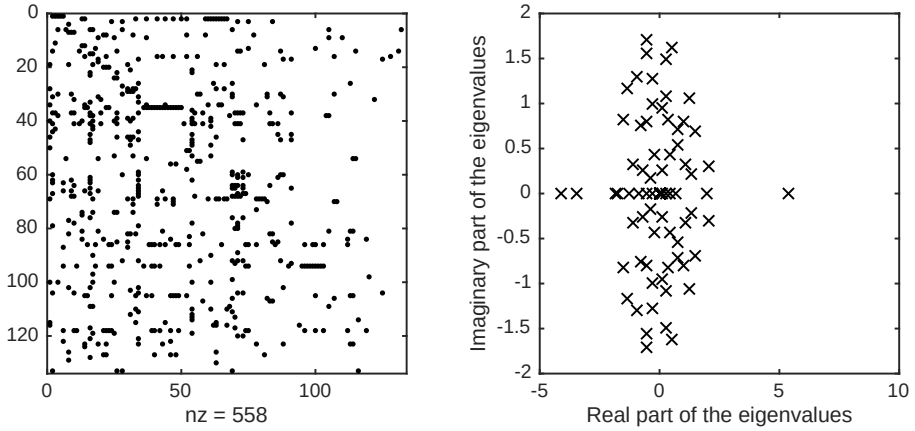


Figure 2.2: Sparsity and eigenvalue distribution plots for the p53 network.

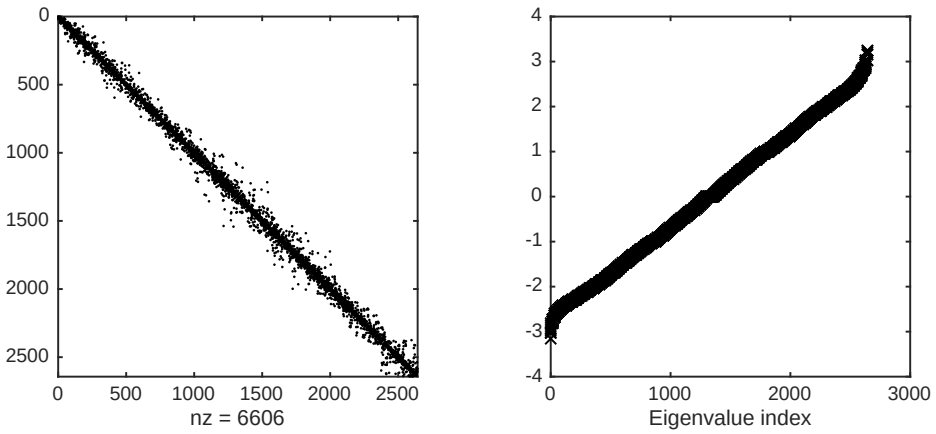


Figure 2.3: Sparsity and eigenvalue distribution plots for the Minnesota network.

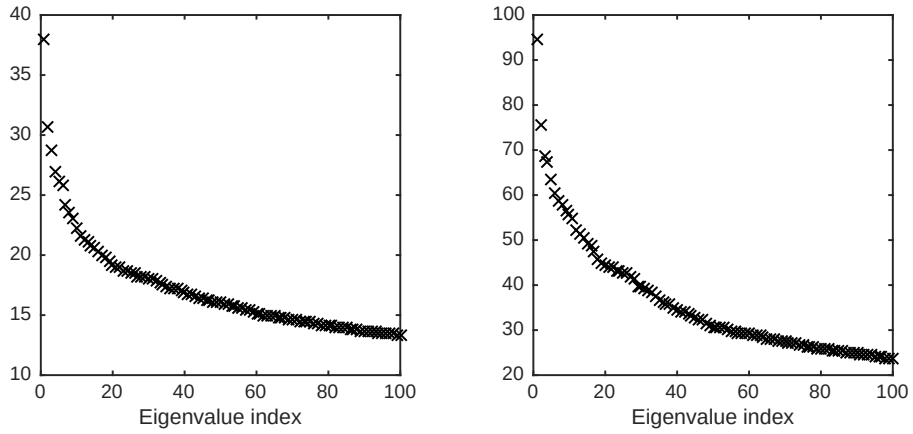


Figure 2.4: Eigenvalue distribution (100 largest positive) plots for the ca-CondMat (left) and ca-AstroPh (right) networks.

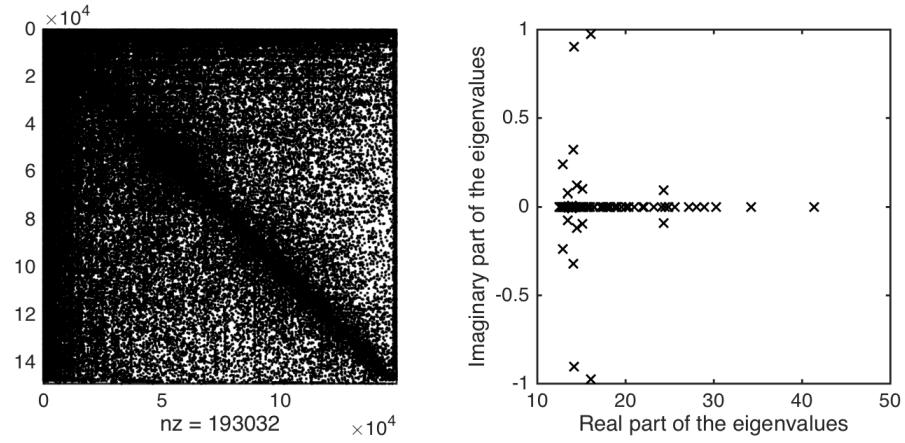


Figure 2.5: Sparsity and eigenvalue distribution (100 with largest real part) plots for the Strathclyde MUFC network.

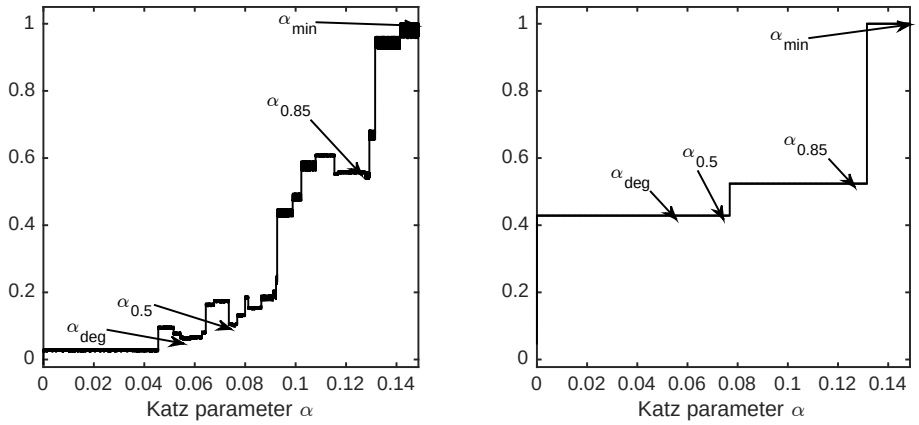


Figure 2.6: Kendall correlation coefficients between node rankings obtained from $\boldsymbol{c_e}(A)$ and $\boldsymbol{c_\alpha}(A)$ for different α for Zachary's karate network with all nodes (left) and top 20% of nodes (right).

First we consider a rather well studied example, Zachary's Karate Club [148]. The network is of dimension 34 and the value of its largest eigenvalue λ_1 is 6.7257. The values of the different Katz parameters and the respective correlation coefficients between the rankings arising from $\boldsymbol{c_\alpha}(A)$ and $\boldsymbol{c_e}(A)$ are summarized in Table 2.2. We observed that all the choices for the Katz parameter agree on the best ranked 5% of the nodes. However this is not very indicative since the network has only 34 elements, so we have shown instead how the parameters perform on the top 20% (7 out of the 34) of the nodes and all the nodes. All four choices for the Katz parameter yield node rankings that are positively correlated with the exponential result. For $\alpha_{\min}$, $\alpha_{0.5}$ and $\alpha_{\deg}$ the correlation between the top ranked 20% of the nodes is stronger than between the full rankings. On the contrary $\alpha_{0.85}$ matches the full rankings better than the partial ones for Kendall's τ and Pearson's r. This observation emphasizes the sensitivity of node rankings to the exact choice of Katz parameter. For the Karate Club network the resolvent-based measure evaluated with $\alpha_{\min}$ yields node rankings identical to its exponential-based counterpart. Figure 2.6 shows the dependence of both $\tau_1(\boldsymbol{c_e}(A), \boldsymbol{c_\alpha}(A))$ and $\tau_{0.20}(\boldsymbol{c_e}(A), \boldsymbol{c_\alpha}(A))$ on different values of the Katz parameter α.

Table 2.2: Correlation coefficients between node rankings (all nodes and top 20%) obtained from exponential-based centrality and resolvent centralities computed using $\alpha_{\min}$, $\alpha_{0.5}$, $\alpha_{0.85}$, and $\alpha_{\deg}$ applied to Zachary's Karate Club network.

Katz parameter	$\tau_{0.20}$	τ_1	$\rho_{0.20}$	ρ_1	$r_{0.20}$	r_1
$\alpha_{\min} = 0.1485$	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
$\alpha_{0.5} = 0.0743$	0.4286	0.1052	0.6429	0.1419	0.1546	0.1474
$\alpha_{0.85} = 0.1264$	0.5238	0.5579	0.6786	0.5866	0.2619	0.5866
$\alpha_{\deg} = 0.0556$	0.4286	0.0624	0.6429	0.0845	0.1489	0.0845

Katz parameter	err_{rel}	$\kappa_1(I - \alpha A)$	err_{dir}
$\alpha_{\min} = 0.1485$	0.0059	5.5e3	1.3e-16
$\alpha_{0.5} = 0.0743$	0.9976	7.1e0	1.9e-16
$\alpha_{0.85} = 0.1264$	0.9920	3.8e1	1.8e-16
$\alpha_{\deg} = 0.0556$	0.9981	4.5e0	1.5e-16

Next we consider a network consisting of 133 nodes arising from recorded levels of the oncogene p53 [144]. The network has 558 directed unweighted edges and an edge from node i to node j exists if i expresses above its usual level while j expresses below its usual level. The p53 network is part of the NESSIE collection of networks [141]. Correlation coefficients between the resolvent- and exponential-based measures are summarized in Table 2.3 and their variation with α can be seen in Figure 2.7. For both the best 10% of the nodes (top 14 out of the total 133 nodes) and all nodes, $\alpha_{\min}$ performs better than the other available options for the Katz parameter. The resolvent-based measure with $\alpha_{\min}$ and the exponential-based measure yield identical rankings for the top-ranked 8 nodes of this network. The parameter based on maximum node degree, $\alpha_{\deg}$, produces partial ranking negatively correlated to the exponential-based one. We note that the eigenvector matrix of A for this network is extremely ill conditioned, but nevertheless $\text{err}(\alpha_{\min})$ is close to being minimal: $|\min_\alpha \text{err}(\alpha) - \text{err}(\alpha_{\min})| / \min_\alpha \text{err}(\alpha) \approx$ 9e-5. For this network, then, our strategy of minimizing the distance between the exponential-based and resolvent-based centrality vectors does not result in the best correlation possible.

Table 2.3: Correlation coefficients between node rankings (all nodes and top 10%) obtained from exponential-based centrality and resolvent centralities computed using $\alpha_{\min}$, $\alpha_{0.5}$, $\alpha_{0.85}$, and $\alpha_{\deg}$ applied to the p53 network.

Katz parameter	$\tau_{0.10}$	τ_1	$\rho_{0.10}$	ρ_1	$r_{0.10}$	r_1
$\alpha_{\min} = 0.1842$	0.4066	0.3830	0.3978	0.4353	0.4225	0.3665
$\alpha_{0.5} = 0.0925$	0.0769	0.0980	0.1033	0.1608	0.1188	0.1462
$\alpha_{0.85} = 0.1573$	0.2088	0.2796	0.2044	0.3645	0.2131	0.3645
$\alpha_{\deg} = 0.0435$	0.0110	0.1107	−0.0110	0.1333	−0.0606	0.1333

Katz parameter	$\mathrm{err}_{\mathrm{rel}}$	$\kappa_1(I - \alpha A)$	$\mathrm{err}_{\mathrm{dir}}$
$\alpha_{\min} = 0.1842$	0.0215	4.2e3	1.8e-16
$\alpha_{0.5} = 0.0925$	0.9924	1.4e1	1.4e-16
$\alpha_{0.85} = 0.1573$	0.9713	9.8e1	1.9e-16
$\alpha_{\deg} = 0.0435$	0.9955	4.3e0	9.5e-17

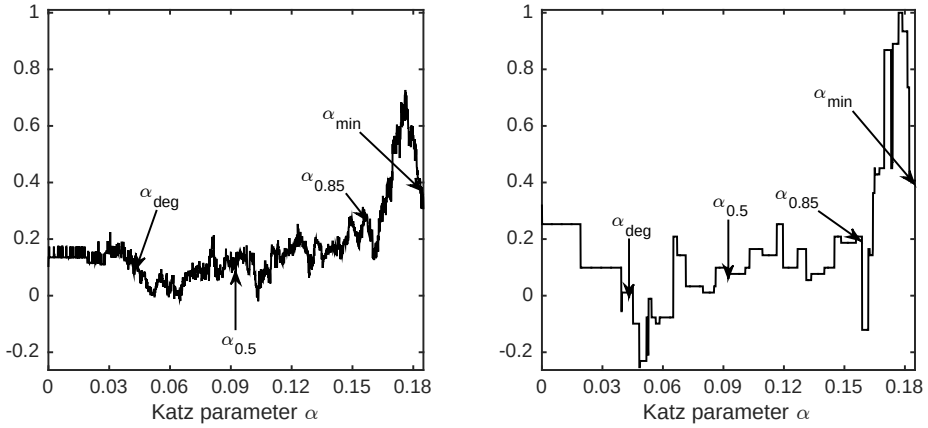


Figure 2.7: Kendall correlation coefficients between node rankings obtained from $\boldsymbol{c_e}(A)$ and $\boldsymbol{c_\alpha}(A)$ for different α, for network p53 with all nodes (left) and top 10% of nodes (right).

Table 2.4: Correlation coefficients between node rankings (all nodes and top 1%) obtained from exponential-based centrality and resolvent centralities computed using $\alpha_{\min}$, $\alpha_{0.5}$, $\alpha_{0.85}$, and $\alpha_{\deg}$ applied to the Minnesota network.

Katz parameter	$\tau_{0.01}$	τ_1	$\rho_{0.01}$	ρ_1	$r_{0.01}$	r_1
$\alpha_{\min} = 0.2972$	−0.0313	0.0089	−0.0885	0.0134	−0.1510	0.0134
$\alpha_{0.5} = 0.1547$	0.0199	0.0656	0.0208	0.0976	−0.1306	0.0976
$\alpha_{0.85} = 0.2630$	−0.0199	0.0429	−0.0440	0.0646	−0.0748	0.0646
$\alpha_{\deg} = 0.1667$	−0.0370	0.0486	−0.0556	0.0712	−0.1548	0.0712

Katz parameter	err_{rel}	$\kappa_1(I - \alpha A)$	err_{dir}
$\alpha_{\min} = 0.2972$	0.5748	1.2e2	3.3e-16
$\alpha_{0.5} = 0.1547$	0.8926	4.3e0	4.0e-16
$\alpha_{0.85} = 0.2630$	0.7600	2.3e0	1.6e-16
$\alpha_{\deg} = 0.1667$	0.8858	4.9e0	5.3e-16

The third network we consider, Minnesota, reflects the road connections of Minnesota and is available from the University of Florida Sparse Matrix Collection (http://www.cise.ufl.edu/research/sparse/matrices/Gleich/minnesota.html). The network consists of 2642 nodes and 3303 undirected weighted edges. Correlation coefficients between the resolvent- and exponential-based measures are summarized in Table 2.4 and their variation with α can be seen in Figure 2.8. The exponential-based centrality scores for this network are very close together, and the correlation results show that in this case minimizing the distance between the exponential-based and resolvent-based centrality vectors may not yield highly correlated rankings. The ranking of the resolvent-based centrality scores changes significantly with very small variations in α due to the clustering of the eigenvalues of the adjacency matrix.

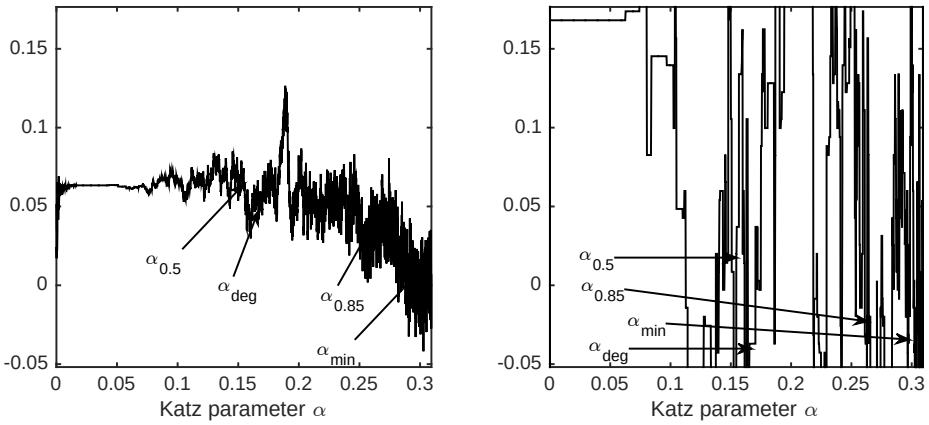


Figure 2.8: Kendall correlation coefficients between node rankings obtained from $\boldsymbol{c_e}(A)$ and $\boldsymbol{c_\alpha}(A)$ for different α for the Minnesota network with all nodes (left) and top 1% of nodes (right).

We next consider two undirected and unweighted networks, ca-AstroPh and ca-CondMat, which record research collaborations in the areas of condensed matter and astrophysics, respectively. Both networks are such that $a_{ij} = 1$ if and only if scholars i and j co-authored at least one publication. They are available from the Stanford Network Analysis Project (SNAP) [106] and are described by Leskovec et al. [107]. Correlation coefficients between the node rankings are presented in Tables 2.5 and 2.6. The dominance of $\alpha_{\min}$ is most appreciable when we compare only the top ranked 1% of the nodes. The alternative choices for the Katz parameter produce rankings which are weakly or even negatively correlated to the exponential ones. This is also true for other choices of the Katz parameter, as can be seen from Figures 2.9 and 2.10. We observed that the resolvent-based measure with $\alpha_{\min}$ and the exponential-based measure yield identical rankings for the top-ranked 113 nodes of ca-CondMat and top-ranked 161 nodes of ca-AstroPh.

Table 2.5: Correlation coefficients between node rankings (all nodes and top 1%) obtained from exponential-based centrality and resolvent centralities computed using $\alpha_{\min}$, $\alpha_{0.5}$, $\alpha_{0.85}$, and $\alpha_{\deg}$ applied to the ca-CondMat network.

Katz parameter	$\tau_{0.01}$	τ_1	$\rho_{0.01}$	ρ_1	$r_{0.01}$	r_1
$\alpha_{\min} = 0.0263$	0.8848	0.4158	0.9422	0.5020	0.9335	0.5020
$\alpha_{0.5} = 0.0132$	−0.0340	0.0171	−0.0476	0.0252	−0.0536	0.0252
$\alpha_{0.85} = 0.0224$	0.0358	0.0022	0.0615	0.0030	0.0344	0.0030
$\alpha_{\deg} = 0.0036$	−0.0299	0.0168	−0.0044	0.0248	−0.0427	0.0248

Katz parameter	err_{rel}	$\kappa_1(I - \alpha A)$	err_{dir}
$\alpha_{\min} = 0.0263$	0.8015	4.6e16	1.3e-15
$\alpha_{0.5} = 0.0132$	1.0000	3.5e1	7.9e-13
$\alpha_{0.85} = 0.0224$	1.0000	2.6e2	5.3e-14
$\alpha_{\deg} = 0.0036$	1.0000	4.2e0	6.5e-13

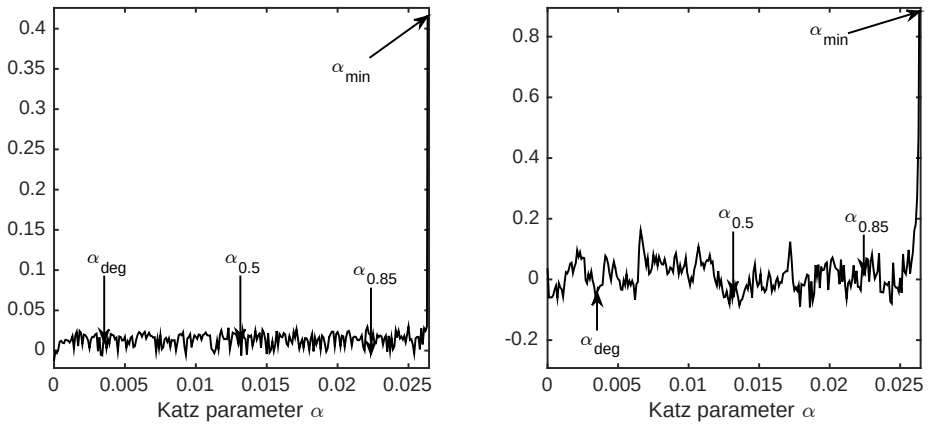


Figure 2.9: Kendall correlation coefficients between node rankings obtained from $\boldsymbol{c_e}(A)$ and $\boldsymbol{c_\alpha}(A)$ for different α, for network ca-CondMat with all nodes (left) and top 1% of nodes (right).

Table 2.6: Correlation coefficients between node rankings (all nodes and top 1%) obtained from exponential-based centrality and resolvent centralities computed using $\alpha_{\min}$, $\alpha_{0.5}$, $\alpha_{0.85}$, and $\alpha_{\deg}$ applied to the ca-AstroPh network.

Katz parameter	$\tau_{0.01}$	τ_1	$\rho_{0.01}$	ρ_1	$r_{0.01}$	r_1
$\alpha_{\min} = 0.0106$	0.9573	0.8283	0.9551	0.9826	0.9548	0.8728
$\alpha_{0.5} = 0.0053$	0.0686	−0.0012	0.1042	−0.025	0.1087	−0.0248
$\alpha_{0.85} = 0.0090$	0.0282	0.0198	0.0427	0.0289	0.0372	0.0244
$\alpha_{\deg} = 0.0020$	0.0162	0.0139	0.0216	0.0207	0.0300	0.0263

Katz parameter	err_{rel}	$\kappa_1(I - \alpha A)$	err_{dir}
$\alpha_{\min} = 0.0106$	1.0000	2.8e16	1.3e-15
$\alpha_{0.5} = 0.0053$	1.0000	2.3e1	8.6e-13
$\alpha_{0.85} = 0.0090$	1.0000	1.8e2	2.7e-13
$\alpha_{\deg} = 0.0020$	1.0000	4.4e0	7.6e-13

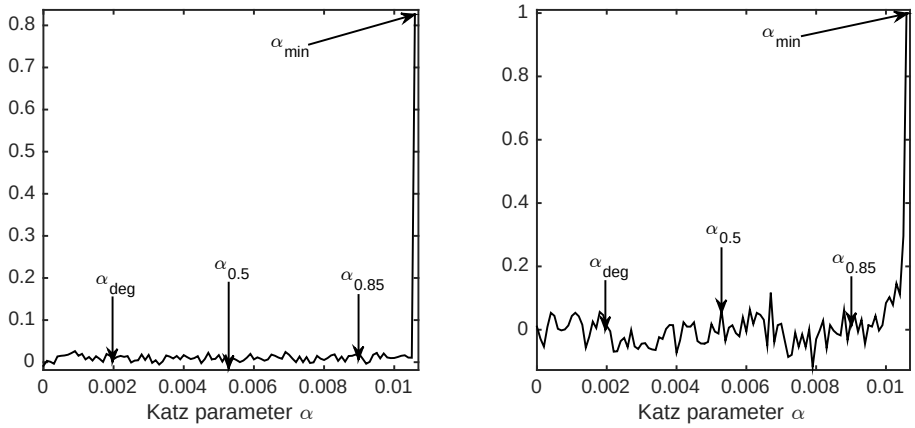


Figure 2.10: Kendall correlation coefficients between node rankings obtained from $\boldsymbol{c_e}(A)$ and $\boldsymbol{c_\alpha}(A)$ for different α, for network ca-AstroPh with all nodes (left) and top 1% of nodes (right).

The final real-world network example arises from the online social networking service Twitter. It consists of 148918 nodes and 193032 directed edges. Unlike the previous examples, this network has edges with nonnegative integer weights. The weight of an edge from node i to node j specifies how many times Twitter account i sent a (meaningful) communication to Twitter account j on the newsworthy topic of Sir Alex Ferguson's retirement from his position as manager of Manchester United Football Club in May 2013. The individual time-stamped interactions are available via the Strathclyde MUFC Twitter Data Set at http://www.mathstat.strath.ac.uk/outreach/twitter/mufc, and have also been studied in [75]. Our network was built by aggregating the tweets over the 12 hour period.

We consider ranking the nodes of the network and its transpose, where the top ranked nodes of Strathclyde MUFC and its transpose represent the best broadcasters and receivers, respectively, of information.

Correlation coefficients between the rankings of the nodes of Strathclyde MUFC and its transpose obtained using resolvent- and exponential-based measures are presented in Tables 2.7 and 2.8, respectively. For the correlations obtained using all the values of the Katz parameter, except $\alpha_{\min}$, we observe that the full rankings are matched significantly better than the partial ones. So $\alpha_{0.5}$, $\alpha_{0.85}$, and $\alpha_{\deg}$ more successfully retrieve the position of the lower ranked nodes. This is usually of less practical interest, especially for the case of very large networks. Only $\alpha_{\min}$ is able to successfully match a greater part of the highly ranked nodes, both in their broadcaster and receiver capacities. To be precise, the resolvent-based measure with $\alpha_{\min}$ and the exponential-based measure yield identical rankings for the top-ranked 91 broadcasters and top-ranked 128 receivers. Figures 2.11 and 2.12 illustrate the variation of the node ranking with respect to parameter α.

Finally, we note that for every network the values of $\mathrm{err}_{\mathrm{dir}}$ are all less than 10^{-12}. Even though by its definition $\alpha_{\min}$ tends to lead to more ill conditioned systems than the other choices of α, it produced values of $\mathrm{err}_{\mathrm{dir}}$ that were sometimes the smallest over all choice of α and never the largest. Our experiments therefore support the conclusions drawn from the analysis of inverse iteration in Section 2.3 that ill conditioning does not vitiate the rankings obtained.

Table 2.7: Correlation coefficients between node rankings (all nodes and top 1%) obtained from exponential-based broadcaster centrality and resolvent broadcaster centralities computed using $\alpha_{\min}$, $\alpha_{0.5}$, $\alpha_{0.85}$, and $\alpha_{\deg}$ applied to the Strathclyde MUFC network.

Katz parameter	$\tau_{0.01}$	τ_1	$\rho_{0.01}$	ρ_1	$r_{0.01}$	r_1
$\alpha_{\min} = 0.0242$	0.7850	0.6939	0.8959	0.7558	0.8893	0.7558
$\alpha_{0.5} = 0.0121$	0.0257	0.4524	0.0287	0.5419	0.0150	0.5419
$\alpha_{0.85} = 0.0205$	0.0512	0.4523	0.0620	0.5423	0.0393	0.5423
$\alpha_{\deg} = 0.0003$	0.0317	0.4467	0.0210	0.5401	0.0023	0.5401

Katz parameter	$\mathrm{err}_{\mathrm{rel}}$	$\kappa_1(I - \alpha A)$	$\mathrm{err}_{\mathrm{dir}}$
$\alpha_{\min} = 0.0242$	0.9997	6.9e17	1.1e-14
$\alpha_{0.5} = 0.0121$	1.0000	1.5e3	1.5e-13
$\alpha_{0.85} = 0.0205$	1.0000	1.2e4	8.9e-13
$\alpha_{\deg} = 0.0003$	1.0000	1.8e0	1.4e-13

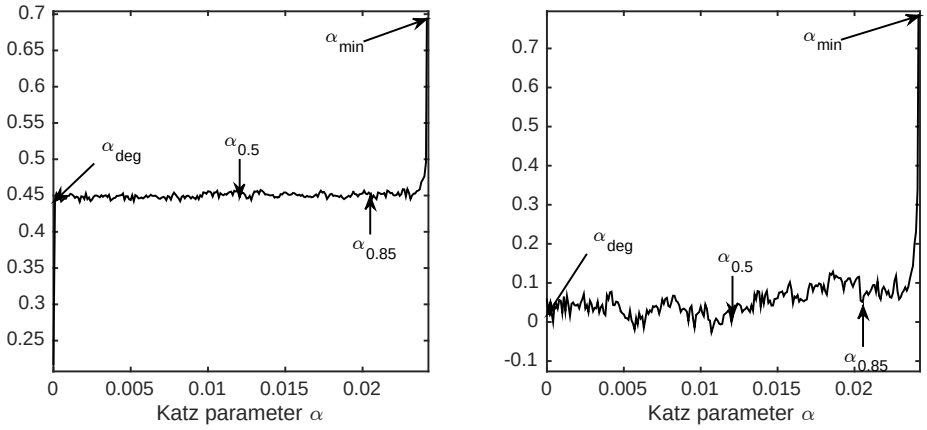


Figure 2.11: Kendall correlation coefficients between node rankings obtained from $\boldsymbol{c_e}(A)$ and $\boldsymbol{c_\alpha}(A)$ for different α for network Strathclyde MUFC, with all nodes (left) and top 1% of nodes (right).

Table 2.8: Correlation coefficients between node rankings (all nodes and top 1%) obtained from exponential-based receiver centrality and resolvent receiver centralities computed using $\alpha_{\min}$, $\alpha_{0.5}$, $\alpha_{0.85}$, and $\alpha_{\deg}$ applied to the transpose of the Strathclyde MUFC network.

Katz parameter	$\tau_{0.01}$	τ_1	$\rho_{0.01}$	ρ_1	$r_{0.01}$	r_1
$\alpha_{\min} = 0.0242$	0.7188	0.7015	0.7735	0.7529	0.7714	0.7529
$\alpha_{0.5} = 0.0121$	0.0237	0.5263	0.0340	0.6287	0.0253	0.6287
$\alpha_{0.85} = 0.0205$	0.0409	0.5405	0.0617	0.6336	0.0563	0.6336
$\alpha_{\deg} = 0.0001$	0.0741	0.5365	0.1407	0.6328	0.0991	0.6328

Katz parameter	err_{rel}	$\kappa_1(I - \alpha A)$	err_{dir}
$\alpha_{\min} = 0.0242$	0.9985	6.5e18	2.6e-14
$\alpha_{0.5} = 0.0121$	1.0000	1.5e4	1.9e-13
$\alpha_{0.85} = 0.0205$	1.0000	1.1e5	5.8e-13
$\alpha_{\deg} = 0.0001$	1.0000	1.6e1	2.6e-14

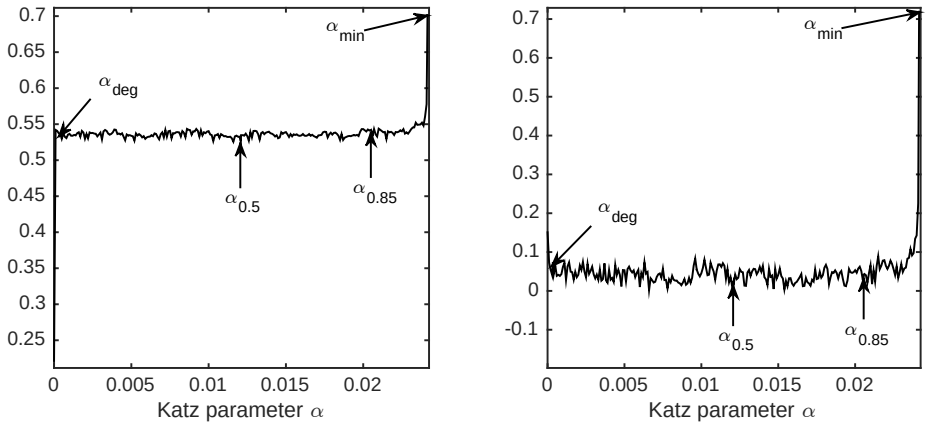


Figure 2.12: Kendall correlation coefficients between node rankings obtained from $\boldsymbol{c_e}(A^T)$ and $\boldsymbol{c_\alpha}(A^T)$ for different α for transpose of network Strathclyde MUFC, with all nodes (left) and top 1% of nodes (right).

2.5 Computational considerations

We now discuss some aspects of the computation of the exponential- and resolvent-based measures, and consider potential advantages and challenges associated with a few different methods.

First, note that to compute the exponential-based centrality scores $\boldsymbol{c}_{\boldsymbol{e}}(A) = \mathrm{e}^A \mathbf{1}$ we do not require the full matrix exponential: just the action of this matrix function on a vector. This opens up the possibility of using algorithms based on matrix–vector products involving A (and optionally A^T). For our numerical examples in Section 2.4 we used Al-Mohy and Higham's `expmv` algorithm for computing the action of the matrix exponential [6], which takes advantage of the nonnegativity of A in its norm estimation phase. It is based on scaling the matrix and applying truncated Taylor series approximations. There is a wide range of alternative algorithms based on Krylov subspace projection techniques [4], [88], [137]. A comparison of available algorithms for computing the action of the matrix exponential on a vector, along with a new implementation of a method based on Leja interpolation, is presented by Caliari et al. [34]. Available software is surveyed by Higham and Deadman [82].

For the resolvent-based measure the computation of $\alpha_{\min}$ requires the computation of λ_1. This can be done, for example, with the power method or the Arnoldi method. These and other methods (see [17] for a broad overview) are able to exploit one or more of the properties of sparsity, nonnegativity, and symmetry associated with adjacency matrices. There has also been interest in approximating λ_1 for directed or undirected, unweighted networks [132].

The resolvent-based centrality vector satisfies the linear system $(I - \alpha A)\boldsymbol{x} = \mathbf{1}$, which can be solved using direct solvers [52]. Iterative methods are not likely to be successful due to the potential ill conditioning of the coefficient matrix. For undirected networks the linear system is symmetric, which can of course be exploited. For directed networks usually both the receiver and broadcaster scores of the nodes are of interest. A matrix decomposition can be re-used to compute both $\boldsymbol{c}_{\boldsymbol{\alpha}}(A)$ and $\boldsymbol{c}_{\boldsymbol{\alpha}}(A^T)$.

There are also established resolvent-based methods for computing centrality scores for the nodes of temporally evolving networks [73], [72]. To see how resolvent-based centrality extends naturally to the case of a time-ordered network sequence, let $A^{[0]}, A^{[1]}, A^{[2]}, \ldots, A^{[M]}$ represent nonnegative integer-valued adjacency matrices for a fixed set of nodes. So, over a discrete time sequence, $t_0 < t_1 < \cdots < t_M$, the matrix $A^{[k]}$ records interactions at time t_k. For example, in the social interaction context, we may have $(A^{[k]})_{ij} = 1$ if individual i contacted individual j at least once in the time period $(t_{k-1}, t_k]$ and $(A^{[k]})_{ij} = 0$ otherwise. In this setting, there is a natural concept of *dynamic walks* through the network—such traversals may use whatever edges are available at one time and then continue at the next time point using the new edge list. The time-ordered product of resolvents

$$(I - \alpha A^{[0]})^{-1}(I - \alpha A^{[1]})^{-1}(I - \alpha A^{[2]})^{-1} \cdots (I - \alpha A^{[M]})^{-1}$$

collects together weighted counts of such dynamic walks, where a walk using k edges is downweighted by α^k. This combinatorial interpretation relies on the index law $\alpha^m \times \alpha^n = \alpha^{m+n}$, which allows walks to be correctly pieced together across time points. By contrast, the product of matrix exponentials

$$\mathrm{e}^{A^{[0]}}\mathrm{e}^{A^{[1]}}\mathrm{e}^{A^{[2]}} \cdots \mathrm{e}^{A^{[M]}}$$

does not allow this simple combinatorial interpretation, since $1/(m!) \times 1/(n!) \neq 1/((m+n)!)$ in general. Hence, from this dynamic walk perspective, the resolvent-based centrality is a more natural choice than the exponential alternative when we wish to extend to time-dependent interactions.

Finally, we present in Table 2.9 the time required to compute the node rankings of the larger real-world networks introduced in Section 2.4, using the exponential-based measure $\boldsymbol{c_e}(A)$ and the resolvent-based measure $\boldsymbol{c_\alpha}(A)$ with the value of the Katz parameter $\alpha_{\min}$. The tests were performed in MATLAB 2014b under Windows 7 on a machine with an Intel Xeon X5650 2.67Ghz 6-core processor and are averaged over ten runs. The total communicability $\boldsymbol{c_e}(A)$ was computed using both the function `expmv` from `http://eprints.ma.man.ac.uk/1591/`, which implements the algorithm of [6], and the function `funm_quad` from `http://guettel.com/funm_quad`, which implements the algorithm of [66], with a stopping accuracy 2^{-53}. The Katz

Table 2.9: Time (seconds) required to compute the centrality vectors $\boldsymbol{c_e}(A)$ and $\boldsymbol{c_\alpha}(A)$ using $\alpha_{\min}$ for the networks ca-CondMat, ca-AstroPh and Strathclyde MUFC from Sec. 2.4.

	ca-CondMat	ca-AstroPh	Strathclyde MUFC
$\boldsymbol{c_e}$: `expmv`	0.2882	0.7741	1.2812
$\boldsymbol{c_e}$: `funm_quad`	0.0838	0.1067	2.1942
$\boldsymbol{c_\alpha}$: `backslash`	0.6143	1.7607	0.8967

centrality $\boldsymbol{c_\alpha}(A)$ was computed using the MATLAB `backslash` function for sparse matrices; the time for computing the Katz centrality includes the time required to compute the parameter $\alpha_{\min}$ (which is done using `eigs`, as before). The Katz centrality is computed faster than the exponential-based centrality for the directed network Strathclyde MUFC, while for the two less sparse, undirected networks the exponential-based centrality is obtained more quickly. Table 2.9 shows that there is no ordering between the three methods. A more thorough investigation would be required to draw any general conclusions about the relative costs of computing $\boldsymbol{c_e}$ and $\boldsymbol{c_\alpha}$.

CHAPTER 3

The Matrix Unwinding Function

3.1 Introduction

Previous work on the matrix logarithm [80, Chap. 11] made use of a scalar function called the unwinding number. In this work we define and investigate the corresponding primary matrix function. We show that the matrix unwinding function is just as useful as its scalar counterpart. First, it is a valuable tool for stating matrix identities involving the matrix logarithm and fractional matrix powers, because it elegantly prescribes the correction needed when identities that are generalized from the real case break down. For example, the unwinding function neatly captures the difference between $\log A^\alpha$ and $\alpha \log A$. In addition to this role as a theoretical tool, the unwinding function is also useful computationally. It enables us to preprocess a matrix so that its eigenvalues all have imaginary parts lying in the interval $(-\pi, \pi]$, while not changing the exponential of the matrix. We show that this matrix argument reduction can provide a large decrease in norm and can thereby lead to significant computational savings when the scaling and squaring method is used to evaluate the matrix exponential. We pursue argument reduction for the matrix exponential in Section 5.2.1.

We define the scalar unwinding number in the next section and recap some of its key properties. The matrix unwinding function $\mathcal{U}(A)$ is defined in Section 3.3, where we deal carefully with a subtlety concerning the meaning of the derivative at points with imaginary parts, which are odd integer multiples of π. Basic properties

of $\mathcal{U}(A)$ are derived in Section 3.3.1. Bounds for the norm and the condition number of $\mathcal{U}(A)$ are given in Section 3.3.2, where we also discuss the estimation of the condition number. In Section 3.3.3 we derive a number of matrix identities involving the functions $\log z$ and z^α. Connections with the matrix sign function are explored in Section 3.3.4. In Section 3.4 we give a Schur–Parlett algorithm for computing $\mathcal{U}(A)$ based on a certain reordering of the Schur form specific to the unwinding function. We also give some analysis connecting the conditioning of the Sylvester equations underlying the Parlett recurrence to the conditioning of $\mathcal{U}$. In Section 3.5 we show via numerical experiments that the algorithm performs well in practice.

3.2 The unwinding number

We first state our conventions for three key functions of a complex variable:

(i) arg is the principal argument: $-\pi < \arg z \le \pi$.

(ii) log is the principal logarithm: $-\pi < \operatorname{Im} \log z \le \pi$.

(iii) For $\alpha, z \in \mathbb{C}$ we define $z^\alpha = \mathrm{e}^{\alpha \log z}$. In particular, $z^{1/2}$ is the principal square root: $\operatorname{Re} z^{1/2} \ge 0$ and $(-1)^{1/2} = i$.

Motivation for these particular choices for the values of the functions on their branch cuts is given by Kahan [95]. We will use repeatedly the key properties $\mathrm{e}^{\log z} = z$ and $\mathrm{e}^{z_1+z_2} = \mathrm{e}^{z_1}\mathrm{e}^{z_2}$. The negative real axis will be denoted by $\mathbb{R}^-$.

The *unwinding number* of $z \in \mathbb{C}$ is defined by

$$\mathcal{U}(z) = \frac{z - \log \mathrm{e}^z}{2\pi i}. \tag{3.1}$$

The definition can be rewritten as

$$z = \log \mathrm{e}^z + 2\pi i \mathcal{U}(z), \tag{3.2}$$

so that $2\pi i \mathcal{U}(z)$ is the discrepancy between $\log e^z$ and z.

The term "unwinding number," with a definition differing from ours only in sign, first appeared in Corless and Jeffrey [45] and Jeffrey, Hare, and Corless [94].

A definition with the same sign as (3.1), and an explanation of why this sign is preferred, is given by Bradford, Corless, Davenport, Jeffrey, and Watt [27]. Related definitions can be found in Apostol [12, Thm. 1.48], Aslaksen [16], Bradford [26], and Patton [125]. With the exception of [12], in all these references the interest in the unwinding number stems from its suitability for use in computer algebra.

The following lemma from [45], [94] gives a formula for the unwinding number that is easier to evaluate than (3.1).

Lemma 3.1. *The unwinding number of $z \in \mathbb{C}$ can be expressed using the ceiling function as*

$$\mathcal{U}(z) = \left\lceil \frac{\operatorname{Im} z - \pi}{2\pi} \right\rceil. \tag{3.3}$$

Proof. Exponentiating both sides of (3.2) we have

$$\mathrm{e}^z = \mathrm{e}^{\log \mathrm{e}^z + 2\pi i \mathcal{U}(z)} = \mathrm{e}^z \mathrm{e}^{2\pi i \mathcal{U}(z)},$$

so $\mathrm{e}^{2\pi i \mathcal{U}(z)} = 1$, and hence $\mathcal{U}(z) \in \mathbb{Z}$. Taking imaginary parts in (3.2) gives $-\pi < \operatorname{Im} z - 2\pi\mathcal{U}(z) \le \pi$, which can be written

$$\frac{\operatorname{Im} z - \pi}{2\pi} \le \mathcal{U}(z) < \frac{\operatorname{Im} z + \pi}{2\pi}.$$

The result follows since $\mathcal{U}(z) \in \mathbb{Z}$. □

Thus $\mathcal{U}$ takes integer values and is constant for $\operatorname{Im} z$ on the intervals $((2k-1)\pi, (2k+1)\pi]$ for all integers k. It is therefore easy to characterize when $\mathcal{U}(z) = 0$, or equivalently, $\log \mathrm{e}^z = z$.

Corollary 3.2. *For $z \in \mathbb{C}$, $\mathcal{U}(z) = 0$ if and only if $\operatorname{Im} z \in (-\pi, \pi]$.*

We now consider some of the most useful properties of the unwinding number. In the formulas below it is implicitly understood that $z = 0$ is excluded from formulas involving $\log z$. We define $\mathcal{D}$ to be the open set comprising $\mathbb{C}$ with the lines on which $\mathcal{U}$ is discontinuous removed:

$$\mathcal{D} = \{\, z \in \mathbb{C} : \operatorname{Im} z \ne (2j+1)\pi \text{ for all } j \in \mathbb{Z} \,\}. \tag{3.4}$$

Lemma 3.3. *For $z \in \mathbb{C}$,*

$$\mathcal{U}(\overline{z}) = \mathcal{U}(-z) = \begin{cases} -\mathcal{U}(z), & z \in \mathcal{D}, \\ -\mathcal{U}(z) - 1, & \text{otherwise.} \end{cases}$$

Proof. We have

$$\begin{aligned}\mathcal{U}(\overline{z}) = \mathcal{U}(-z) &= \left\lceil \frac{\operatorname{Im}(-z) - \pi}{2\pi} \right\rceil \\ &= \left\lceil \frac{-\operatorname{Im}(z) - \pi}{2\pi} \right\rceil \\ &= \left\lceil -\left(\frac{\operatorname{Im}(z) - \pi}{2\pi} \right) - 1 \right\rceil.\end{aligned}$$

The result follows by noting that $\lceil -w - 1 \rceil = -\lceil w \rceil$ or $-\lceil w \rceil - 1$. ∎

Lemma 3.4. *For $z \in \mathbb{C}$ and $\alpha \in [-1, 1]$,*

$$\mathcal{U}(\alpha \log z) = \begin{cases} 0, & z \in \mathbb{C},\ \alpha \in (-1, 1] \text{ or } z \notin \mathbb{R}^-,\ \alpha = -1, \\ -1, & z \in \mathbb{R}^-,\ \alpha = -1. \end{cases}$$

Proof. Since $\operatorname{Im} \log z \in (-\pi, \pi]$, $\mathcal{U}(\alpha \log z) = 0$ for $\alpha \in (-1, 1]$ by Corollary 3.2. For $\alpha = -1$, $\mathcal{U}(-\log z) = 0$ unless $\operatorname{Im}(-\log z) = -\pi$, that is, $z \in \mathbb{R}^-$, in which case $\mathcal{U}(-\log z) = \mathcal{U}(-\pi i) = -1$. ∎

The next three results are some of the "useful theorems" that motivated the introduction of the unwinding number in [45]. They show that $\mathcal{U}$ provides the appropriate correction term in three important formulas involving the logarithm.

Lemma 3.5. *For $z_1, z_2 \in \mathbb{C}$, $\log(z_1 z_2) = \log z_1 + \log z_2 - 2\pi i \mathcal{U}(\log z_1 + \log z_2)$.*

Proof. From (3.2) we have

$$\begin{aligned}\log z_1 + \log z_2 &= \log(\mathrm{e}^{\log z_1 + \log z_2}) + 2\pi i \mathcal{U}(\log z_1 + \log z_2) \\ &= \log(\mathrm{e}^{\log z_1} \mathrm{e}^{\log z_2}) + 2\pi i \mathcal{U}(\log z_1 + \log z_2) \\ &= \log(z_1 z_2) + 2\pi i \mathcal{U}(\log z_1 + \log z_2),\end{aligned}$$

as required. ∎

Lemma 3.6. *For $\alpha, z \in \mathbb{C}$, $\log(z^\alpha) = \alpha \log z - 2\pi i \mathcal{U}(\alpha \log z)$.*

Proof. Using (3.2), we have

$$\log(z^\alpha) = \log(\mathrm{e}^{\alpha \log z}) = \alpha \log z - 2\pi i \mathcal{U}(\alpha \log z). \qquad ∎$$

Lemmas 3.4 and 3.6 together give that for $\alpha \in (-1, 1]$ the identity $\log(z^\alpha) = \alpha \log z$ holds. Note that $\alpha = 1/2$ yields the important special case $\log(z^{1/2}) = \frac{1}{2}\log z$. Lemmas 3.4 and 3.6 also give $\log(z^{-1}) = -\log z$ for $z \notin \mathbb{R}^-$.

Lemma 3.7. *For $z_1, z_2 \in \mathbb{C}$, $(z_1 z_2)^{1/2} = z_1^{1/2} z_2^{1/2} (-1)^{\mathcal{U}(\log z_1 + \log z_2)}$.*

Proof. Using Lemma 3.5 we have

$$\begin{aligned}
(z_1 z_2)^{1/2} &= \exp\left(\tfrac{1}{2}\log(z_1 z_2)\right) \\
&= \exp\left(\tfrac{1}{2}\left(\log z_1 + \log z_2 - 2\pi i\,\mathcal{U}(\log z_1 + \log z_2)\right)\right) \\
&= z_1^{1/2} z_2^{1/2} \exp(-\pi i\,\mathcal{U}(\log z_1 + \log z_2)) \\
&= z_1^{1/2} z_2^{1/2} (-1)^{\mathcal{U}(\log z_1 + \log z_2)}. \qquad \square
\end{aligned}$$

An important application of the unwinding number is in the accurate evaluation of elements of functions of triangular matrices. It is well known that for $\lambda_1 \neq \lambda_2$ [80, Sec. 4.6],

$$f\left(\begin{bmatrix} \lambda_1 & t_{12} \\ 0 & \lambda_2 \end{bmatrix}\right) = \begin{bmatrix} f(\lambda_1) & t_{12}\dfrac{f(\lambda_2) - f(\lambda_1)}{\lambda_2 - \lambda_1} \\ 0 & f(\lambda_2) \end{bmatrix},$$

but in floating point arithmetic evaluation of the $(1,2)$ element from this formula can incur subtractive cancellation when λ_1 is close to λ_2. Consider the case where $f = \log$. Let $z = (\lambda_2 - \lambda_1)/(\lambda_2 + \lambda_1)$ and assume that a means for accurate evaluation of atanh is available. The following formula suggested by Higham [80, Sec. 11.6.2] allows accurate evaluation when λ_1 and λ_2 are close, but not equal:

$$f_{12} = t_{12}\frac{2\,\mathrm{atanh}(z) + 2\pi i\,\mathcal{U}(\log \lambda_2 - \log \lambda_1)}{\lambda_2 - \lambda_1}.$$

This formula is used in `logm` in MATLAB. A similar formula is obtained by Higham and Lin [84, (5.6)] for $f(t) = t^p$, $p \in \mathbb{R}$ and is used in [84] and [85].

3.3 The matrix unwinding function

We define the *matrix unwinding function* to be the matrix function corresponding to the unwinding number:

$$\mathcal{U}(A) = \frac{A - \log \mathrm{e}^A}{2\pi i}, \qquad A \in \mathbb{C}^{n\times n}. \tag{3.5}$$

To make this definition precise we need to clarify which matrix logarithm is being used. We cannot use the usual principal matrix logarithm, for which $\log X$ is defined only for X with no eigenvalues on $\mathbb{R}^-$ [80, Thm. 1.31]. Instead we take log to be the matrix function corresponding to the principal scalar logarithm defined at the start of Section 3.2. However, this is not sufficient to define $\log A$ for any nonsingular matrix. To see why, consider the Jordan canonical form (1.1) of $A \in \mathbb{C}^{n\times n}$. The principal logarithm log is discontinuous on its branch cut $\mathbb{R}^-$ and so does not have any derivatives there. As explained in more generality in Section 5.3, we will define the first derivative for $z \in \mathbb{R}^-$ as the one-sided limit

$$\log'(z) = \lim_{h\to 0,\ \operatorname{Im} h\ge 0} [\log(z+h) - \log z]/h,$$

and so on for higher derivatives, which are simply the usual derivatives evaluated on $\mathbb{R}^-$. Hence $\log A$ is now well defined.

Another way to define $\mathcal{U}(A)$ that is equivalent to (3.5) is by applying the Jordan form definition directly to the scalar unwinding number $\mathcal{U}(z)$, where the derivatives $\mathcal{U}'(z)$, $\mathcal{U}''(z)$, $\ldots$, are necessarily zero for $\operatorname{Im} z \ne (2j+1)\pi$, $j \in \mathbb{Z}$, and we define them to be zero for $\operatorname{Im} z = (2j+1)\pi$. That this definition is equivalent to (3.5) follows from the fact that the underlying scalar functions have the same values on the spectrum of A [80, Sec. 1.2.2]. It is immediate from (1.3) that $\mathcal{U}(J_k(\lambda_k)) = \mathcal{U}(\lambda_k)I$ for any Jordan block $J_k(\lambda_k)$. Hence, in terms of the Jordan form (1.1),

$$\mathcal{U}(A) = Z \operatorname{diag}(\mathcal{U}(\lambda_k) I_{m_k}) Z^{-1}, \tag{3.6}$$

so that $\mathcal{U}(A)$ is diagonalizable and has integer eigenvalues. In particular, if all the eigenvalues of A have the same unwinding number u, then $\mathcal{U}(A) = uI$.

Note that $\mathcal{U}(z)$ is continuously differentiable as many times as we like in the open subset $\mathcal{D}$ of $\mathbb{C}$ in (3.4). This implies, for example, that $\mathcal{U}$ is a continuous matrix function on the set of matrices $A \in \mathbb{C}^{n\times n}$ with spectrum in $\mathcal{D}$ [80, Thm. 1.19] and that $\mathcal{U}$ is Fréchet differentiable on this set [80, Thm. 3.8]. However, for most of the results in this chapter we need just the following standard properties we discussed

in Section 1.1, and which hold for general matrix functions f:

$$\mathcal{U}(A) \text{ is a polynomial in } A, \tag{3.7a}$$

$$A, B \in \mathbb{C}^{n\times n},\ AB = BA \ \Rightarrow \begin{cases} \mathcal{U}(A)\mathcal{U}(B) = \mathcal{U}(B)\mathcal{U}(A), \\ \mathcal{U}(A+B) \text{ commutes with } A \text{ and } B. \end{cases} \tag{3.7b}$$

3.3.1 Properties of the unwinding function

We now derive some properties of the matrix unwinding function, and in particular generalize some of the properties of the unwinding number given in Section 3.2.

Theorem 3.8. *For $A \in \mathbb{C}^{n\times n}$, $\mathcal{U}(A) = 0$ if and only if the imaginary parts of all the eigenvalues of A lie in the interval $(-\pi, \pi]$.*

Proof. The result is immediate from (3.6) and Corollary 3.2. □

Note that $\mathcal{U}(A) = 0$ is equivalent to $\log \mathrm{e}^A = A$, and essentially the same conditions as in Theorem 3.8 for this equation to hold are proved in [80, Prob. 1.39] for the usual principal matrix logarithm without explicitly referring to the matrix unwinding function. The theorem implies that the spectral radius condition $\rho(A) < \pi$, or the stronger condition $\|A\| < \pi$ for some consistent matrix norm, are sufficient for $\log \mathrm{e}^A = A$ to hold. For several important classes of matrices the conditions of Theorem 3.8 are always satisfied: matrices with real eigenvalues (in particular, Hermitian matrices), and unitary, idempotent, or stochastic matrices (for all of which $|\lambda| \le 1$ for every eigenvalue λ).

The next result, which gives a characterization of a class of matrix functions of which the matrix unwinding function is a special case, enables us to determine the behavior of $\mathcal{U}$ under conjugation and the form of $\mathcal{U}$ for real matrices and pure imaginary matrices. We denote by $\Lambda(A)$ the spectrum of A.

Theorem 3.9. *Let f be analytic on an open subset $\Omega \subseteq \mathbb{C}$ such that for each connected component $\widetilde{\Omega}$ of Ω, $z \in \widetilde{\Omega}$ if and only if $-\overline{z} \in \widetilde{\Omega}$. Consider the corresponding matrix function f on its natural domain in $\mathbb{C}^{n\times n}$, and the set $\mathcal{S} = \{ A \in \mathbb{C}^{n\times n} : \Lambda(A) \subseteq \Omega \}$. Then the following are equivalent:*

(a) *$f(A^*) = f(-A)^*$ for all $A \in \mathcal{S}$.*

(b) $f(\overline{A}) = \overline{f(-A)}$ *for all* $A \in \mathcal{S}$.

(c) $f(i\mathbb{R}^{n\times n} \cap \mathcal{S}) \subseteq \mathbb{R}^{n\times n}$.

(d) $f(i\mathbb{R} \cap \Omega) \subseteq \mathbb{R}$.

Proof. Rewrite the characterization of [80, Thm. 1.18] or [86, Thm. 3.2] by replacing the function $f(z)$ therein by $f(iz)$. □

Applying Theorem 3.9 to the matrix unwinding function we obtain the next result.

Corollary 3.10. *For* $A \in \mathbb{C}^{n\times n}$ *with spectrum in* $\mathcal{D}$,

(a) $\mathcal{U}(A^*) = \mathcal{U}(-A)^* = -\mathcal{U}(A)^*$.

(b) $\mathcal{U}(\overline{A}) = \overline{\mathcal{U}(-A)} = -\overline{\mathcal{U}(A)}$.

(c) $\mathcal{U}(A)$ *is real if* A *is pure imaginary.*

(d) $\mathcal{U}(A)$ *is pure imaginary if* A *is real.*

Proof. Note first that the second equalities in (a) and (b) follow from Lemma 3.3 and (3.6). $\mathcal{U}$ is analytic on the open subset $\mathcal{D}$ of $\mathbb{C}$, which satisfies $z \in \widetilde{\mathcal{D}}$ if and only if $-\overline{z} \in \widetilde{\mathcal{D}}$ for each connected component $\widetilde{\mathcal{D}}$. Hence to prove the first equalities in (a) and (b), and (c), it suffices to show that any one of the statements in Theorem 3.9 holds. Indeed, for any $z \in i\mathbb{R} \cap \mathcal{D}$, $\mathcal{U}(z) \in \mathbb{R}$, which is condition (d) of Theorem 3.9. To show (d), we note that if A is real then $A = \overline{A}$, so $\mathcal{U}(A) = \mathcal{U}(\overline{A})$ and so from (b), $\mathcal{U}(A) = -\overline{\mathcal{U}(A)}$ and $\mathcal{U}(A)$ is pure imaginary. □

We give two examples to illustrate the corollary. First,

$$A = \begin{bmatrix} 4 & 16 \\ -4 & 4 \end{bmatrix}, \quad \Lambda(A) = \{4 \pm 8i\}, \quad \mathcal{U}(A) = \begin{bmatrix} 0 & -2i \\ 0.5i & 0 \end{bmatrix}, \quad \Lambda(\mathcal{U}(A)) = \{\pm 1\}.$$

Second, a matrix due to Rutishauser, which is `gallery('toeppen',3)` in MATLAB (non-integers are shown here to three significant figures):

$$A = \begin{bmatrix} 0 & 10 & 1 \\ -10 & 0 & 10 \\ 1 & -10 & 0 \end{bmatrix}, \qquad \Lambda(A) = \{1, -0.500 \pm 1.41i\},$$

$$\mathcal{U}(A) = i \begin{bmatrix} 0.0354 & -1.42 & -0.0354 \\ 1.42 & -0.0708 & -1.42 \\ -0.0354 & 1.42 & 0.0354 \end{bmatrix}, \qquad \Lambda(\mathcal{U}(A)) = \{-2, 0, 2\}.$$

In both cases, $\mathcal{U}(A)$ is pure imaginary and a further computation shows that $\mathcal{U}(A^*) = \mathcal{U}(-A)^*$.

We can give an explicit formula for the unwinding function of real 2×2 matrices of the form that appear as diagonal blocks in the real Schur decomposition computed by LAPACK.

Lemma 3.11. *For $A = \left[\begin{smallmatrix} a & b \\ c & a \end{smallmatrix}\right] \in \mathbb{R}^{2\times 2}$ with $bc < 0$,*

$$\mathcal{U}(A) = \begin{cases} -i\dfrac{\mathcal{U}(i\mu)}{\mu}(A - aI), & \mu \neq (2k+1)\pi,\ k \in \mathbb{Z}, \\ -\dfrac{i}{\mu}\left[(\mathcal{U}(i\mu) + \tfrac{1}{2})(A - aI) - \tfrac{1}{2}i\mu I\right], & \text{otherwise}, \end{cases} \tag{3.8}$$

where $\mu = (-bc)^{1/2}$.

Proof. The eigenvalues of A are $\lambda = a + i\mu$ and $\overline{\lambda}$. Let $Z^{-1}AZ = \operatorname{diag}(\lambda, \overline{\lambda}) = aI + i\mu K$, where $K = \left[\begin{smallmatrix} 1 & 0 \\ 0 & -1 \end{smallmatrix}\right]$. Thus $A = aI + \mu W$, where $W = iZKZ^{-1} \in \mathbb{R}^{2\times 2}$. Hence, for $\mu \neq (2k+1)\pi$ with $k \in \mathbb{Z}$,

$$\begin{aligned} \mathcal{U}(A) &= Z \operatorname{diag}(\mathcal{U}(\lambda), \mathcal{U}(\overline{\lambda}))Z^{-1} \\ &= \mathcal{U}(\lambda) Z \operatorname{diag}(1, -1) Z^{-1} = \mathcal{U}(\lambda) ZKZ^{-1} \\ &= \frac{\mathcal{U}(\lambda)}{i} W = -i\mathcal{U}(\lambda) W \\ &= -i\mathcal{U}(\lambda)(A - aI)/\mu. \end{aligned}$$

If $\mu = (2k+1)\pi$ for some $k \in \mathbb{Z}$, then $\mathcal{U}(\overline{\lambda}) = -\mathcal{U}(\lambda) - 1$, by Lemma 3.3. Hence we need to add a correction term $-Z \operatorname{diag}(0, 1) Z^{-1}$ to the formula above. This correction term can be written

$$-\frac{1}{2}Z(I-K)Z^{-1} = \frac{1}{2}(ZKZ^{-1} - I) = \frac{1}{2}\left(\frac{A - aI}{i\mu} - I\right) = -\frac{i}{2\mu}(A - aI - i\mu I). \qquad \square$$

The 2×2 example above illustrates the theorem.

Lemma 3.12. *For $A \in \mathbb{C}^{n\times n}$, $\mathrm{e}^{2\pi i \mathcal{U}(A)} = I$.*

Proof. Multiplying (3.5) by $2\pi i$ and exponentiating, and using the fact that $\log \mathrm{e}^A$ and $\mathcal{U}(A)$ commute, as they are both polynomials in A, we have

$$\mathrm{e}^A = \mathrm{e}^{\log \mathrm{e}^A + 2\pi i \mathcal{U}(A)} = \mathrm{e}^{\log \mathrm{e}^A} \mathrm{e}^{2\pi i \mathcal{U}(A)} = \mathrm{e}^A \mathrm{e}^{2\pi i \mathcal{U}(A)},$$

hence $\mathrm{e}^{2\pi i \mathcal{U}(A)} = I$. $\square$

3.3.2 Norm and conditioning

We now obtain an upper bound for the norm of $\mathcal{U}(A)$ and a lower bound for its condition number. These will be useful in Section 3.4 for understanding the behavior of an algorithm for computing $\mathcal{U}(A)$. The norm is any consistent norm for which $\|\operatorname{diag}(d_i)\| = \max_i |d_i|$, $\rho(A)$ denotes the spectral radius, and $\kappa(A) = \|A\|\|A^{-1}\|$ is the condition number with respect to inversion.

The following result is a special case of Lemma 5.8 on noting that the period of the exponential function is $p = 2\pi i$. Observe that we can use the definition of the unwinding number to give a sharper bound than the general one in Lemma 5.8.

Lemma 3.13. *For $A \in \mathbb{C}^{n\times n}$ with Jordan canonical form $A = ZJZ^{-1}$,*

$$\|\mathcal{U}(A)\| \le \frac{\kappa(Z)(\rho(A)+\pi)}{2\pi}.$$

Proof. Using (3.6) we have $\|\mathcal{U}(A)\| \le \kappa(Z)\max_k |U(\lambda_k)|$. But

$$\max_k |\mathcal{U}(\lambda_k)| = \max_k \left| \left\lceil \frac{\operatorname{Im}\lambda_k - \pi}{2\pi} \right\rceil \right| \le \frac{\rho(A)+\pi}{2\pi}. \qquad \square$$

Recall from Section 1.2 that the (relative) condition number of the matrix unwinding function is defined by

$$\operatorname{cond}_{\mathcal{U}}(A) = \lim_{\epsilon\to 0} \sup_{\|E\|\le\epsilon\|A\|} \frac{\|\mathcal{U}(A+E)-\mathcal{U}(A)\|}{\epsilon\|\mathcal{U}(A)\|}.$$

Note that because of the discontinuity of $\mathcal{U}(z)$ at points z whose imaginary part is an odd integer multiple of π, $\operatorname{cond}_{\mathcal{U}}(A) = \infty$ for any A with an eigenvalue of this form. The next result gives a lower bound for $\operatorname{cond}_{\mathcal{U}}(A)$. Recall that $\mathcal{D}$ is defined in (3.4).

Lemma 3.14. *For $A \in \mathbb{C}^{n\times n}$ with Jordan canonical form $A = ZJZ^{-1}$ and spectrum in $\mathcal{D}$,*

$$\operatorname{cond}_{\mathcal{U}}(A) \ge \frac{\pi}{\kappa(Z)} \max_{\lambda,\mu\in\Lambda(A)} \mathcal{U}[\lambda,\mu], \tag{3.9}$$

where

$$\mathcal{U}[\lambda,\mu] = \begin{cases} \dfrac{\mathcal{U}(\lambda)-\mathcal{U}(\mu)}{\lambda-\mu}, & \lambda \ne \mu, \\ \mathcal{U}'(\lambda) = 0, & \lambda = \mu \end{cases}$$

is a divided difference.

Proof. Apply Lemma 5.9 with $p = 2\pi i$. □

When $\operatorname{Im}\lambda$ and $\operatorname{Im}\mu$ lie close to, but on opposite sides of $(2k+1)\pi$, for some k, then $\mathcal{U}[\lambda,\mu] = (\lambda-\mu)^{-1}$, and hence $\mathcal{U}[\lambda,\mu]$ is necessarily large if $\operatorname{Re}\lambda \approx \operatorname{Re}\mu$; in this case the lower bound for $\operatorname{cond}_{\mathcal{U}}(A)$ is large unless $\kappa(Z)$ is large.

We now turn to estimation of the condition number. In the rest of this section we assume that the spectrum of A lies in $\mathcal{D}$. Denote the Fréchet derivative of a function f at A by $L_f(A,\cdot)$. By (1.12),

$$\operatorname{cond}_{\mathcal{U}}(A) = \frac{\|L_{\mathcal{U}}(A)\|\,\|A\|}{\|\mathcal{U}(A)\|},$$

where

$$\|L_{\mathcal{U}}(A)\| := \max_{Z\neq 0} \frac{\|L_{\mathcal{U}}(A,Z)\|}{\|Z\|}.$$

Moreover, since $L_{\mathcal{U}}$ is a linear operator,

$$\operatorname{vec}(L_{\mathcal{U}}(A,E)) = K_{\mathcal{U}}(A)\operatorname{vec}(E), \tag{3.10}$$

where $K_{\mathcal{U}}(A) \in \mathbb{C}^{n^2\times n^2}$ is the Kronecker form of the Fréchet derivative and vec is the operator that stacks the columns of a matrix on top of each other [80, Chap. 3]. Following [80, Alg. 3.22] we will approximate $\|L_{\mathcal{U}}(A)\|_1$ by $\|K_{\mathcal{U}}(A)\|_1$ and estimate the latter quantity using the block 1-norm estimation algorithm of Higham and Tisseur [87]. This algorithm requires the ability to evaluate matrix–vector products involving $K_{\mathcal{U}}(A)$ and $K_{\mathcal{U}}(A)^*$. A product $K_{\mathcal{U}}(A)y$ can be evaluated as the left-hand side of (3.10) with $\operatorname{vec}(E) = y$. This can be done using the formula

$$L_{\mathcal{U}}(A,E) = \frac{E - L_{\log}(\mathrm{e}^A, L_{\exp}(A,E))}{2\pi i}$$

obtained by applying the chain rule [80, Thm. 3.4] to (3.5), or by evaluating the unwinding function of a $2n\times 2n$ matrix [80, (3.13)] and then extracting the upper right $n\times n$ block, as explained in (1.13):

$$\mathcal{U}\left(\begin{bmatrix} A & E \\ 0 & A \end{bmatrix}\right) = \begin{bmatrix} \mathcal{U}(A) & L_{\mathcal{U}}(A,E) \\ 0 & \mathcal{U}(A) \end{bmatrix}. \tag{3.11}$$

How to evaluate a product $K_{\mathcal{U}}(A)^*y$ is not immediately obvious. We need to introduce the adjoint $L_f^\star$ of the Fréchet derivative L_f, which is defined by the condition

$$\langle L_f(A,G), H\rangle = \langle G, L_f^\star(A,H)\rangle \tag{3.12}$$

for all $G, H \in \mathbb{C}^{n\times n}$, where $\langle X, Y\rangle = \operatorname{trace}(Y^*X) = \operatorname{vec}(Y)^* \operatorname{vec}(X)$.

Lemma 3.15. *Let f be $2n-1$ times continuously differentiable on an open subset Ω of $\mathbb{R}$ or $\mathbb{C}$ such that for each connected component $\widetilde{\Omega}$ of Ω, $z \in \widetilde{\Omega}$ if and only if $-\bar{z} \in \widetilde{\Omega}$. Suppose that $\bar{f}(A)^* = -\bar{f}(A^*)$ for all $A \in \mathbb{C}^{n\times n}$ with spectrum in Ω, where $\bar{f}(z) := \overline{f(\bar{z})}$. Then*

$$L_f^\star(A, E) = L_{\bar{f}}(A^*, E) = -L_{\bar{f}}(A, E^*)^*. \tag{3.13}$$

Proof. The proof of the first equality is exactly the same as that of the corresponding equality in the analogous result [85, Lem. 6.2]. We reproduce it here, for convenience. Suppose, first, that f has the form $f(x) = \alpha x^k$, so that $L_f(A, G) = \alpha \sum_{i=1}^k A^{i-1} G A^{k-i}$. Then

$$\begin{aligned}
\langle L_f(A, G), H\rangle &= \operatorname{trace}\Big(H^*\alpha \sum_{i=1}^k A^{i-1} G A^{k-i}\Big) \\
&= \operatorname{trace}\Big(\alpha \sum_{i=1}^k A^{k-i} H^* A^{i-1} G\Big) \\
&= \Big\langle G, \overline{\alpha} \sum_{i=1}^k (A^*)^{i-1} H (A^*)^{k-i}\Big\rangle \\
&= \langle G, L_{\bar{f}}(A^*, H)\rangle,
\end{aligned}$$

and so $L_f^\star(A, H) = L_{\bar{f}}(A^*, H)$, which is the first equality in (3.13). By the linearity of L_f it follows that this equality holds for any polynomial. Finally, the equality holds for all f satisfying the conditions of the theorem because the Fréchet derivative of f is the same as that of the polynomial that interpolates f and its derivatives at the zeros of the characteristic polynomial of the block diagonal matrix $\operatorname{diag}(A, A)$ [80, Thm. 3.7], [89, Thm. 6.6.14].

To prove the second equality we consider $g = \bar{f}$. By the definition of the Fréchet derivative, $L_g(A, E) = g(A+E) - g(A) + o(\|E\|)$. Taking the conjugate transpose, $L_g(A, E)^* = g(A + E)^* - g(A)^* + o(\|E\|) = -g(A^* + E^*) + g(A^*) + o(\|E\|) = -L_g(A^*, E^*) + o(\|E\|)$. By the linearity of the Fréchet derivative we then have $L_g(A, E)^* = -L_g(A^*, E^*)$, which gives the second equality in (3.13). ∎

It is shown by Higham and Lin [85, Lem. 6.1] that $K_f(A)^* \operatorname{vec}(E) = \operatorname{vec}(L_f^\star(A, E))$ for any f. Combined with (3.13) this yields $K_f(A)^* \operatorname{vec}(E) = -\operatorname{vec}(L_{\bar{f}}(A, E^*)^*)$.

For the unwinding function we have $\overline{f} = -f$ by Lemma 3.3, so $L_{\overline{f}} = -L_f$ and $K_{\mathcal{U}}(A)^* \operatorname{vec}(E) = \operatorname{vec}(L_{\mathcal{U}}(A, E^*)^*)$, and hence products with $K_{\mathcal{U}}(A)^*$ can be computed in exactly the same way as products with $K_{\mathcal{U}}(A)$.

3.3.3 Identities involving the logarithm and powers

We now use the matrix unwinding function to derive mathematical identities involving the matrix logarithm and fractional matrix powers.

For any nonsingular $A \in \mathbb{C}^{n\times n}$ and any $\alpha \in \mathbb{C}$ we define the principal matrix power

$$A^\alpha = \mathrm{e}^{\alpha \log A}, \tag{3.14}$$

where we recall that log denotes the principal matrix logarithm defined at the start of Section 3.3. The following result is immediate from the definitions of $\mathcal{U}(A)$ and A^α.

Lemma 3.16. *For nonsingular $A \in \mathbb{C}^{n\times n}$ and $\alpha \in \mathbb{C}$,*

$$\log A^\alpha = \alpha \log A - 2\pi i \mathcal{U}(\alpha \log A).$$

To establish when $\log A^\alpha = \alpha \log A$, we need to determine when $\mathcal{U}(\alpha \log A) = 0$. The following result describes a particularly useful context in which the latter condition holds.

Corollary 3.17. *For nonsingular $A \in \mathbb{C}^{n\times n}$, $\log A^\alpha = \alpha \log A$ for $\alpha \in (-1, 1]$ and for $\alpha = -1$ if A has no eigenvalues on $\mathbb{R}^-$.*

Proof. It is immediate from (3.6) and Lemma 3.4 that $\mathcal{U}(\alpha \log A) = 0$ under the given conditions, so the result follows by Lemma 3.16. □

Note the special cases $\alpha = -1$ and $\alpha = 1/2$. We have $\log A^{-1} = -\log A$ if A has no eigenvalues on $\mathbb{R}^-$ and $\log A^{1/2} = \frac{1}{2}\log A$ for all A. The latter identity can be used to write $2^k \log A^{1/2^k} = \log A$, for any $k \in \mathbb{Z}$, which underlies the inverse scaling and squaring algorithm for computing the matrix logarithm [7], [8], [39].

We next describe the result of powering successively by α and $1/\alpha$.

Lemma 3.18. *For nonsingular $A \in \mathbb{C}^{n\times n}$ and $\alpha \in \mathbb{C}$,*

$$(A^\alpha)^{1/\alpha} = A\mathrm{e}^{-\frac{2}{\alpha}\pi i \mathcal{U}(\alpha \log A)}.$$

Proof. Using (3.14) and Lemma 3.16 we have

$$\begin{aligned}(A^\alpha)^{1/\alpha} &= \mathrm{e}^{\frac{1}{\alpha}\log A^\alpha} = \mathrm{e}^{\frac{1}{\alpha}(\alpha\log A - 2\pi i\mathcal{U}(\alpha\log A))} \\ &= A\mathrm{e}^{-\frac{2}{\alpha}\pi i\mathcal{U}(\alpha\log A)}. \qquad \square\end{aligned}$$

We note the special case of Lemma 3.18 with $\alpha = 2$, which will be needed in the next subsection:

$$(A^2)^{1/2} = A\mathrm{e}^{-\pi i\mathcal{U}(2\log A)}. \tag{3.15}$$

We proceed to study the relation between a logarithm of a matrix product and the logarithms of the matrices involved.

Lemma 3.19. *Let $A, B \in \mathbb{C}^{n\times n}$ be nonsingular matrices such that $AB = BA$. Then*

$$\log(AB) = \log A + \log B - 2\pi i\mathcal{U}(\log A + \log B).$$

Proof. Recall that $\mathrm{e}^{(A+B)t} = \mathrm{e}^{At}\mathrm{e}^{Bt}$ for all t if and only if A and B commute, [80, Thm. 10.2]. Since A and B commute, so do $\log A$ and $\log B$. We therefore have

$$\begin{aligned}\log(AB) &= \log(\mathrm{e}^{\log A}\mathrm{e}^{\log B}) = \log(\mathrm{e}^{\log A+\log B}) \\ &= \log A + \log B - 2\pi i\mathcal{U}(\log A + \log B),\end{aligned}$$

where we have used the definition (3.5) of the matrix unwinding function. $\square$

Recall from [80, Cor. 1.41] that if A and B commute, then for each eigenvalue μ_j of A there is an eigenvalue ν_j of B such that $\mu_j + \nu_j$ is an eigenvalue of $A + B$. We will call ν_j the eigenvalue corresponding to μ_j.

Corollary 3.20. *Let $A, B \in \mathbb{C}^{n\times n}$ be nonsingular matrices such that $AB = BA$. Then*

$$\log(AB) = \log A + \log B$$

if and only if $\arg\mu_j + \arg\nu_j \in (-\pi, \pi]$ for every eigenvalue μ_j of A and the corresponding eigenvalue ν_j of B.

Proof. Using Lemma 3.19, we write $\log(AB) = \log A + \log B$ if and only if $\mathcal{U}(\log A + \log B) = 0$. From Theorem 3.8 the latter equality holds if and only if the imaginary parts of the eigenvalues of $\log A + \log B$ lie in the interval $(-\pi, \pi]$, which is equivalent to $\arg \mu_j + \arg \nu_j \in (-\pi, \pi]$ for all j. □

Corollary 3.20 was proved by Higham [80, Thm. 11.3] directly from the definition of principal logarithm, and a variant of the result was obtained by Cheng, Higham, Kenney, and Laub [39, Lem. 2.1]; in both cases the additional assumption that A and B have no real negative eigenvalues was in force. The benefit of the matrix unwinding function is that it provides the correction term for the general case in Lemma 3.19.

The next result gives a relation between the power of a matrix product and the product of the powers.

Theorem 3.21. *Let $A, B \in \mathbb{C}^{n\times n}$ be nonsingular matrices such that $AB = BA$. Then, for any $\alpha \in \mathbb{C}$,*

$$(AB)^\alpha = A^\alpha B^\alpha \mathrm{e}^{-2\pi\alpha i \mathcal{U}(\log A + \log B)}.$$

Proof. Applying (3.14), Lemma 3.19, and (3.7b) we have

$$\begin{aligned}(AB)^\alpha &= \mathrm{e}^{\alpha \log(AB)} \\ &= \mathrm{e}^{\alpha(\log A + \log B - 2\pi i \mathcal{U}(\log A + \log B))} \\ &= A^\alpha B^\alpha \mathrm{e}^{-2\alpha\pi i \mathcal{U}(\log A + \log B)}. \qquad \square\end{aligned}$$

The next corollary characterizes when $(AB)^\alpha = A^\alpha B^\alpha$ holds in terms of the eigenvalues of A and B rather than $\log A$ and $\log B$.

Corollary 3.22. *Let $A, B \in \mathbb{C}^{n\times n}$ be nonsingular matrices such that $AB = BA$. Then $(AB)^\alpha = A^\alpha B^\alpha$ if and only if $\alpha \mathcal{U}(\log \mu_j + \log \nu_j) \in \mathbb{Z}$ for every eigenvalue μ_j of A and the corresponding eigenvalue ν_j of B.*

Proof. It follows from [80, Thm. 1.27] that all solutions of $\mathrm{e}^X = I$ are of the form $X = V \operatorname{diag}(2\pi i k_1, \dots, 2\pi i k_n) V^{-1}$, where V is an arbitrary nonsingular matrix and $k_j \in \mathbb{Z}$ for all j. Hence, given that $\mathcal{U}(\log A + \log B)$ is diagonalizable,

$\exp(-2\alpha\pi i\mathcal{U}(\log A + \log B)) = I$ if and only if the eigenvalues of $2\alpha\pi i\mathcal{U}(\log A + \log B)$ are of the form $2\pi i k_j$, $k_j \in \mathbb{Z}$, which yields the result. □

We note that $\alpha\mathcal{U}(\log\mu_j + \log\nu_j) \in \mathbb{Z}$ holds when either $\alpha \in \mathbb{Z}$ or $\mathcal{U}(\log\mu_j + \log\nu_j) = 0$, since $\mathcal{U}(\log\mu_j + \log\nu_j) \in \{-1, 0, 1\}$.

An important case in which the condition of Corollary 3.22 holds for all α is when the eigenvalues of A and B have arguments in $(-\pi/2, \pi/2]$, for then $\mathrm{Im}(\log\mu_j + \log\nu_j) \in (-\pi, \pi]$ and so $\mathcal{U}(\log\mu_j + \log\nu_j) = 0$. As a special case we recover the result of [80, Prob. 1.35], which states that $(AB)^{1/2} = A^{1/2}B^{1/2}$ when A and B commute and both have eigenvalues lying in the open right half-plane.

The following result clarifies the relation between $(\mathrm{e}^A)^\alpha$ and $\mathrm{e}^{\alpha A}$, for $\alpha \in \mathbb{C}$.

Theorem 3.23. *For $A \in \mathbb{C}^{n\times n}$ and $\alpha \in \mathbb{C}$, $(\mathrm{e}^A)^\alpha = \mathrm{e}^{\alpha A}\mathrm{e}^{-2\pi i\alpha\mathcal{U}(A)}$. Hence $(\mathrm{e}^A)^\alpha = \mathrm{e}^{\alpha A}$ if and only if $\alpha\mathcal{U}(\lambda) \in \mathbb{Z}$ for every eigenvalue λ of A.*

Proof. From the definitions (3.14) of matrix power and (3.5) of matrix unwinding function we have

$$(\mathrm{e}^A)^\alpha = \mathrm{e}^{\alpha\log\mathrm{e}^A} = \mathrm{e}^{\alpha(A-2\pi i\mathcal{U}(A))} = \mathrm{e}^{\alpha A}\mathrm{e}^{-2\pi i\alpha\mathcal{U}(A)}.$$

The last part follows as in the proof of Corollary 3.22. □

When α is an integer, the correction term in Theorem 3.23 is the identity matrix, and after rescaling $A \leftarrow \alpha^{-1}A$ and setting $\alpha = 2^s$ we obtain the basis of the scaling and squaring method for computing the matrix exponential: $(\mathrm{e}^{A/2^s})^{2^s} = A$.

Note that Theorem 3.23 shows that it is not the case that $\mathrm{e}^A = (\mathrm{e}^{A/\alpha})^\alpha$ holds for all $\alpha \in \mathbb{C}$, as is incorrectly stated in [80, p. 241]!

3.3.4 Relation with the matrix sign function

We now explore some interesting connections between the matrix unwinding function and the matrix sign function. The scalar variants of some of these are given in [41, Table A.1].

Recall that the matrix sign function is defined only for $A \in \mathbb{C}^{n\times n}$ with no purely imaginary eigenvalues and is given by $\mathrm{sign}(A) = A(A^2)^{-1/2}$, as well as by various other equivalent formulas [80, Chap. 5], [99].

Taking the inverse of equation (3.15) we can write

$$\operatorname{sign}(A) = A(A^2)^{-1/2} = AA^{-1}\mathrm{e}^{\pi i\mathcal{U}(2\log A)} = \mathrm{e}^{\pi i\mathcal{U}(2\log A)}. \tag{3.16}$$

If the eigenvalues of A lie in the open right half-plane then the eigenvalues of $\log A$ have imaginary parts in the interval $(-\pi/2, \pi/2)$, hence $\mathcal{U}(2\log A) = 0$ and $\operatorname{sign}(A) = I$. Conversely, if the eigenvalues of A lie in the open left half-plane then the imaginary part of every eigenvalue λ of $\log A$ lies in $(-\pi, -\pi/2)$ or $(\pi/2, \pi]$, and hence $\mathcal{U}(2\lambda) = -1$ or 1, respectively, yielding $\operatorname{sign}(A) = -I$.

We note that the right-hand side of our formula (3.16) is defined for any nonsingular A. The formula gives a meaning to the sign function on the imaginary axis: for $y > 0$, $\operatorname{sign}(iy) = 1$ and $\operatorname{sign}(-iy) = -1$. Indeed, this conforms with the counter-clockwise continuity principle introduced by Kahan [95]. We will call $\operatorname{sign}(A) := \mathrm{e}^{\pi i\mathcal{U}(2\log A)}$ the extended matrix sign function and we note that it is different to other extensions in the literature of the sign function to arbitrary nonsingular matrices [99, Sec. I.D].

This result can be generalized for the matrix sector function [80, Sec. 2.14.3], [136], which for a given integer p and $A \in \mathbb{C}^{n\times n}$ with no eigenvalues with argument $(2k+1)\pi/p$, $k = 0 : p-1$, is defined as $\operatorname{sect}_p(A) = A(A^p)^{-1/p}$. From Lemma 3.18 we have

$$\operatorname{sect}_p(A) = \mathrm{e}^{\frac{2}{p}\pi i\mathcal{U}(p\log A)}.$$

Analogously to the relation $\operatorname{sign}(A) = A(A^2)^{-1/2}$, we have the following result involving the extended matrix sign function. Recall that $\mathcal{D}$ is defined in (3.4).

Lemma 3.24. *For a nonsingular $A \in \mathbb{C}^{n\times n}$ with spectrum in $\mathcal{D}$,*

$$\mathcal{U}(A) = \operatorname{sign}(A)\mathcal{U}\big((A^2)^{1/2}\big). \tag{3.17}$$

Proof. It suffices to prove the result for diagonalizable A, by [80, Thm. 1.20] (or simply because $\mathcal{U}(\cdot)$ and $\operatorname{sign}(\cdot)$ are diagonalizable), so the result reduces to the scalar case, $\mathcal{U}(z) = \operatorname{sign}(z)\mathcal{U}((z^2)^{1/2})$. If we now suppose z lies in the open left half-plane, or on $i\mathbb{R}^-$, $\operatorname{sign}(z) = -1$ and $(z^2)^{1/2} = -z$. Since for any $z \in \mathcal{D}$, $\mathcal{U}(-z) = -\mathcal{U}(z)$ by Lemma 3.3, the desired result follows. A similar argument applies for z in the open right half-plane or on $i\mathbb{R}^+$. ∎

We can use (3.17) to derive a formula for the matrix unwinding function of a particular block matrix.

Theorem 3.25. *For nonsingular $A, B \in \mathbb{C}^{n\times n}$ such that $(AB)^{1/2}$ has spectrum in $\mathcal{D}$,*

$$\mathcal{U}\left(\begin{bmatrix} 0 & A \\ B & 0 \end{bmatrix}\right) = \begin{bmatrix} 0 & A(BA)^{-1/2}\mathcal{U}((BA)^{1/2}) \\ B(AB)^{-1/2}\mathcal{U}((AB)^{1/2}) & 0 \end{bmatrix}.$$

Proof. The result is obtained by applying (3.17) to the matrix $C = \left[\begin{smallmatrix} 0 & A \\ B & 0 \end{smallmatrix}\right]$ and then using

$$\mathcal{U}\big((C^2)^{1/2}\big) = \mathcal{U}\big(\operatorname{diag}((AB)^{1/2}, (BA)^{1/2})\big) = \operatorname{diag}\big(\mathcal{U}((AB)^{1/2}), \mathcal{U}((BA)^{1/2})\big)$$

and the following result of Higham, Mackey, Mackey, and Tisseur [86, Lem. 4.3], [80, Thm. 5.2] (which holds even when AB has eigenvalues on $\mathbb{R}^-$, given that we are using the extended matrix sign function):

$$\operatorname{sign}\left(\begin{bmatrix} 0 & A \\ B & 0 \end{bmatrix}\right) = \begin{bmatrix} 0 & A(BA)^{-1/2} \\ B(AB)^{-1/2} & 0 \end{bmatrix}. \qquad \square$$

3.4 Algorithm

The matrix unwinding function can be computed directly via the definition (3.5), but this requires a matrix exponential and a matrix logarithm. Instead we will compute a Schur decomposition $A = QTQ^* \in \mathbb{C}^{n\times n}$, where Q is unitary and T is upper triangular, after which $\mathcal{U}(A) = Q\mathcal{U}(T)Q^*$. The problem is reduced to computing $\mathcal{U}(T)$, which we will do directly rather than via the exponential and logarithm.

The Parlett recurrence for computing a function of a triangular matrix T is not appropriate because it breaks down when T has repeated diagonal elements. The Schur–Parlett method [51], which is implemented in the MATLAB function `funm`, reorders and blocks T so that the eigenvalues within a diagonal block are close while distinct diagonal blocks have well separated eigenvalues. We do not have a

special way of evaluating the unwinding function of a triangular matrix with close eigenvalues, so we cannot use this general method.

Instead we adapt the Schur–Parlett method by putting eigenvalues having the same unwinding number in the same block. We use the algorithm of Bai and Demmel [18] (implemented in the MATLAB function `ordschur`) to compute a unitary V such that $\widetilde{T} = V^*TV = (\widetilde{T}_{ij})$ is upper triangular with all the diagonal elements of $\widetilde{T}_{ii}$ having imaginary parts in the same interval $((2k_i-1)\pi, (2k_i+1)\pi]$, for some $k_i \in \mathbb{Z}$. The diagonal blocks of $F = \mathcal{U}(\widetilde{T})$ are therefore given by $F_{ii} = u_i I$ for all i, by (3.6). The off-diagonal blocks are obtained from the block Parlett recurrence, which is obtained by equating blocks in $F\widetilde{T} = \widetilde{T}F$:

$$\widetilde{T}_{ii}F_{ij} - F_{ij}\widetilde{T}_{jj} = (u_i - u_j)\widetilde{T}_{ij} + \sum_{k=i+1}^{j-1} (F_{ik}\widetilde{T}_{kj} - \widetilde{T}_{ik}F_{kj}), \qquad i < j. \tag{3.18}$$

These Sylvester equations are nonsingular, since $\widetilde{T}_{ii}$ and $\widetilde{T}_{jj}$ have no eigenvalue in common, and they can be solved a block column or a block superdiagonal at a time.

For notational simplicity, we express our algorithm at the scalar level, but it is mathematically equivalent to carrying out the blocking described above and using the block Parlett recurrence; blocking is preferred in practice as it allows the use of higher level BLAS.

Algorithm 3.26. *Given $A \in \mathbb{C}^{n\times n}$ this algorithm computes the unwinding function $U = \mathcal{U}(A)$ using the Schur–Parlett method with a particular reordering and blocking.*

1 Compute the Schur decomposition $A = QTQ^*$ (Q unitary, T upper triangular).
2 If $\operatorname{Im} t_{ii} \in (-\pi, \pi]$ for all i, $U = 0$, quit, end
3 Assign t_{ii} to set $S_{\mathcal{U}(t_{ii})}$, $i = 1\colon n$, and use a unitary similarity transformation to reorder T so that all elements belonging to each set $S_{\mathcal{U}(t_{ii})}$ are contiguous, and update Q.
4 $f_{ii} = \mathcal{U}(t_{ii})$, $i = 1\colon n$
5 for $j = 2\colon n$
6 for $i = j-1\colon -1\colon 1$
7 if $f_{ii} = f_{jj}$

```
8            f_ij = 0
9          else
10           f_ij = ( t_ij(f_ii − f_jj) + Σ_{k=i+1}^{j−1} (f_ik t_kj − t_ik f_kj) ) / (t_ii − t_jj)
11         end
12       end
13     end
14     U = QFQ*
```

Cost: $25n^3$ flops for the Schur decomposition plus the cost of the reordering, $n^3/3$ flops for F, and $3n^3$ flops to form U.

The cost of the reordering is at most $10n^3 - 20n^2$ flops. We look for the minimum number of swaps of adjacent elements in an index array to obtain a confluent permutation. This will be used to re-order the eigenvalues appearing on the diagonal of the Schur form. Suppose we want to rearrange an array of total length n containing k different groups/blocks of size greater than 1. Each one of them has size s_i, $i = 1 : k$. Note that for our purposes we can ignore the groups of size 1.

For the first group we choose to rearrange, we need at most $n - s_1$ swaps for each element, i.e., $s_1(n - s_1)/2$ in total.

We can apply the same logic to the second group we chose to rearrange, with the difference that our starting array is of length $n - s_1$ since we have already arranged the first s_1 positions in the permutation. The total number of swaps for the second group is then $(s_2(n - s_1 - s_2))/2$.

The total number of swaps we require then is at most

$$\sum_{i=1}^{k} \frac{s_i(n - s_1 - s_2 - \cdots - s_i)}{2}. \tag{3.19}$$

This bound is attainable and at its maximum for $k = n/2$, i.e., each group has size 2. Then we need $n^2/4 - n/2$ swaps. The total number of operations required to rearrange the Schur decomposition is $40n \times$ total number of swaps [18], so at most $10n^3 - 20n^2$. This figure will usually be much smaller, because this bound assumes a worst case distribution of diagonal entries of the Schur factor and block sizes.

Note that an alternative in Algorithm 3.26 is to reorder T in such way that $\operatorname{Im} t_{11} \leq \cdots \leq \operatorname{Im} t_{nn}$. However, this requires more swaps in general, so it is more

expensive and introduces more rounding errors.

The condition number of the Sylvester equations (3.18) is proportional to the reciprocal of the separation of $\widetilde{T}_{ii}$ and $\widetilde{T}_{jj}$ [78, Sec. 16.3], which is bounded below by the reciprocal of $\min\{\,|\lambda-\mu| : \lambda \in \Lambda(\widetilde{T}_{ii}),\ \mu \in \Lambda(\widetilde{T}_{jj})\,\}$. Hence (3.18) can be ill conditioned, as there is no lower bound on the absolute value of the differences between eigenvalues of $\widetilde{T}_{ii}$ and $\widetilde{T}_{jj}$. A small eigenvalue difference occurs precisely when two consecutive blocks have eigenvalues λ_i and λ_j such that $\mathrm{Re}(\lambda_i - \lambda_j) < \epsilon$, and $\operatorname{Im}\lambda_i = (2k+1)\pi - \delta_1$, $\operatorname{Im}\lambda_j = (2k+1)\pi + \delta_2$, for some $k \in \mathbb{Z}$ and some small $\epsilon \in \mathbb{R}$, $\delta_1, \delta_2 \in \mathbb{R}^+$, which is equivalent to $\mathcal{U}[\lambda_i, \lambda_j]$ being large. The latter condition implies that $\mathrm{cond}_{\mathcal{U}}(A)$ is large if A is close to normal, by (3.9). Hence one particular cause of ill conditioning in the Sylvester equations is linked to ill conditioning of $\mathcal{U}(A)$.

When A is real, $\mathcal{U}(A)$ is pure imaginary if A has no eigenvalues with imaginary parts an odd integer multiple of π, by Corollary 3.10. In this case we would like our algorithm to guarantee a pure imaginary result. We can compute a real Schur decomposition $A = QTQ^T$, where T is real and upper quasitriangular. The matrix unwinding function of any 2×2 diagonal blocks can be computed using (3.8). However, the block Parlett recurrence may break down due to two different diagonal blocks having the same eigenvalues, so this approach is not reliable and we will not consider it further. This inability to split complex conjugate eigenvalues affects the standard Schur–Parlett algorithm in the same way, preventing the derivation of a version tailored to real matrices [51], [80, Sec. 9.4].

3.5 Numerical experiments

We investigate experimentally two algorithms for computing $\mathcal{U}(A)$:

- Algorithm 3.26.
- log-exp: evaluation of (3.5) using the scaling and squaring algorithm of Al-Mohy and Higham [5] for the exponential and the inverse scaling and squaring algorithm of Al-Mohy, Higham, and Relton [8] for the logarithm; the matrix

A is reduced to Schur form (the real Schur form if A is real) before applying these algorithms.

MATLAB implementation of Algorithm 3.26 is available at `https://github.com/aprahamian/matrix-unwinding`. All tests were done in MATLAB R2015a, for which the unit roundoff $u \approx 1.1 \times 10^{-16}$.

While log-exp is useful as a means for comparison, we note that it has two flaws that make it unsuitable as a general way to compute $\mathcal{U}(A)$. First, it is prone to overflow, since e^A can overflow when $\mathcal{U}(A)$ does not, as is clear from the scalar case; indeed, $\mathcal{U}(A)$ is very unlikely to overflow, in view of Lemma 3.13. Second, e^A can be singular, making the log computation fail. For example, for

$$A = \begin{bmatrix} 1 & 1 \\ 0 & -10^3 \end{bmatrix}, \quad fl(\mathrm{e}^A) = \begin{bmatrix} \mathrm{e} & fl(\mathrm{e}/(10^3+1)) \\ 0 & 0 \end{bmatrix},$$

since $fl(\mathrm{e}^{-10^3}) = 0$. Evaluation with log-exp fails since $fl(\mathrm{e}^A)$ is singular, yet $\mathcal{U}(A) = 0$, as is correctly computed by Algorithm 3.26.

To test the performance, we first constructed a set of 10×10 random matrices, Set 1, with eigenvalues λ_i that are odd integer (between 1 and 11) multiples of πi perturbed by 10^{-6} times complex numbers with $N(0,1)$ distributed real and imaginary parts. The matrices, 40 in total, are the upper triangular Schur factors of $A = XDX^{-1}$, where $D = \mathrm{diag}(\lambda_i)$ and X is random with 2-norm condition number 2, 10, 100, or 1000, obtained in MATLAB as `gallery('randsvd',...)`. Figure 3.1 shows the relative error $\|\mathcal{U}(A) - \widehat{U}\|_F / \|\mathcal{U}(A)\|_F$, where $\widehat{U}$ is the computed unwinding function and $\mathcal{U}(A)$ is the correctly rounded one; $\mathcal{U}(A)$ is obtained by evaluating (3.6) at 100 digit precision using the Symbolic Math Toolbox then rounding to double precision. The matrices are arranged according to decreasing value of $\kappa_2(X)$, with 10 matrices for each value. An estimate of $\mathrm{cond}_{\mathcal{U}}(A)u$ is shown in the figure, where $\mathrm{cond}_{\mathcal{U}}(A)$ is estimated as indicated in Section 3.3.2, using Algorithm 3.26 with (3.11) to obtain the Fréchet derivatives. We see that both algorithms produce errors smaller than $\mathrm{cond}_{\mathcal{U}}(A)$ in every case, showing that they are performing in a forward stable fashion. Algorithm 3.26 produces errors substantially smaller than log-exp in many cases.

Our second test uses a set of 24 matrices, Set 2, drawn from `gallery`, the Matrix Computation Toolbox [76] and test problems provided with EigTool [147]. These matrices have been scaled to make them have nonzero unwinding functions or otherwise make them useful for test purposes. Figure 3.2 shows the relative errors for both algorithms, with the matrices arranged by decreasing estimated condition number $\mathrm{cond}_{\mathcal{U}}(A)$. For matrix 24, the relative error for Algorithm 3.26 is 0 and so is not plotted in the figure. Algorithm 3.26 performs in a forward stable way in every case, but log-exp is unstable on five matrices.

The conclusion from these tests is that Algorithm 3.26 is usually more accurate than log-exp and performs in a forward stable manner in the test sets. In fact, in further experiments we have not been able to generate an example where the computed result from Algorithm 3.26 has a relative error substantially larger than $\mathrm{cond}_{\mathcal{U}}(A)u$.

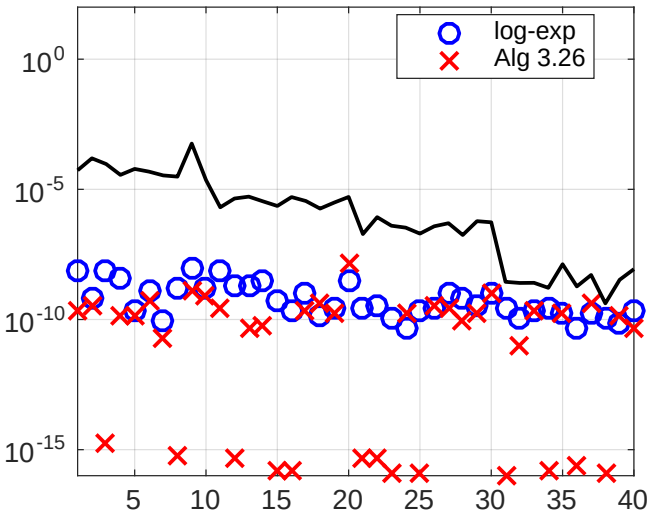


Figure 3.1: Relative errors for Set 1. The solid line is an estimate of $\mathrm{cond}_{\mathcal{U}}(A)u$.

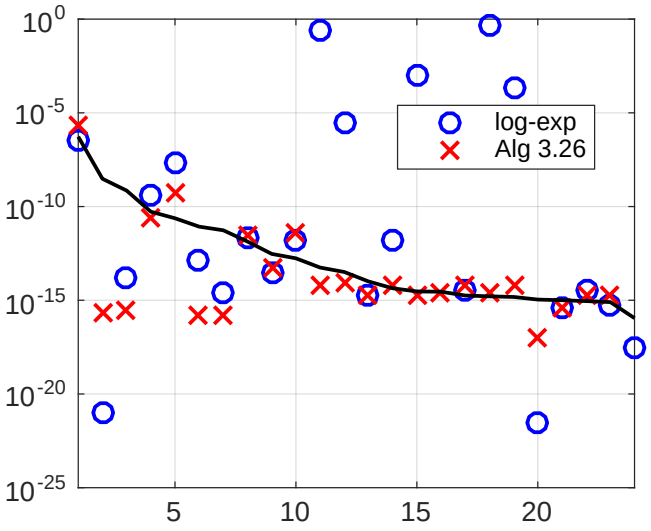


Figure 3.2: Relative errors for Set 2. The solid line is an estimate of $\mathrm{cond}_{\mathcal{U}}(A)u$.

CHAPTER 4

Matrix Inverse Trigonometric and Inverse Hyperbolic Functions

4.1 Introduction

Trigonometric functions of matrices play an important role in the solution of second order differential equations; see, for example, [9], [135], and the references therein. The inverses of such functions, and of their hyperbolic counterparts, also have practical applications, but have been less well studied. An early appearance of the matrix inverse cosine was in a 1954 paper on the energy equation of a free-electron model [134]. The matrix inverse hyperbolic sine arises in a model of the motion of rigid bodies, expressed via Moser–Veselov matrix equations [35]. The matrix inverse sine and inverse cosine were used by Al-Mohy, Higham, and Relton [9] to define the backward error in approximating the matrix sine and cosine. In Chapter 5 of this thesis we use matrix inverse trigonometric and inverse hyperbolic functions to study argument reduction for use in computing the matrix sine, cosine, and hyperbolic sine and cosine.

This work has two aims. The first is to develop the theory of matrix inverse trigonometric functions and inverse hyperbolic functions. Most importantly, we define the principal values acos, asin, acosh, and asinh, prove their existence and uniqueness, and develop various useful identities involving them. In particular, we determine the precise relationship between $\operatorname{acos}(\cos A)$ and A, and similarly for the

other functions. The second aim is to develop algorithms and software for computing acos, asin, acosh, and asinh of a matrix, for which we employ variable-degree Padé approximation together with appropriate initial transformations. Very little has been published on computation of these matrix functions and the only publicly available software we are aware of that is designed specifically for computing these functions is in GNU Octave [70], [83].

Corless et al. [43] note that in the elementary function literature definitions and identities are often imprecise or inconsistent and need careful interpretation. While it is arguably reasonable to ask a reader to determine the correct sign or choice of branch in a scalar formula, in a formula in $n \times n$ matrices, whose information content is at least n scalar identities involving the (unknown) eigenvalues, imprecision is a recipe for confusion and controversy. We are therefore scrupulous in this work to give precise definitions and derive formulas that are valid under clearly stated conditions.

The rest of this chapter is organized as follows. In the next section we give necessary and sufficient conditions for the existence of matrix inverse cosine and sine functions and their hyperbolic counterparts and characterize all their possible values. Then we define the branch points, branch cuts, and principal values, and prove the uniqueness of the principal values. In Section 4.3 we develop a variety of identities involving the matrix inverse functions, some of which are new even in the scalar case. In Section 4.4 we discuss the conditioning of the inverse functions. An algorithm for computing acos that combines a Schur decomposition and Padé approximation with a square root recurrence is given in Section 4.5; the algorithm yields algorithms for asin, acosh, and sinh. In Section 4.6 we give numerical experiments that compare the new algorithms with the use of formulas based on the matrix logarithm and square root.

4.2 The inverse functions

We first define and characterize the matrix inverse trigonometric and inverse hyperbolic functions and then treat their principal values. We will repeatedly use the

principal matrix logarithm, principal matrix square root, and matrix sign function, with extensions on their respective branch cuts. These are defined as follows.

A logarithm of a nonsingular $A \in \mathbb{C}^{n\times n}$, written $X = \operatorname{Log} A$, is a solution of $\mathrm{e}^X = A$. The principal logarithm of a nonsingular $A \in \mathbb{C}^{n\times n}$, denoted $\log A$, is the logarithm all of whose eigenvalues have imaginary parts in the interval $(-\pi, \pi]$. We take the branch cut to be the negative real axis $\mathbb{R}^-$. Note that the principal matrix logarithm is usually not defined for matrices with eigenvalues on the negative real axis [80, Chap. 11], but for the purposes of this work it is convenient to allow the extension of the logarithm on the branch cut and to adopt the convention that $\log(-y) = \log y + \pi i$ for $y > 0$.

A square root of $A \in \mathbb{C}^{n\times n}$, written $X = \sqrt{A}$, is a solution of $X^2 = A$. We take the branch cut to be $\mathbb{R}^-$ and define the the principal square root to be the one all of whose eigenvalues have nonnegative real parts and such that $(-y)^{1/2} = y^{1/2} i$ for $y > 0$. Consistent with the principal logarithm defined above, we can write the principal square root of any nonsingular complex matrix A as $A^{1/2} = \mathrm{e}^{\frac{1}{2}\log A}$.

We also need the matrix sign function $\operatorname{sign} A$ [80, Chap. 5], which maps each eigenvalue of A to the sign (± 1) of its real part. To include the case where A has an eigenvalue on the imaginary axis we define $\operatorname{sign}(0) = 1$ and $\operatorname{sign}(yi) = \operatorname{sign}(y)$ for nonzero $y \in \mathbb{R}$.

These particular choices for the values of the sign function and the logarithm and square root on their branch cuts, which we previously used in Chapter 3, adhere to the counter-clockwise continuity principle introduced by Kahan [95, Sec. 5].

We recall that for a multivalued function f a nonprimary matrix function $f(A)$ is obtained if, in the definition of matrix function via the Jordan canonical form, some eigenvalue λ appears in more than one Jordan block and is assigned different values $f(\lambda)$ on at least two of the blocks [80, sec. 1.2]. This means that $f(A)$ is not expressible as a polynomial in A.

4.2.1 Existence and characterization

An inverse cosine of $A \in \mathbb{C}^{n\times n}$ is any solution of the equation $\cos X = A$. Inverse sines, and inverse hyperbolic sines and cosines, are defined in an analogous way.

Using Euler's formula, for $X \in \mathbb{C}^{n\times n}$,

$$\mathrm{e}^{iX} = \cos X + i \sin X, \tag{4.1}$$

we can write the matrix cosine and sine functions in their exponential forms

$$\cos X = \frac{\mathrm{e}^{iX} + \mathrm{e}^{-iX}}{2}, \qquad \sin X = \frac{\mathrm{e}^{iX} - \mathrm{e}^{-iX}}{2i}. \tag{4.2}$$

To establish whether solutions to the equation $A = \cos X$ exist we use the exponential form to write $A = (\mathrm{e}^{iX} + \mathrm{e}^{-iX})/2$. This equation implies that A commutes with the nonsingular matrix e^{iX}, and after multiplying through by e^{iX} the equation can be written as

$$(\mathrm{e}^{iX} - A)^2 = A^2 - I.$$

Taking square roots gives

$$\mathrm{e}^{iX} = A + \sqrt{A^2 - I}, \tag{4.3}$$

provided that $A^2 - I$ has a square root. The matrix $A + \sqrt{A^2 - I}$ is always nonsingular and so we can take logarithms to obtain $X = -i\,\mathrm{Log}(A + \sqrt{A^2 - I})$. Any inverse matrix cosine must have this form. In order to reverse the steps of this argument we need to show that e^{iX} commutes with A, which can be guaranteed when $\sqrt{A^2 - I}$ can be expressed as a polynomial in A, which in turn is true if the square root is a primary matrix function [80, Sec. 1.2], that is, if each occurrence of any repeated eigenvalue is mapped to the same square root. If a nonprimary square root is taken it may or may not yield an inverse cosine.

Similar analysis can be done for the matrix inverse sine. Results for the inverse hyperbolic functions can be obtained using the relations

$$\cosh X = \cos iX, \qquad \sinh X = -i \sin iX, \tag{4.4}$$

which hold for any $X \in \mathbb{C}^{n\times n}$ and can be taken as the definitions of cosh and sinh.

Theorem 4.1. *Let $A \in \mathbb{C}^{n\times n}$.*

1. *The equation $\cos X = A$ has a solution if and only if $A^2 - I$ has a square root. Every solution has the form $X = -i\,\mathrm{Log}(A + \sqrt{A^2 - I})$ for some square root and logarithm.*

2. *The equation* $\sin X = A$ *has a solution if and only if* $I - A^2$ *has a square root. Every solution has the form* $X = -i\operatorname{Log}(iA + \sqrt{I - A^2})$ *for some square root and logarithm.*

3. *The equation* $\cosh X = A$ *has a solution if and only if* $A^2 - I$ *has a square root. Every solution has the form* $X = \operatorname{Log}(A + \sqrt{A^2 - I})$ *for some square root and logarithm.*

4. *The equation* $\sinh X = A$ *has a solution if and only if* $A^2 + I$ *has a square root. Every solution has the form* $X = \operatorname{Log}(A + \sqrt{A^2 + I})$ *for some square root and logarithm.*

In 1–4 the given expression for X *is guaranteed to be a solution when the square root is a primary square root.*

We emphasize that the square roots and logarithms in the statement of the theorem need not be primary. Note also that the existence of a square root of a matrix is in question only when the matrix is singular. Necessary and sufficient conditions for the existence of a square root of a singular matrix are given in [48], [80, Thm. 1.22].

To illustrate the use of these results, we consider the existence of an inverse sine of the 2×2 matrix $A = \left[\begin{smallmatrix} 1 & 1996 \\ 0 & 1 \end{smallmatrix}\right]$ [80, Prob. 1.50] (Putnam Problem 1996–B4). It is easy to see that $I - A^2 = \left[\begin{smallmatrix} 0 & -3992 \\ 0 & 0 \end{smallmatrix}\right]$ does not have a square root and hence the equation $A = \sin X$ has no solutions. Two very similar 2×2 examples are given by Pólya and Szegö [130, p. 35, Prob. 210].

4.2.2 Branch points, branch cuts, and principal values

The inverse cosine and inverse sine functions, and their hyperbolic counterparts, are multivalued. We now specify their branch points and branch cuts. The branch points of acos and asin are at 1 and -1 and, in accordance with popular convention [119, secs. 4.23(ii), 4.23(vii)], we consider their branch cuts to be on the two segments of the real line

$$\Omega = \Omega_1 \cup \Omega_2 = (-\infty, -1] \cup [1, \infty). \tag{4.5}$$

The branch points of asinh are at i and $-i$ and the branch cuts are the segments of the imaginary line $i\,\Omega$; the branch points of acosh are at 1 and -1 and the branch cut is the segment of the real line [119, Sec. 4.37(ii)]

$$\widetilde{\Omega} = \Omega_1 \cup \Omega_3 \equiv (-\infty, -1] \cup [-1, 1] = (-\infty, 1]. \tag{4.6}$$

In the following definition we specify the principal values of the functions, in a way consistent with the scalar case [119, Secs 4.23(ii), 4.37(ii)] and with the counter-clockwise continuity principle [95]. We refer to Figure 4.1 for plots of the domains and ranges of the principal branches of the scalar functions (the plots extend ones in [128]). The figure also shows where the branch cuts are and what values the principal functions take on these branch cuts. The hashes placed on the sides of the branch cuts indicate that if a sequence $\{z_k\}$ tends to a point w on the branch cut from the side with the hashes then $\lim_{k\to\infty} f(z_k) \neq f(w)$.

Definition 4.2 (Principal values)**.** *Let $A \in \mathbb{C}^{n\times n}$.*

1. *The principal inverse cosine of A, denoted* $\operatorname{acos} A$*, is the inverse cosine for which every eigenvalue*

   (a) *has real part lying in $(0, \pi)$, or*

   (b) *has zero real part and nonnegative imaginary part (corresponding to A having an eigenvalue in Ω_2), or*

   (c) *has real part π and nonpositive imaginary part (corresponding to A having an eigenvalue in Ω_1).*

2. *The principal inverse sine of A, denoted* $\operatorname{asin} A$*, is the inverse sine for which every eigenvalue*

   (a) *has real part lying in $(-\pi/2, \pi/2)$, or*

   (b) *has real part $-\pi/2$ and nonnegative imaginary part (corresponding to A having an eigenvalue in Ω_1), or*

   (c) *has real part $\pi/2$ and nonpositive imaginary part (corresponding to A having an eigenvalue in Ω_2).*

3. *The principal inverse hyperbolic cosine of A, denoted* $\operatorname{acosh} A$*, is the inverse hyperbolic cosine for which every eigenvalue*

   (a) *has imaginary part lying in $(-\pi, \pi)$ and positive real part, or*

   (b) *has imaginary part in $[0, \pi)$ and zero real part (corresponding to A having an eigenvalue in Ω_3), or*

   (c) *has imaginary part π and nonnegative real part (corresponding to A having an eigenvalue in Ω_1).*

4. *The principal inverse hyperbolic sine of A, denoted* $\operatorname{asinh} A$*, is the inverse hyperbolic sine for which every eigenvalue*

   (a) *has imaginary part lying in $(-\pi/2, \pi/2)$, or*

   (b) *has imaginary part $-\pi/2$ and nonpositive real part (corresponding to A having an eigenvalue in $i\Omega_1$), or*

   (c) *has imaginary part $\pi/2$ and nonnegative real part (corresponding to A having an eigenvalue in $i\Omega_2$).*

Note that if A has no eigenvalues on the respective branch cuts then part (i) of each of (a)–(d) in Definition 4.2 is in operation. Moreover, under this condition the principal inverse function exists, is unique, and is a primary matrix function of A, as shown by the next result.

Theorem 4.3. *Let $A \in \mathbb{C}^{n\times n}$.*

1. *If A has no eigenvalues equal to 1 or -1 then there is a unique principal inverse cosine* $\operatorname{acos} A$*, a unique inverse sine* $\operatorname{asin} A$*, and a unique inverse hyperbolic cosine* $\operatorname{acosh} A$*, and all are primary matrix functions of A.*

2. *If A has no eigenvalues equal to i or $-i$ then there is a unique principle inverse hyperbolic sine* $\operatorname{asinh} A$ *and it is a primary matrix function of A.*

Proof. Consider asin, which by Definition 4.2 must have eigenvalues with real parts in the interval $(-\pi/2, \pi/2)$, or real parts $-\pi/2$ and nonnegative imaginary parts, or real parts $\pi/2$ and nonpositive imaginary parts. Note first that inverse sines exist

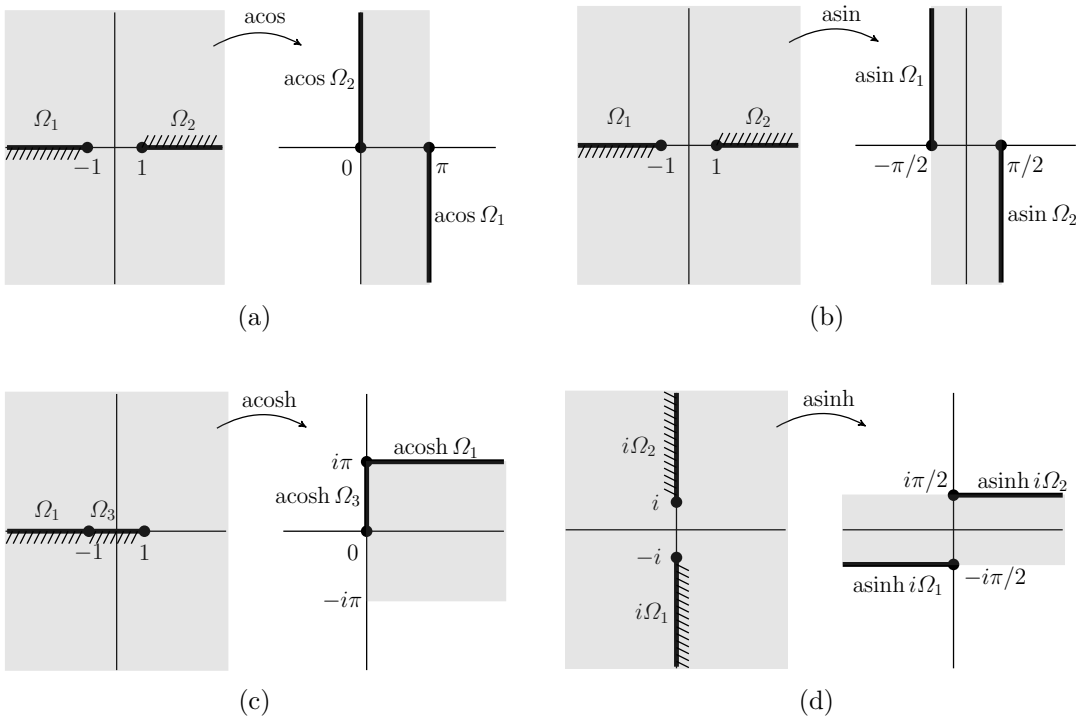


Figure 4.1: Domains and ranges of the principal branches of the complex functions acos (a), asin (b), acosh (c), and asinh (d).

by Theorem 4.1 2, since $I - A^2$ is nonsingular under the assumptions on A. Observe that a nonprimary inverse sine of A (if one exists) must have two eigenvalues μ_i and μ_j with $\mu_j = (-1)^k \mu_i + k\pi$ for some nonzero integer k. Since A has no eigenvalues equal to 1 or -1 such an inverse sine cannot satisfy Definition 4.2 (b). Therefore no nonprimary inverse sine can be a principal inverse sine. Finally, there exists a way, and hence precisely one way, to map the eigenvalues with the inverse sine in such a way that all eigenvalues have the characterization given in Definition 4.2, and that is with asin.

The proofs for acos, acosh, and asinh are completely analogous. □

4.3 Identities

Now we derive identities involving the principal matrix inverse trigonometric and inverse hyperbolic functions. Some of the results generalize existing scalar results, but others are new even in the scalar case.

The first result provides explicit formulas for the principal inverse functions in terms of the principal logarithm and the principal square root. Note that the exclusion of the branch points as eigenvalues of A in the next result, and later results, is necessary in order to ensure the existence of the inverse functions.

Theorem 4.4. *For $A \in \mathbb{C}^{n\times n}$, assuming that A has no eigenvalues at the branch points of the respective inverse functions,*

$$\begin{aligned}\operatorname{acos} A &= -i\log(A + i(I - A^2)^{1/2}) \qquad (4.7)\\ &= -2i\log\left(\left(\frac{I+A}{2}\right)^{1/2} + i\left(\frac{I-A}{2}\right)^{1/2}\right),\\ \operatorname{asin} A &= -i\log(iA + (I - A^2)^{1/2}), \qquad (4.8)\\ \operatorname{acosh} A &= \log(A + (A-I)^{1/2}(A+I)^{1/2}) \qquad (4.9)\\ &= 2\log\left(\left(\frac{A+I}{2}\right)^{1/2} + \left(\frac{A-I}{2}\right)^{1/2}\right),\\ \operatorname{asinh} A &= \log(A + (A^2 + I)^{1/2}). \qquad (4.10)\end{aligned}$$

Proof. These identities are known to hold for complex scalars [42], [95], [119, Secs 4.23(iv), 4.37(iv)]. If we were to exclude the eigenvalues of A from the branch cuts, which are the only points of non-differentiability of the inverse functions, it would follow from [80, Thm. 1.20], [89, Thm. 6.2.27 (2)] that the identities hold in the matrix case. In fact, they hold even if A has eigenvalues on the branch cuts. We show only that the first equality in (4.7) holds, as the proofs of the remaining identities are analogous. From the given conditions, the matrix $-i\log(A + i(I - A^2)^{1/2})$ exists and by Theorem 4.1 (a) it is an inverse cosine of A. It is readily verified that the eigenvalues of $-i\log(A + i(I - A^2)^{1/2})$ satisfy the conditions of Definition 4.2 (a) and therefore $-i\log(A + i(I - A^2)^{1/2})$ must be the principal inverse cosine of A. □

The next result completely describes the relation between the acos and asin functions. It is the matrix counterpart of [119, eq. (4.23.16)].

Lemma 4.5. *If $A \in \mathbb{C}^{n\times n}$ has no eigenvalues ± 1 then*

$$\operatorname{acos} A + \operatorname{asin} A = \frac{\pi}{2} I. \qquad (4.11)$$

Proof. Using the addition formula for the cosine we find that $\cos(\frac{\pi}{2}I - \operatorname{asin} A) = A$, so $\frac{\pi}{2}I - \operatorname{asin} A$ is some inverse cosine of A. That it is the principal inverse cosine is easily seen from Definition 4.2 (a) and (b). □

A known identity for scalars is $\operatorname{acosh} z = \pm i \operatorname{acos} z$ [2, eq. (4.6.15)]. The correct choice of sign depends on the complex argument of $1 - z$ (see Corless et al. [43, Sec. 6.2]). In the next result we show that the ± 1 term can be explicitly expressed in terms of the sign function and generalize the identity to matrices. We also generalize a corresponding identity for asinh.

Theorem 4.6. *If $A \in \mathbb{C}^{n\times n}$ has no eigenvalues ± 1 then*

$$\operatorname{acosh} A = i \operatorname{sign}(-iA) \operatorname{acos} A \quad \text{if } A \text{ has no eigenvalues in } (0,1], \tag{4.12}$$

$$\operatorname{asinh}(iA) = i \operatorname{asin} A. \tag{4.13}$$

Proof. From (4.4) along with the fact that cosh is an even function, we see that if X is an inverse cosine of A then $\pm iX$ is an inverse hyperbolic cosine of A. By passing to the Jordan canonical form and applying the argument to each Jordan block it follows that $i \operatorname{sign}(-iA) \operatorname{acos} A$ is some hyperbolic inverse cosine of A, and we need to show that it is the principal hyperbolic inverse cosine. We therefore need to show that the eigenvalues of $i \operatorname{sign}(-iA) \operatorname{acos} A$ satisfy the conditions in Definition 4.2 (c), which is equivalent to showing that $i \operatorname{sign}(-iz) \operatorname{acos} z$ satisfies these conditions for all $z \in \mathbb{C} \setminus (0,1]$.

Write $\operatorname{acos} z = x + iy$, where $z \in \mathbb{C}$ and $x, y \in \mathbb{R}$. We can also write $z = \cos(x + iy)$, which, using the addition formula for cosine, we can expand to $z = \cos x \cos(iy) - \sin x \sin(iy)$. Assuming that $y \neq 0$ and $x \in (0, \pi)$, we have $\operatorname{sign}(-iz) = \operatorname{sign}(i \sin x \sin(iy))$, because $\cos x$, $\sin x$, and $\cos(iy)$ are all real, and $\sin(iy)$ is pure imaginary. Since $x \in (0, \pi)$, $\sin x > 0$ and $\sin(iy) = i \sinh y$, we have $\operatorname{sign}(-iz) = \operatorname{sign}(-\sinh y) = -\operatorname{sign} y$. Finally, $i \operatorname{sign}(-iz) \operatorname{acos} z = -i \operatorname{sign}(y)(x + iy)$, which has real part $y \operatorname{sign} y > 0$ and imaginary part $-x \operatorname{sign} y \in (-\pi, \pi)$, satisfying Definition 4.2 (c) (i). If $y = 0$, then $z \in (-1, 1)$, and now we consider in turn $z \in (-1, 0]$ and $z \in (0, 1)$. In the former case, $x \in [\pi/2, \pi)$ and so $i \operatorname{sign}(-iz) \operatorname{acos} z \in i[\pi/2, \pi)$. In the other case, $x \in (0, \pi/2)$ and so $i \operatorname{sign}(-iz) \operatorname{acos} z \in i(-\pi/2, 0)$, which is not in the range of the principal branch of acosh. This means that Definition 4.2 (c) (ii)

is satisfied for $z \in (-1,0]$, but it is not satisfied for $z \in (0,1)$. If $x = 0$, by Definition 4.2 (a) (ii) we have $y > 0$, and so $z > 1$ and $i\,\mathrm{sign}(-iz)\mathrm{acos}z = y$, which satisfies Definition 4.2 (c) (i). Similarly, if $x = \pi$, then $y < 0$ by Definition 4.2 (a) (iii) and so $z < -1$ and $i\,\mathrm{sign}(-iz)\mathrm{acos}z = -y + \pi i$, which satisfies Definition 4.2 (c) (iii).

Turning to (4.13), from (4.4) we see that if X is an inverse sine of A then iX is some inverse hyperbolic sine of iA. We therefore just need to check that $i\,\mathrm{asin}\,A$ is the principal inverse hyperbolic sine of iA, and this reduces to the scalar case, which is a known identity [2, eq. (4.6.14)]. □

The following result will be needed to prove the next set of identities.

Lemma 4.7. *For $A \in \mathbb{C}^{n\times n}$ with no eigenvalues ± 1,*

$$(I - A)^{1/2}(I + A)^{1/2} = (I - A^2)^{1/2}. \tag{4.14}$$

Moreover, if all the eigenvalues of A have arguments in the interval $(-\pi/2, \pi/2]$ then

$$(A - I)^{1/2}(A + I)^{1/2} = (A^2 - I)^{1/2}. \tag{4.15}$$

Proof. The result is given for scalars in [27, Lem. 2]. It follows for matrices by [80, Thm. 1.20], [89, Thm. 6.2.27 (2)]. □

Note that, unlike (4.14), the identity (4.15) does not hold for all A!

The formulas in the next result will be useful in the construction of algorithms for computing acos, asin, acosh, and asinh in Section 4.5. These formulas do not follow directly from the scalar addition formulas in [119, Secs 4.24(iii), 4.38(iii)] because the latter formulas do not specify the branches of the constituent terms.

Theorem 4.8. *For $A \in \mathbb{C}^{n\times n}$, assuming that A has no eigenvalues at the branch*

points of the respective functions,

$$\operatorname{acos} A = 2\operatorname{acos}\left(\left(\frac{I+A}{2}\right)^{1/2}\right), \tag{4.16}$$

$$\operatorname{asin} A = 2\operatorname{asin}\left(\frac{(I+A)^{1/2}-(I-A)^{1/2}}{2}\right), \tag{4.17}$$

$$\operatorname{acosh} A = 2\operatorname{acosh}\left(\left(\frac{I+A}{2}\right)^{1/2}\right), \tag{4.18}$$

$$\operatorname{asinh} A = 2\operatorname{asinh}\left(\frac{i(I-iA)^{1/2}-i(I+iA)^{1/2}}{2}\right). \tag{4.19}$$

Proof. To prove (4.16) we use the first and second logarithmic representations (4.7) of acos, in that order:

$$\begin{aligned}
2\operatorname{acos}\left(\left(\frac{I+A}{2}\right)^{1/2}\right) &= -2i\log\left(\left(\frac{I+A}{2}\right)^{1/2}+i\left(I-\frac{I+A}{2}\right)^{1/2}\right)\\
&= -2i\log\left(\left(\frac{I+A}{2}\right)^{1/2}+i\left(\frac{I-A}{2}\right)^{1/2}\right)\\
&= \operatorname{acos} A.
\end{aligned}$$

The proof of (4.18) is analogous to that of (4.16) but requires the use of

$$\left(\left(\frac{I+A}{2}\right)^{1/2}-I\right)^{1/2}\left(\left(\frac{I+A}{2}\right)^{1/2}+I\right)^{1/2}=\left(\frac{I+A}{2}-I\right)^{1/2}.$$

The latter equality is valid by Lemma 4.7, since $((I+A)/2)^{1/2}$ has eigenvalues with arguments in the interval $(-\pi/2,\pi/2]$ by the definition of the principal square root.

For the proof of (4.17) we use the logarithmic representation (4.8) of asin. Denoting $B = ((I+A)^{1/2}-(I-A)^{1/2})/2$, after some manipulations we have $iA+(I-A^2)^{1/2} = (iB+(I-B^2)^{1/2})^2$. It is straightforward to show that for any $z\in\mathbb{C}$, $\operatorname{Re}(iz\ +(1-z^2)^{1/2})\geq 0$, from which we can conclude that $(iA+(I-A^2)^{1/2})^{1/2} = iB+(I-B^2)^{1/2}$. Taking logarithms, and using Corollary 3.17,

$$\begin{aligned}
\frac{1}{2}\operatorname{asin} A &= -\frac{1}{2}i\log(iA+(I-A^2)^{1/2})\\
&= -i\log(iB+(I-B^2)^{1/2})\\
&= \operatorname{asin} B,
\end{aligned}$$

which is (4.17). To show that (4.19) holds, we use (4.17) and the relation (4.13) between asin and asinh. ∎

We will also use the formulas in the next result, which relate the trigonometric functions cos and sin and their inverses acos and asin, and generalize formulas for scalars in [119, Table 4.16.3].

Lemma 4.9. *If $A \in \mathbb{C}^{n\times n}$ has no eigenvalues ± 1 then*

$$\sin(\operatorname{acos} A) = \cos(\operatorname{asin} A) = (I - A^2)^{1/2}.$$

Proof. Using the exponential form (4.2) of the sine and the logarithmic representation of acos given in Theorem 4.4, we write

$$\begin{aligned}\sin(\operatorname{acos} A) &= \frac{\mathrm{e}^{i\operatorname{acos} A} - (\mathrm{e}^{i\operatorname{acos} A})^{-1}}{2i} \\ &= \frac{A + i(I - A^2)^{1/2} - \big(A + i(I - A^2)^{1/2}\big)^{-1}}{2i}.\end{aligned}$$

But $(A + i(I - A^2)^{1/2})^{-1} = A - i(I - A^2)^{1/2}$, so

$$\sin(\operatorname{acos} A) = \frac{A + i(I - A^2)^{1/2} - \big(A - i(I - A^2)^{1/2}\big)}{2i} = (I - A^2)^{1/2}.$$

In a similar way, it can be shown that $\cos(\operatorname{asin} A) = (I - A^2)^{1/2}$. ∎

For the next results we need to use the matrix unwinding function, which was defined in (3.5) as

$$\mathcal{U}(A) = \frac{A - \log \mathrm{e}^A}{2\pi i}.$$

We now give summation formulas for the principal inverse sine and cosine functions. These identities are known to hold for real scalars, but by using the matrix unwinding function we can generalize them to complex square matrices.

In the remaining results of this section we make assumptions that are stronger than A having no eigenvalues at the branch points of the respective inverse functions. This is done so that we can obtain necessary and sufficient conditions for identities to hold.

The first result is given for scalars in [119, eq. (4.24.13)] for the multivalued inverse sine, with the branch for each occurrence of an inverse sine not specified.

Theorem 4.10. *For all $A, B \in \mathbb{C}^{n\times n}$ with no eigenvalues in Ω and such that $AB = BA$,*

$$\operatorname{asin} A + \operatorname{asin} B = \operatorname{asin}\big(A(I - B^2)^{1/2} + B(I - A^2)^{1/2}\big)$$

if and only if all the eigenvalues of $-AB + (I - A^2)^{1/2}(I - B^2)^{1/2}$ *have arguments in the interval* $(-\pi/2, \pi/2]$.

Proof. Applying the logarithmic representation (4.8) and the formula describing the logarithm of a matrix product via the unwinding function Lemma 3.19, we have

$$\begin{aligned}
\operatorname{asin} A + \operatorname{asin} B &= -i \log\big(iA + (I - A^2)^{1/2}\big) - i \log\big(iB + (I - B^2)^{1/2}\big) \\
&= -i \log\big(\big(iA + (I - A^2)^{1/2}\big)\big(iB + (I - B^2)^{1/2}\big)\big) \\
&\quad + 2\pi \mathcal{U}\big(\log\big(iA + (I - A^2)^{1/2}\big) + \log\big(iB + (I - B^2)^{1/2}\big)\big) \\
&= -i \log\big(\big(iA + (I - A^2)^{1/2}\big)\big(iB + (I - B^2)^{1/2}\big)\big) \\
&\quad + 2\pi \mathcal{U}\big(i \operatorname{asin} A + i \operatorname{asin} B\big).
\end{aligned}$$

Expanding the product and rearranging, using the fact that A and B commute, gives

$$\big(iA + (I - A^2)^{1/2}\big)\big(iB + (I - B^2)^{1/2}\big) = iC - AB + (I - A^2)^{1/2}(I - B^2)^{1/2},$$

where $C = A(I - B^2)^{1/2} + B(I - A^2)^{1/2}$. We also note that

$$\big(-AB + (I - A^2)^{1/2}(I - B^2)^{1/2}\big)^2 = I - C^2,$$

and using Lemma 3.18 we have

$$\big(I - C^2\big)^{1/2} = \big(-AB + (I - A^2)^{1/2}(I - B^2)^{1/2}\big) \mathrm{e}^{-\pi i \mathcal{U}\left(2 \log\left(-AB + (I - A^2)^{1/2}(I - B^2)^{1/2}\right)\right)}.$$

Since A and B have no eigenvalues in Ω, $\operatorname{asin} A$ and $\operatorname{asin} B$ both have eigenvalues with real parts in the interval $(-\frac{\pi}{2}, \frac{\pi}{2})$. Using the commutativity of A and B and Lemma 3.8 we then have $\mathcal{U}(i \operatorname{asin} A + i \operatorname{asin} B) = 0$. We can finally write

$$\operatorname{asin} A + \operatorname{asin} B = -i \log\big(iC + \big(I - C^2\big)^{1/2} \mathrm{e}^{\pi i \mathcal{U}\left(2 \log\left(-AB + (I - A^2)^{1/2}(I - B^2)^{1/2}\right)\right)}\big).$$

By Lemma 3.8 the unwinding term vanishes if and only if the arguments of all the eigenvalues of $-AB + (I - A^2)^{1/2}(I - B^2)^{1/2}$ lie in the interval $(-\pi/2, \pi/2]$. □

Now we give an analogous result for the inverse cosine.

Theorem 4.11. *For all $A, B \in \mathbb{C}^{n\times n}$ with no eigenvalues in Ω and such that $AB = BA$,*

$$\operatorname{acos} A + \operatorname{acos} B = \operatorname{acos}\bigl(AB - (I - A^2)^{1/2}(I - B^2)^{1/2}\bigr) \tag{4.20}$$

if and only if the arguments of all the eigenvalues of $iA(I-B^2)^{1/2}+iB(I-A^2)^{1/2}$ lie in the interval $(-\pi/2, \pi/2]$ and the real parts of the eigenvalues of $\operatorname{acos} A + \operatorname{acos} B$ lie in $[0, \pi]$.

Proof. We omit the proof because it follows the same framework as the proof of Theorem 4.10. □

By definition, $\cos(\operatorname{acos} A) = A$, but the inverse relation $\operatorname{acos}(\cos A) = A$ does not always hold. In the next few theorems we give explicit formulas for $\operatorname{acos}(\cos A)$ and the counterparts for the sine and the inverse hyperbolic cosine and sine, and identify when these formulas reduce to A. These "round trip" formulas are new even in the scalar case. We note that scalar functional identities relating all four functions and their respective inverses are given in [42, App. B], but they have the unattractive feature that the identity for $\operatorname{acos}(\cos z)$ involves $\sin z$ and similarly for the other identities.

Theorem 4.12. *If $A \in \mathbb{C}^{n\times n}$ has no eigenvalue with real part of the form $k\pi$, $k \in \mathbb{Z}$, then*

$$\operatorname{acos}(\cos A) = \bigl(A - 2\pi\,\mathcal{U}(iA)\bigr)\operatorname{sign}\bigl(A - 2\pi\,\mathcal{U}(iA)\bigr).$$

Proof. Let $B = A - 2\pi\,\mathcal{U}(iA)$. We first show that $\cos(B \operatorname{sign} B) = \cos A$. With $G = \operatorname{sign} B$ we have $\cos B = \cos(BG)$, which can be seen using the Jordan canonical form definitions of cos and sign along with the fact that $\cos(-X) = \cos X$ for any matrix X. Using the exponential representation (4.2) of the cosine function,

$$\begin{aligned}\cos B &= \frac{\mathrm{e}^{iB} + \mathrm{e}^{-iB}}{2}\\ &= \frac{\mathrm{e}^{i(A-2\pi\,\mathcal{U}(iA))} + \mathrm{e}^{-i(A-2\pi\,\mathcal{U}(iA))}}{2}\\ &= \frac{\mathrm{e}^{iA}\mathrm{e}^{-2\pi i\,\mathcal{U}(iA)} + \mathrm{e}^{-iA}\mathrm{e}^{2\pi i\,\mathcal{U}(iA)}}{2}.\end{aligned}$$

Now $\mathrm{e}^{2\pi i\mathcal{U}(iA)} = \mathrm{e}^{-2\pi i\mathcal{U}(iA)} = I$, since $\mathcal{U}(iA)$ is diagonalizable and has integer eigenvalues, so

$$\cos(BG) = \cos B = \frac{\mathrm{e}^{iA} + \mathrm{e}^{-iA}}{2} = \cos A.$$

Finally, since $iB = iA - 2\pi i\mathcal{U}(iA) = \log \mathrm{e}^{iA}$ by the definition (3.5) of the unwinding function, iB has eigenvalues with imaginary parts in the interval $(-\pi, \pi]$, hence B has eigenvalues with real parts in the interval $(-\pi, \pi]$. Therefore $B \operatorname{sign} B$ has eigenvalues with real parts in the interval $[0, \pi]$. We note that the end points of this interval are excluded because of the conditions in the statement of the theorem. Therefore the eigenvalues of B satisfy the condition in Definition 4.2 (a)(i). □

The following corollary of Theorem 4.12 gives necessary and sufficient conditions under which $A = \operatorname{acos}(\cos A)$ holds.

Corollary 4.13. *For $A \in \mathbb{C}^{n\times n}$ with no eigenvalue with real part of the form $k\pi$, $k \in \mathbb{Z}$, $\operatorname{acos}(\cos A) = A$ if and only if every eigenvalue of A has real part in the interval $(0, \pi)$.*

Proof. If all the eigenvalues of A satisfy the condition of this corollary, then, by Lemma 3.8, we have $\mathcal{U}(iA) = 0$. Then, since $\operatorname{sign} A = I$, by Theorem 4.12 we have $\operatorname{acos}(\cos A) = A$. Conversely, if $\operatorname{acos}(\cos A) = A$, then, since the condition of the corollary rules out A having an eigenvalue with real part 0 or π, the eigenvalues of A have real parts in the interval $(0, \pi)$, by Definition 4.2 (a). □

Theorem 4.14. *If $A \in \mathbb{C}^{n\times n}$ has no eigenvalue with real part of the form $(2k + 1)\pi/2$, $k \in \mathbb{Z}$, then*

$$\operatorname{asin}(\sin A) = \mathrm{e}^{\pi i\mathcal{U}(2iA)}\big(A - \pi\,\mathcal{U}(2iA)\big).$$

Proof. Let $C = A - \pi\,\mathcal{U}(2iA)$ and $H = \mathrm{e}^{\pi i\mathcal{U}(2iA)}$. We will first prove that $\sin(HC) = \sin A$.

The matrix unwinding function $\mathcal{U}(2iA)$ is diagonalizable with integer eigenvalues, so the matrix H is diagonalizable with eigenvalues equal to ± 1. It is not hard to show that $\sin(HC) = H \sin C$. Now

$$\sin C = \sin\big(A - \pi\,\mathcal{U}(2iA)\big) = \sin A \cos\big(\pi\,\mathcal{U}(2iA)\big) - \cos A \sin\big(\pi\,\mathcal{U}(2iA)\big).$$

Since $\mathcal{U}(2iA)$ is diagonalizable and has integer eigenvalues, $\sin(\pi\mathcal{U}(2iA)) = 0$. From the properties of H described above, $H = \mathrm{e}^{\pi i\mathcal{U}(2iA)} = \mathrm{e}^{-\pi i\mathcal{U}(2iA)}$ and so

$$\cos(\pi\mathcal{U}(2iA)) = \frac{\mathrm{e}^{\pi i\mathcal{U}(2iA)} + \mathrm{e}^{-\pi i\mathcal{U}(2iA)}}{2} = H.$$

Therefore $\sin(HC) = H\sin C = H^2\sin A = \sin A$, which completes the first part of the proof.

Finally, we show that every eigenvalue of HC satisfies the condition in Definition 4.2 (b). We note that the real parts of the eigenvalues of HC lie in $[-\pi, 2/\pi/2]$ and the conditions in the statement of the theorem exclude the endpoints of this interval, so conditions (ii) and (iii) in Definition 4.2 (b) need not be checked. Using the definition (3.5) of the unwinding function we have

$$iC = iA - \pi i\mathcal{U}(2iA) = iA - \pi i\left(\frac{2iA - \log \mathrm{e}^{2iA}}{2\pi i}\right) = \frac{\log \mathrm{e}^{2iA}}{2}.$$

Here log is the principal matrix logarithm, so C has eigenvalues with real parts in the interval $(-\pi/2, \pi/2]$ and therefore HC has eigenvalues with real parts in the interval $[-\pi/2, \pi/2]$. But, as already noted, the end points of this interval are excluded because of the assumptions in the statement of the theorem. □

Corollary 4.15. *For $A \in \mathbb{C}^{n\times n}$ with no eigenvalue with real part of the form $(2k+1)/2\pi$, $k \in \mathbb{Z}$, $\operatorname{asin}(\sin A) = A$ if and only if every eigenvalue of A has real part in the interval $(-\pi/2, \pi/2)$.*

Proof. If the eigenvalues of A have real parts in the interval $(-\pi/2, \pi/2)$ then by Lemma 3.8 we have $\mathcal{U}(2iA) = 0$. Applying Theorem 4.14 we then have $\operatorname{asin}(\sin A)) = A$.

Conversely, if $\operatorname{asin}(\sin A) = A$ then, since the condition of the corollary rules out A having an eigenvalue with real part $\pm\pi/2$, the eigenvalues of A have real parts in the interval $(-\pi/2, \pi/2)$, by Definition 4.2 (b). □

Similar results hold for the inverse hyperbolic cosine and sine functions.

Theorem 4.16. *For $A \in \mathbb{C}^{n\times n}$ with no eigenvalue with imaginary part of the form $k\pi$, with odd $k \in \mathbb{Z}$, and no pure imaginary eigenvalue,*

$$\operatorname{acosh}(\cosh A) = \big(A - 2\pi i\mathcal{U}(A)\big)\operatorname{sign}\big(A - 2\pi i\mathcal{U}(A)\big).$$

Proof. Let $B = A - 2\pi i\mathcal{U}(A)$. We follow the same framework as in the proofs of the previous two results. First, we note that $\cosh(B\operatorname{sign} B) = \cosh B$. Expressing cosh in terms of exponentials we have

$$\begin{aligned}\cosh B &= \frac{1}{2}\bigl(\mathrm{e}^{A-2\pi i\mathcal{U}(A)} + \mathrm{e}^{-A+2\pi i\mathcal{U}(A)}\bigr)\\ &= \frac{1}{2}\bigl(\mathrm{e}^{A} + \mathrm{e}^{-A}\bigr) = \cosh A,\end{aligned}$$

where we used $\mathrm{e}^{\pm 2\pi i\mathcal{U}(A)} = I$ by Lemma 3.12.

Finally, we have to show that the eigenvalues of $B\operatorname{sign} B$ satisfy the requirements of Definition 4.2 (c). Using the definition of the unwinding function, $B = A - 2\pi i\mathcal{U}(A) = \log \mathrm{e}^{A}$, we see that the imaginary parts of the eigenvalues of B lie in $(-\pi, \pi]$. Therefore each eigenvalue of $B\operatorname{sign} B$ has eigenvalues with nonnegative real part and imaginary part in the interval $[-\pi, \pi]$. The end points of this interval and the case when the eigenvalues of $B\operatorname{sign} B$ are pure imaginary are excluded because of the assumptions in the statement of the theorem. Therefore the conditions of Definition 4.2(c) are satisfied. ∎

Corollary 4.17. *For $A \in \mathbb{C}^{n\times n}$ with no eigenvalue with imaginary part of the form $k\pi$, for odd $k \in \mathbb{Z}$, and no pure imaginary eigenvalue,* $\operatorname{acosh}(\cosh A) = A$ *if and only if every eigenvalue of A has imaginary part in the interval $(-\pi, \pi)$ and positive real part.*

Proof. If the eigenvalues of A all have imaginary parts in the interval $(-\pi, \pi)$ then $\mathcal{U}(A) = 0$ and if they all have positive real parts then $\operatorname{sign} A = I$. Therefore Theorem 4.16 gives $\operatorname{acosh}(\cosh A) = A$. Conversely, if $\operatorname{acosh}(\cosh A) = A$ and if A satisfies the conditions of the corollary then the eigenvalues of A have imaginary parts in the interval $(-\pi, \pi)$ and positive real parts, by Definition 4.2 (c)(i). ∎

Theorem 4.18. *If $A \in \mathbb{C}^{n\times n}$ has no eigenvalue with imaginary part of the form $(2k+1)\pi i/2$, $k \in \mathbb{Z}$, then*

$$\operatorname{asinh}(\sinh A) = \mathrm{e}^{\pi i\mathcal{U}(2A)}\bigl(A - \pi i\mathcal{U}(2A)\bigr).$$

Proof. Suppose first that A does not have any eigenvalues whose imaginary parts are of the form $(2k+1)\pi/2, k \in \mathbb{Z}$. This implies that $\sin(-iA)$ has no eigenvalues

± 1, so we can use the identity $\sinh A = i\sin(-iA)$ from (4.4) and Theorem 4.6 and Theorem 4.14 to write

$$\begin{aligned}\operatorname{asinh}(\sinh A) &= \operatorname{asinh}\bigl(i\sin(-iA)\bigr)\\ &= i\operatorname{asin}\bigl(\sin(-iA)\bigr)\\ &= i\mathrm{e}^{\pi i\mathcal{U}(2A)}\bigl(-iA - \pi\mathcal{U}(2A)\bigr)\\ &= \mathrm{e}^{\pi i\mathcal{U}(2A)}\bigl(A - \pi i\mathcal{U}(2A)\bigr). \qquad \square\end{aligned}$$

Corollary 4.19. *For $A \in \mathbb{C}^{n\times n}$ with no eigenvalue with imaginary part of the form $(2k+1)\pi i/2$, $k \in \mathbb{Z}$, $\operatorname{asinh}(\sinh A) = A$ if and only if every eigenvalue of A has imaginary part in the interval $(-\pi/2, \pi/2)$.*

Proof. If every eigenvalue of A has imaginary part in $(-\pi/2, \pi/2)$, then by Lemma 3.8 we have $\mathcal{U}(2A) = 0$ and Theorem 4.18 gives $\operatorname{asinh}(\sinh A) = A$.

Conversely, if $\operatorname{asinh}(\sinh A) = A$, by Definition 4.2 (d), since d(ii) and d(iii) are excluded by the assumptions on A, the eigenvalues of A must have imaginary parts in the interval $(-\pi/2, \pi/2)$. $\square$

4.4 Conditioning

We introduced the absolute condition number of a function f at the matrix A in (1.11) as

$$\operatorname{cond}_{\mathrm{abs}}(f, A) = \max_{E\neq 0} \frac{\|L_f(A,E)\|}{\|E\|}.$$

Here, L_f is the Fréchet derivative of f, which is a linear operator, which we defined in Section 1.2.

To study the conditioning of the inverse sine and cosine we need only study one of them, in view of the relation given in the next result between the respective Fréchet derivatives.

Lemma 4.20. *For $A \in \mathbb{C}^{n\times n}$ with no eigenvalues in Ω in (4.5),*

$$L_{\operatorname{acos}}(A,E) + L_{\operatorname{asin}}(A,E) = 0.$$

Proof. Fréchet differentiate (4.11). □

A simple relation also exists between the Fréchet derivatives of asin and asinh.

Lemma 4.21. *For $A \in \mathbb{C}^{n\times n}$ with no eigenvalues in Ω in (4.5),*

$$L_{\mathrm{asin}}(A,E) = L_{\mathrm{asinh}}(iA,E).$$

Proof. Fréchet differentiate (4.13) using the chain rule [80, Thm. 3.4]. □

We now study further the Fréchet derivative of acos. Assume that A has no eigenvalues in Ω. By [80, Thm. 3.5] we have

$$L_{\cos}\left(\operatorname{acos} A, L_{\operatorname{acos}}(A,E)\right) = E. \tag{4.21}$$

Recall the integral representation of the Fréchet derivative of the matrix cosine function [80, Sec. 12.2]

$$L_{\cos}(A,E) = -\int_0^1 \left[\cos\left(A(1-t)\right) E \sin(At) + \sin\left(A(1-t)\right) E \cos(At)\right] dt. \tag{4.22}$$

Substituting into the relation (4.21) we find that the Fréchet derivative of acos satisfies

$$\begin{aligned} E = -\int_0^1 \big[&\cos\left((1-t)\operatorname{acos} A\right) L_{\operatorname{acos}}(A,E) \sin\left(t \operatorname{acos} A\right) \\ &+ \sin\left((1-t)\operatorname{acos} A\right) L_{\operatorname{acos}}(A,E) \cos\left(t \operatorname{acos} A\right)\big]\, dt. \end{aligned} \tag{4.23}$$

If A and E commute the relation (4.23) simplifies to $E = -L_{\operatorname{acos}}(A,E)\sin\left(\operatorname{acos} A\right)$. Now we can apply Lemma 4.9 to obtain the more useful expression

$$L_{\operatorname{acos}}(A,E) = -E(I-A^2)^{-1/2} \qquad (AE = EA). \tag{4.24}$$

Setting $E = I$ in (4.24) gives $L_{\operatorname{acos}}(A,I) = -(I-A^2)^{-1/2}$ and for any subordinate norm we obtain the bound

$$\operatorname{cond}_{\mathrm{abs}}(\operatorname{acos}, A) \geq \|(I-A^2)^{-1/2}\|.$$

The condition number is necessarily large when A has an eigenvalue close to 1 or -1, which are the branch points of acos.

One would also expect acos to be ill conditioned when a pair of eigenvalues lie close to, but on either side of, the branch cut. This is revealed by applying a general lower bound from [80, Thm. 3.14], which gives

$$\operatorname{cond}_{\mathrm{abs}}(\operatorname{acos}, A) \geq \max_{\lambda,\mu \in \Lambda(A)} |\operatorname{acos}[\mu, \lambda]|,$$

where $\Lambda(A)$ is the spectrum of A and the lower bound contains the divided difference $\operatorname{acos}[\lambda, \mu] = (\operatorname{acos}\lambda - \operatorname{acos}\mu)/(\lambda - \mu)$. For example, if $\lambda = -2 + \epsilon i$ and $\mu = -2 - \epsilon i$, with $0 < \epsilon \ll 1$, then $\operatorname{acos}\lambda = \pi - \operatorname{acos}2 + O(\epsilon)$ and $\operatorname{acos}\mu = \pi + \operatorname{acos}2 + O(\epsilon)$, so $\operatorname{acos}[\lambda, \mu] = O(1/\epsilon)$.

4.5 Algorithms

Lemma 4.5 and Theorem 4.6 show that if we have an algorithm for computing any one of the four functions $\operatorname{acos} A$, $\operatorname{asin} A$, $\operatorname{acosh} A$, and $\operatorname{asinh} A$ then the others can be obtained from it, although this may necessitate using complex arithmetic for a real problem. In the next subsection we propose an algorithm for computing the principal matrix inverse cosine based on Padé approximation. In Section 4.5.2 we consider an alternative algorithm for computing the inverse trigonometric and inverse hyperbolic functions via their logarithmic representations given in Theorem 4.4.

We exploit a Schur factorization $A = QTQ^*$, where Q is a unitary matrix and T is upper triangular with the eigenvalues of A on its diagonal, along with the property $f(A) = Qf(T)Q^*$. The problems of computing $\operatorname{acos} A$, $\operatorname{asin} A$, $\operatorname{acosh} A$, and $\operatorname{asinh} A$ are thus reduced to computing the same functions of the triangular matrix T. We will explain how Schur-free variants of the algorithms can also be constructed; these are of interest for situations in which a highly efficient implementation of the Schur decomposition is not available (for example, in certain parallel computing environments).

4.5.1 Schur–Padé algorithm

We develop an algorithm analogous to the inverse scaling and squaring method for computing the matrix logarithm.

For $\rho(A) < 1$, where ρ is the spectral radius, we can write $\operatorname{acos} A$ as the power series [119, eq. (4.24.1)]

$$\operatorname{acos} A = \frac{\pi}{2} I - \sum_{k=0}^{\infty} \frac{\binom{2k}{k}}{4^k(2k+1)} A^{2k+1}, \quad \rho(A) < 1.$$

Alternatively, we can expand as a series in $I - A$ [119, eq. (4.24.2)]:

$$\operatorname{acos} A = 2^{1/2}(I-A)^{1/2} \sum_{k=0}^{\infty} \frac{\binom{2k}{k}}{8^k(2k+1)} (I-A)^k, \quad \rho(I-A) < 2.$$

Here, to ensure the existence of $(I-A)^{1/2}$, we require that A has no eigenvalues equal to 1. Replacing A by $I - A$ gives, for nonsingular A,

$$\operatorname{acos}(I-A) = (2A)^{1/2} \sum_{k=0}^{\infty} \frac{\binom{2k}{k}}{8^k(2k+1)} A^k, \quad \rho(A) < 2. \tag{4.25}$$

We will compute acos using Padé approximants of the function $f(x) = (2x)^{-1/2}\operatorname{acos}(1-x)$, which (4.25) shows is represented by a power series in x that converges for $|x| \le 2$ and so should be well approximated by Padé approximants near the origin. Let $r_m(x) = p_m(x)/q_m(x)$ denote the diagonal $[m/m]$ Padé approximant of $f(x)$, so that $p_m(x)$ and $q_m(x)$ are polynomials of degrees at most m.

We now consider the backward error of approximating acos. For $A \in \mathbb{C}^{n\times n}$ we define $h_m : \mathbb{C}^{n\times n} \to \mathbb{C}^{n\times n}$ by

$$(2A)^{1/2} r_m(A) = \operatorname{acos}\bigl(I - A + h_m(A)\bigr),$$

assuming that all of the eigenvalues of $(2A)^{1/2} r_m(A)$ have real parts in the interval $(0, \pi)$. We can rewrite this equation as

$$h_m(A) = \cos\bigl((2A)^{1/2} r_m(A)\bigr) - (I - A).$$

The relative backward error in approximating $\operatorname{acos}(I-A)$ by $(2A)^{1/2} r_m(A)$ is given by $\|h_m(A)\|/\|A\|$ and we wish to bound it by the unit roundoff for IEEE double precision arithmetic, $u = 2^{-53} \approx 1.11 \times 10^{-16}$, that is, we would like to ensure that

$$\frac{\|h_m(A)\|}{\|A\|} = \frac{\|\cos\bigl((2A)^{1/2} r_m(A)\bigr) - (I-A)\|}{\|A\|} \le u. \tag{4.26}$$

We now follow the same framework as for backward error analysis of the exponential [5] and the cosine and sine [9, Sec. 2]. We have $r_m(x) = (2x)^{-1/2}\operatorname{acos}(1-x) +$

$O(x^{2m+1})$ and so $h_m(x) = \cos((2x)^{1/2} r_m(x)) - (1-x) = O(x^{2m+2})$, where the last equality is obtained after some manipulations.

We can write

$$h_m(A) = \sum_{\ell=0}^{\infty} c_\ell A^{2m+\ell+2} = A \sum_{\ell=0}^{\infty} c_\ell A^{2m+\ell+1},$$

for some coefficients c_ℓ. We now use [5, Thm. 4.2(a)] to obtain the bound on the relative backward error

$$\frac{\|h_m(A)\|}{\|A\|} \le \sum_{\ell=0}^{\infty} |c_\ell| \alpha_p(A)^{2m+\ell+1},$$

where

$$\alpha_p(A) = \max\bigl(\|A^p\|^{1/p}, \|A^{p+1}\|^{1/(p+1)}\bigr)$$

and p is an integer such that $2m+1 \ge p(p-1)$. It can be shown that $\alpha_1(A) \ge \alpha_2(A) \ge \alpha_3(A)$, but for $p \ge 4$ the relation between $\alpha_{p-1}(A)$ and $\alpha_p(A)$ depends on the matrix A. We need to find the smallest value of $\alpha_p(A)$ subject to the constraint $2m+1 \ge p(p-1)$.

With the definition

$$\beta_m = \max\left\{ \beta : \sum_{\ell=0}^{\infty} |c_\ell| \beta^{2m+\ell+1} \le u \right\},$$

the inequality $\alpha_p(A) \le \beta_m$ implies that the relative backward error is bounded by u. Table 4.5.1 gives the values of β_m for a range of values of m, determined experimentally using a combination of high precision arithmetic and symbolic calculations.

Table 4.5.1 also gives the number of matrix multiplications π_m required to evaluate the Padé approximant $r_m(A)$ of order $[m/m]$ using the Paterson–Stockmeyer scheme [80, Sec. 4.2 and Table 4.2], [124] for both p_m and q_m.

To ensure that $\alpha_p(A) \le \beta_m$ for a suitable value of m we use repeatedly the identity $\operatorname{acos} X = 2\operatorname{acos}\bigl(((I+X)/2)^{1/2}\bigr)$ in (4.16), which brings the argument close to the identity, as shown by the next result.

Lemma 4.22. *For any $X_0 \in \mathbb{C}^{n\times n}$, the sequence defined by*

$$X_{k+1} = \left(\frac{I + X_k}{2} \right)^{1/2} \tag{4.27}$$

satisfies $\lim_{k\to\infty} X_k = I$.

Table 4.1: Values of β_m, values of p to be considered, and number of matrix multiplications π_m required to evaluate r_m.

m	1	2	3	4	5	6
β_m	3.44e-5	4.81e-3	3.97e-2	1.26e-1	2.59e-1	4.17e-1
$p \le$	2	2	3	3	3	4
π_m	0	1	2	3	4	4

m	7	8	9	10	11	12
β_m	5.81e-1	7.39e-1	8.84e-1	1.01	1.13	1.22
$p \le$	4	4	4	5	5	5
π_m	5	5	6	6	7	7

Proof. First, consider the scalar iteration $x_{k+1} = ((1+x_k)/2)^{1/2}$. It is easy to see that

$$x_{k+1} - 1 = \frac{x_k - 1}{2\left(\left(\frac{1+x_k}{2}\right)^{1/2} + 1\right)}$$

and hence that $|x_{k+1} - 1| \le |x_k - 1|/2$, since $\operatorname{Re}((1+x_k)/2])^{1/2} \ge 0$. Therefore $\lim_{k\to\infty} x_k = 1$. The function $((1+x)/2)^{1/2}$ is holomorphic for $\operatorname{Re} x \ge 0$ and furthermore its derivative at $x = 1$ satisfies $|\frac{d}{dx}(\frac{1}{2}(x+1)^{1/2}|_{x=1} = \frac{1}{4} < 1$. The convergence of the matrix iteration follows from a general result of Iannazzo [92, Thm. 3.20]. □

We apply the recurrence (4.27) with $X_0 = T$, selecting the scaling parameter s so that $\alpha_p(I - X_s) \le \beta_m$. To compute the square roots required to obtain X_s we use the Björck–Hammarling method [24], [80, Alg. 6.3]. Increasing the scaling parameter s by one has a cost of $n^3/3$ flops, so it is worth doing if it decreases the number π of (triangular) matrix multiplications, which also cost $n^3/3$ flops each, by more than 1. From the relation

$$(I - X_{s+1})(I + X_{s+1}) = I - X_{s+1}^2 = I - \frac{I + X_s}{2} = \frac{I - X_s}{2}, \tag{4.28}$$

it is clear that for large s (so that $\|X_s\|$ is of order 1) $\|I - X_{s+1}\| \approx \|I - X_s\|/4$. From the values of β_m in Table 4.5.1 we see that for $m \ge 9$ it is more efficient to continue the recursion and consequently use an approximant of a lower degree. Indeed for $m = 9$, $\beta_9/4 = 2.21\text{e-}1 < 2.59\text{e-}1 = \beta_5$, so the effect of taking an extra

step in the recursion would be that we could use an approximant of type [5/5] and so the number of matrix multiplications required would be reduced from 6 to 4.

In computing $\alpha_p(A)$ we avoid explicit computation of powers of A by estimating $\|A^p\|_1^{1/p}$ and $\|A^{p+1}\|_1^{1/(p+1)}$ using the block 1-norm estimator of Higham and Tisseur [87].

A further computational saving can be provided by computing a lower bound on the scaling parameter s. Denote by $D = \mathrm{diag}(T)$ the diagonal matrix containing the eigenvalues of A on its diagonal and observe that $\rho(I-D) = \rho(I-T) \le \alpha_p(I-T)$. The largest β we consider is β_8, and the inequality $\alpha_p(I-T) \le \beta_8$ also requires that $\rho(I-D) \le \beta_8$, so we can apply the recurrence (4.27) to the matrix D to obtain a lower bound s_0 on s at negligible cost.

We are now ready to state the algorithm for computing acos. In the pseudocode the statement "break" denotes that execution jumps to the first statement after the while loop.

Algorithm 4.23 (Schur–Padé algorithm)**.** *Given $A \in \mathbb{C}^{n\times n}$ with no eigenvalues equal to ± 1, this algorithm computes $X = \operatorname{acos} A$. The algorithm is intended for use with IEEE double precision arithmetic.*

Compute the Schur decomposition $A = QTQ^*$ (Q unitary, T upper triangular).
Find s_0, the smallest s such that $\rho(I - X_s) \le \beta_8$, where the X_s are the iterates from (4.27) with $X_0 = \mathrm{diag}(T)$.
for $i = 1\colon s_0$
    $T = \left(\frac{1}{2}(I+T)\right)^{1/2}$
end
$s = s_0$, $m = 0$
while $m = 0$
    $Z = I - T$
    Estimate $d_2(Z) = \|Z^2\|_1^{1/2}$.
    Estimate $d_3(Z) = \|Z^3\|_1^{1/3}$.
    $\alpha_2(Z) = \max(d_2, d_3)$
    if $\alpha_2(Z) \le \beta_1$, $m = 1$, break, end
    if $\alpha_2(Z) \le \beta_2$, $m = 2$, break, end

14 Estimate $d_4(Z) = \|Z^4\|_1^{1/4}$.
15 $\alpha_3(Z) = \max(d_3, d_4)$
16 if $\alpha_3(Z) \le \beta_3$, $m = 3$, break, end
17 if $\alpha_3(Z) \le \beta_4$, $m = 4$, break, end
18 if $\alpha_3(Z) \le \beta_5$, $m = 5$, break, end
19 Estimate $d_5(Z) = \|Z^5\|_1^{1/5}$.
20 $\alpha_4(Z) = \max(d_4, d_5)$
21 $\gamma(Z) = \min(\alpha_3(Z), \alpha_4(Z))$
22 if $\gamma(Z) \le \beta_6$, $m = 6$, break, end
23 if $\gamma(Z) \le \beta_7$, $m = 7$, break, end
24 if $\gamma(Z) \le \beta_8$, $m = 8$, break, end
25 $T = \left(\frac{1}{2}(I + T)\right)^{1/2}$
26 $s = s + 1$
27 end
28 Compute $U = r_m(Z)$ by using the Paterson–Stockmeyer scheme to evaluate $p_m(Z)$ and $q_m(Z)$ and then solving $q_m(Z)U = p_m(Z)$.
29 $Y = Z^{1/2}$
30 $V = 2^{1/2}UY$
31 $W = 2^s V$
32 $X = QWQ^*$

Cost: $25n^3$ flops for the Schur decomposition, $sn^3/3$ flops to compute the square roots for the scaling stage, $(\pi_m + 1)n^3/3$ flops to compute the Padé approximation of order $[m/m]$, $n^3/3$ flops for the final square root, and $3n^3$ flops to form X: about $(28\frac{2}{3} + \frac{\pi_m + s}{3})n^3$ flops in total.

A Schur-free variant of Algorithm 4.23 can be obtained by removing lines 1–5 and 32, setting $s = 0$ on line 6, and computing the square roots using (for example) a scaled Denman–Beavers iteration [80, Sec. 6.3]. For real A, a variant of the algorithm that uses only real arithmetic can be derived by using a real Schur decomposition on line 1 and using ideas from [8].

The other functions of interest can be computed by using Algorithm 4.23 in

conjunction with the formulas, from Lemma 4.5 and Theorem 4.6,

$$\operatorname{asin} A = (\pi/2)I - \operatorname{acos} A, \tag{4.29}$$

$$\operatorname{asinh} A = i \operatorname{asin}(-iA) = i\big((\pi/2)I - \operatorname{acos}(-iA)\big), \tag{4.30}$$

$$\operatorname{acosh} A = i \operatorname{sign}(-iA) \operatorname{acos} A \quad \text{if } A \text{ has no eigenvalues in } (0,1]. \tag{4.31}$$

The last relation requires computation of the matrix sign function of a triangular matrix (exploiting the Schur form), which can be done by [80, Alg. 5.5] at a cost of up to $2n^3/3$ flops, with a further $n^3/3$ flops for the final (triangular) matrix multiplication. A fast, blocked implementation of [80, Alg. 5.5] has recently been developed by Toledo [142]. For a Schur-free algorithm, the matrix sign function can be computed using a Newton algorithm or some other rational iteration [80, Chap. 5], [115]. Equation (4.31) is applicable only when A has no eigenvalue in the interval $(0,1]$. When this condition is not satisfied acosh can be computed using the logarithmic representations of acosh given in (4.9), as described in the next subsection. Alternatively, a special purpose algorithm could be designed, using analysis similar to that in Section 4.5.1.

4.5.2 Algorithms based on logarithmic formulas

Another way to compute the matrix inverse trigonometric and inverse hyperbolic functions is via their logarithmic representations given in Theorem 4.4. The most popular method for computing the matrix logarithm is the inverse scaling and squaring method. It was introduced by Kenney and Laub [98] and has undergone extensive development [7], [37], [39], [80, sect. 11.5], with special attention to computation in real arithmetic [8], [53]. The inverse scaling and squaring method is based on the relation $\log X = 2^s \log(X^{1/2^s})$ for $s \in \mathbb{Z}$, with s taken sufficiently large that $X^{1/2^s}$ is close to the identity matrix and a Padé approximant to $\log(1+x)$ used to approximate $\log(X^{1/2^s})$. In the most recent algorithms the degree of the Padé approximant is variable.

Using the first formula in (4.7) we obtain the following algorithm.

Algorithm 4.24. *Given $A \in \mathbb{C}^{n\times n}$ with no eigenvalues equal to ± 1, this algorithm computes $C = \operatorname{acos} A$ via the matrix logarithm.*

1 Compute the Schur decomposition $A = QTQ^*$.
2 $R = (I - T^2)^{1/2}$
3 Compute $X = -i \log(T + iR)$ using [7, Alg. 4.1].
4 $C = QXQ^*$

Cost: $25n^3$ flops for the Schur decomposition, $2n^3/3$ flops to compute R, $(\hat{s} + \hat{m})n^3/3$ flops to compute X (where $\hat{s}$ is the scaling parameter in the inverse scaling and squaring method and $\hat{m}$ is the degree of the Padé approximation used), plus $3n^3$ flops to form C: about $(28\frac{2}{3} + \frac{\hat{s}+\hat{m}}{3})n^3$ flops in total.

A Schur-free algorithm can be obtained by omitting the first and last lines of the algorithm, replacing T by A on lines 2–3, and computing the logarithm using [7, Alg. 5.2].

Corresponding algorithms for asin, acosh, and asinh are obtained by using (4.8), the first formula in (4.9), and (4.10).

In the `linear-algebra` package of GNU Octave (version 4.0.0) [70], the function `thfm.m` ("trigonometric/hyperbolic functions of square matrix") implements logarithmic formulas for acos, asin, acosh, and asinh. This function has two weaknesses. First, the formula used for acosh is $\operatorname{acosh} A = \log(A + (A^2 - I)^{1/2})$, which differs from (4.9) (cf. Lemma 4.7) and does not produce the principal branch as we have defined it. Second, the formulas are implemented as calls to `logm` and `sqrtm` and so two Schur decompositions are computed rather than one.

4.6 Numerical experiments

We present numerical experiments with the following algorithms.

- Algorithm 4.23, which computes acos by the Schur–Padé algorithm. In the case of asin, acosh, and asinh, the algorithm is used together with (4.29)–(4.31). In (4.31) the sign function of a triangular matrix is computed with the function `signm` from [77]. When (4.31) is not applicable we use the first logarithmic formula for acosh in (4.9).

- Algorithm 4.24 and its counterparts for asin, acosh, and asinh based on the logarithmic representations.

We note that an algorithm for computing the matrix inverse hyperbolic sine has been proposed by Cardoso and Silva Leite [36, Alg. 1]. They compute $\operatorname{asinh} A$ using its logarithmic representation (4.10). In computing the logarithm they use the relation $\log((1+x)/(1-x)) = 2\operatorname{atanh} x$, where atanh is the inverse hyperbolic tangent, along with Padé approximations of atanh. The degree of the Padé approximant is fixed at 8 and is not chosen optimally. For this reason we will not consider this algorithm further.

All computations are performed in MATLAB 2015b, for which the unit roundoff is $u \approx 1.11 \times 10^{-16}$.

We consider a set of 20 test matrices, which are mostly 10×10 and are based on matrices from the MATLAB `gallery` function, the Matrix Computation Toolbox [76], test problems provided with EigTool [147], and matrix exponential test problems [5]. Figure 4.2 gives the the relative errors $\|\operatorname{acos} A - \widehat{C}\|_1 / \|\operatorname{acos} A\|_1$. Here, an accurate $\operatorname{acos} A$ was obtained using 100-digit arithmetic with the Advanpix Multiprecision Computing Toolbox for MATLAB [3], exploiting the eigendecomposition $A = VDV^{-1}$ and the property $f(A) = Vf(D)V^{-1}$. For each matrix we also estimated the relative condition number

$$\operatorname{cond}_{\mathrm{rel}}(f, A) = \frac{\operatorname{cond}_{\mathrm{abs}}(f, A)\|A\|}{\|f(A)\|},$$

where $\operatorname{cond}_{\mathrm{abs}}$ is defined in (4.4), using the algorithm `funm_condest1` from the Matrix Function Toolbox [77], which implements [80, Alg. 3.22]. The latter algorithm requires the Fréchet derivatives $L_{\operatorname{acos}}(A, E)$, which are obtained using the identity (1.13)

$$\operatorname{acos}\left(\begin{bmatrix} A & E \\ 0 & A \end{bmatrix}\right) = \begin{bmatrix} \operatorname{acos} A & L_{\operatorname{acos}}(A, E) \\ 0 & \operatorname{acos} A \end{bmatrix}, \tag{4.32}$$

and we use Algorithm 4.23 for this computation. We use Lemmas 4.20 and 4.21 to obtain the Fréchet derivatives of asin and asinh, and the analog of (4.32) for acosh, along with (4.31), to obtain the Fréchet derivative of acosh.

Figures 4.3–4.5 give the 1-norm relative errors for asin, acosh, and sinh computed using the variants of Algorithm 4.24, based on the matrix logarithm.

For all four functions it can be seen that Algorithm 4.23 gives the best results overall and behaves in a forward stable fashion, that is, the relative error is not much

larger than $\mathrm{cond}_{\mathrm{rel}}(f,A)u$. The algorithms based on the logarithmic representations have a major disadvantage. The branch point of the logarithm is at zero, and so when the argument of the logarithm has an eigenvalue close to this branch point there may be large relative errors in computing the logarithm. However, the argument of the logarithm can have eigenvalues close to zero when the argument of asin, acos, asinh, or acosh is not close to a branch point of that function. Consider, for example, a matrix A with some eigenvalues with large negative imaginary parts. The corresponding eigenvalues of $A + i(I - A^2)^{1/2}$ are close to zero, which may be detrimental for the computation of the logarithm in Algorithm 4.24. We observed this for the matrix indexed 19 in Figure 4.2. The matrix is

$$A = \begin{bmatrix} 0 & b \\ -b & 0 \end{bmatrix}, \quad b = 1000,$$

with eigenvalues $\pm 1000i$. The eigenvalues of $A + i(I - A^2)^{1/2}$ are approximately $5 \times 10^{-4}i$ and $2000i$, so one of them is very close to zero and this is reflected in the relative error for Algorithm 4.24 for computing acos, which is $\|\operatorname{acos} A - \widehat{C}\|_1 / \|\operatorname{acos} A\|_1 \approx 1.98 \times 10^{-9}$ versus 3.68×10^{-16} for Algorithm 4.23. This is not surprising in view of the large difference between the (estimated) relative 1-norm condition numbers, which are 0.83 for $\operatorname{acos} A$ and 2.1×10^7 for $\log(A + i(I - A^2)^{1/2})$.

Finally, we reiterate that $\operatorname{acosh} A$ can be computed using Algorithm 4.23 and equation (4.31) only if A has no eigenvalue in the interval $(0, 1]$. This restriction was satisfied for all but two of the matrices—those indexed 3 and 7 in Figure 4.4. However, as we see from the figure, the acosh variant of Algorithm 4.24 provides a good alternative for such cases.

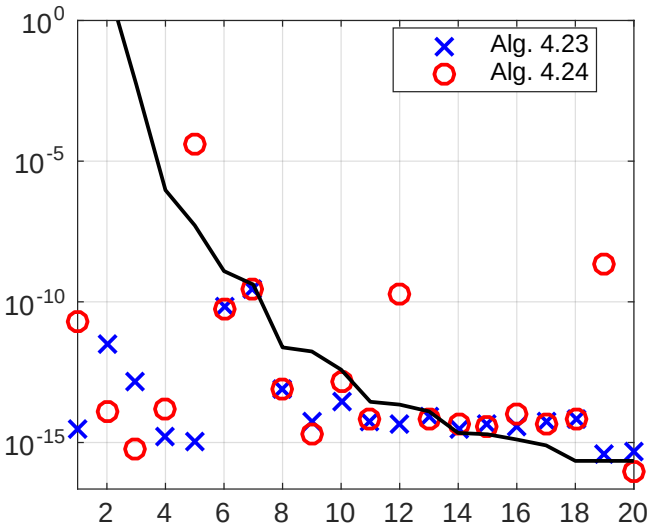


Figure 4.2: Relative error in computing $\operatorname{acos} A$ using Algorithms 4.23 and 4.24. The solid line is $\operatorname{cond}_{\operatorname{acos}}(A)u$.

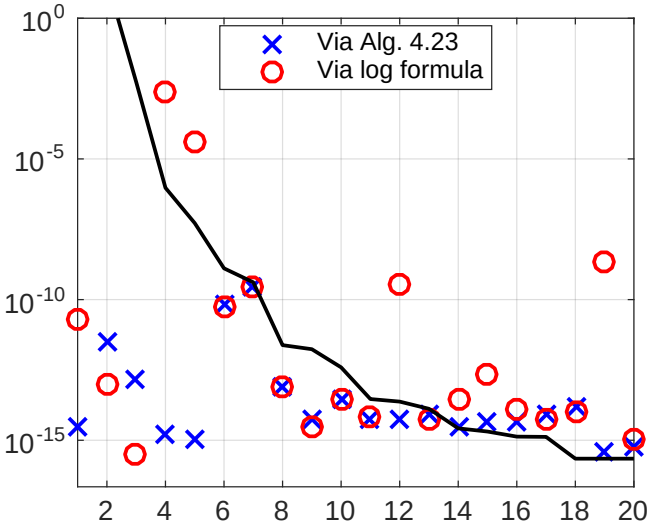


Figure 4.3: Relative error in computing $\operatorname{asin} A$ using Algorithm 4.23 (with (4.29)) and via log formula (variant of Algorithm 4.24). The solid line is $\operatorname{cond}_{\operatorname{asin}}(A)u$.

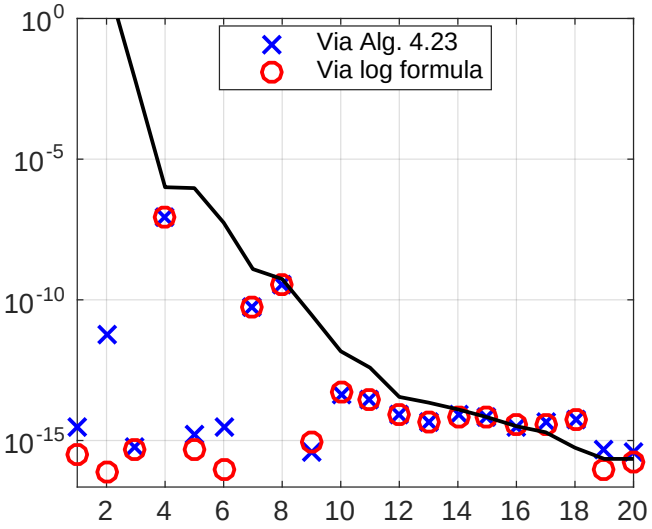


Figure 4.4: Relative error in computing $\operatorname{acosh} A$ using Algorithm 4.23 (with (4.31)) and via log formula (variant of Algorithm 4.24). The solid line is $\operatorname{cond}_{\operatorname{acosh}}(A)u$.

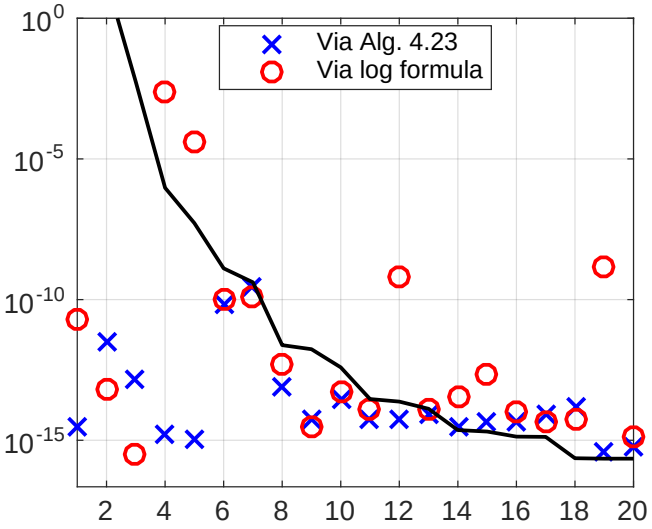


Figure 4.5: Relative error in computing $\operatorname{asinh} A$ using Algorithm 4.23 (with (4.30)) and via log formula (variant of Algorithm 4.24). The solid line is $\operatorname{cond}_{\operatorname{asinh}}(A)u$.

CHAPTER 5

Argument Reduction for Periodic Functions of Matrices

5.1 Introduction

Argument reduction, or range reduction, is a fundamental tool in the construction of algorithms for evaluating functions of real and complex variables. Its goal is to reduce the argument to an interval or region where the function is easier to evaluate. Argument reduction for computing $f(x)$ consists of three main steps [29, Sec. 4.3], [114, Sec. 9.1]:

1. Transform x into $\widetilde{x}$ so that the next step can be performed efficiently.

2. Compute $f(\widetilde{x})$.

3. Reconstruct $f(x)$ from $f(\widetilde{x})$ using appropriate functional identities.

Typically, the identities on which argument reduction is based are addition formulas, such as those for the exponential and related functions.

Our interest here is in argument reduction for matrix functions. Just as in the scalar case we can distinguish between additive reduction and multiplicative reduction. The latter aims to find complex constants c and k such that $f(A)$ is simply related to $f(A/c^k)$ and $f(A/c^k)$ is easier to compute than $f(A)$. For example, for the matrix exponential function, the scaling and squaring method [80, Sec. 10.3] sets $c = 2$ and uses the relation $\mathrm{e}^A = (\mathrm{e}^{A/2^k})^{2^k}$ for positive integers k.

Similarly, double and triple angle formulas underlie recent algorithms for computing the matrix cosine and sine functions [9].

Additive reduction aims to find a matrix $C \in \mathbb{C}^{n\times n}$ and a scalar μ such that $f(A)$ and $f(A-\mu C)$ are simply related and $f(A-\mu C)$ is easier to compute than $f(A)$. Additive reduction is natural for periodic functions, but does not necessarily rely on periodicity. For example, the relation $\mathrm{e}^A = \mathrm{e}^{\mu}\mathrm{e}^{A-\mu I}$ holds for any μ. The latter relation is used on the first step of a recent algorithm of Güttel and Nakatsukasa [74] for computing the matrix exponential, in order to shift the eigenvalues to the left half-plane.

In this work we focus on additive argument reduction for periodic functions. In the scalar case it is straightforward to exploit periodicity. For example, $\sin x = \sin(x - 2k\pi)$ for all integers k and we can choose k so that $x - 2k\pi$ has real part in the interval $(-\pi, \pi]$. In the matrix case we have the complication that replacing A by $A - 2k\pi I$ is not in general effective because each eigenvalue of A may need a different shift.

We introduce the generalized matrix unwinding function $\mathcal{U}_f$ corresponding to a given periodic function f and use it to reduce the matrix argument so that its eigenvalues are lying in a bounded region of the complex plane. When f is the exponential, $\mathcal{U}_f$ is the matrix unwinding function that we introduced in Section 3.3. Here we use the matrix unwinding function to give an additive argument reduction for the matrix exponential and for the matrix sine and cosine. We also derive an algorithm to compute $\mathcal{U}_f$ based on a reordered Schur decomposition and the Parlett recurrence and explain how $\mathcal{U}_f$ can be used in argument reduction for general functions.

A similar additive matrix argument reduction technique was proposed by Ng in his PhD thesis [117]. Ng did not identify the function $\mathcal{U}_f$ and required the reduced eigenvalues to lie on an interval symmetric about the origin. Here we exploit properties of the generalized matrix unwinding function and show that for sine and cosine it relates to the standard matrix unwinding function. Our results allow for additional speedups in the computation; see the end of Section 5.3.2.

This chapter is organized as follows. An argument reduction algorithm for the

exponential is discussed in Section 5.2.1, and algorithms for the sine and cosine are discussed in Section 5.2.2. These algorithms use the matrix unwinding function as a tool for deriving an expression for the reduced argument. In Section 5.3 we discuss more generally argument reduction for computing periodic functions of matrices by introducing a generalized matrix unwinding function. In Section 5.3.1 we briefly consider the norm and conditioning of the generalized matrix unwinding function. The complete argument reduction algorithm is given in Section 5.3.2. Numerical experiments that illustrate the computational savings from applying argument reduction are given in Section 5.4.

5.2 Argument reduction for elementary periodic functions

The computation of some of the most widely used elementary periodic functions of matrices, including the exponential, trigonometric, and hyperbolic functions, can potentially benefit from matrix argument reduction. We discuss in more detail argument reduction algorithms tailored to the exponential, and the sine and cosine functions.

We will make use of the matrix unwinding function, which we defined in (3.5), as a tool to deal with the periodic nature of the exponential, the sine and the cosine functions.

5.2.1 Algorithm for the matrix exponential

A notion of matrix argument reduction for the matrix exponential was introduced by Ng in his PhD thesis [117] and pursued with a particular choice of "modulus" ($2\pi i$) by McCurdy, Ng, and Parlett [110, Sec. 5.3]. In the latter paper, a matrix function mod is defined that is related to the matrix unwinding function by

$$\operatorname{mod}(A) = A - 2\pi i \mathcal{U}(A). \tag{5.1}$$

The motivation for the use of mod was that for $A \in \mathbb{C}^{n\times n}$,

$$\mathrm{e}^A = \mathrm{e}^{A-2\pi i\mathcal{U}(A)}, \tag{5.2}$$

by Lemma 3.12 and, because $\mathrm{mod}(A)$ has eigenvalues with bounded imaginary parts, indeed, in $(-\pi, \pi]$, the computation of $\mathrm{e}^{\mathrm{mod}(A)}$ may be easier than the computation of e^A. In [110] and [117] the authors do not explicitly identify the matrix unwinding function.

Many techniques are available for computing the matrix exponential, as explained in the classic paper by Moler and Van Loan [112], [113]. The MATLAB function `expm` uses the scaling and squaring algorithm of Higham [79], [81]. The algorithm is based on the relation $\mathrm{e}^A = (\mathrm{e}^{2^{-s}A})^{2^s}$ and the use of $[m/m]$ Padé approximants r_m to the exponential; it approximates $\mathrm{e}^{2^{-s}A} \approx r_m(2^{-s}A)$ with a choice of s and m that depends on $\|A\|$. Opting for a larger than necessary value for s may lead to overscaling and possibly inaccurate results [80, Sec. 10.3]. Overscaling can be avoided by using the sequence $\{\|A^k\|^{1/k}\}$ instead of $\|A\|$ when choosing the value of s, as shown by Al-Mohy and Higham [5], who derive an improved scaling and squaring algorithm, which we denote by `expm_new`. The cost of both algorithms is roughly $6 + s$ matrix multiplications and one solution of a multiple right-hand side system.

We combine `expm_new` with the matrix unwinding function in the following algorithm.

Algorithm 5.1. *Given $A \in \mathbb{C}^{n\times n}$, this algorithm computes e^A using the scaling and squaring method in conjunction with the matrix unwinding function.*

1 Compute the Schur decomposition $A = QTQ^*$ (Q unitary, T upper triangular).
2 Compute $U = \mathcal{U}(T)$ using Algorithm 3.26.
3 $T_r = T - 2\pi i U$
4 if $\|T_r\|_F > \|T\|_F$, $T_r = T$; end.
5 Compute $V = e^{T_r}$ using `expm_new` from [5].
6 $X = QVQ^*$.

Cost: $(30\frac{2}{3} + \theta + \frac{s}{3})n^3$, where θn^3 flops is the cost of the reordering in Algorithm 3.26 and s is the scaling parameter used by `expm_new`.

Note that in line 4 we test whether the transformation from T to T_r has increased the norm, and if it has we work with T. The reason is that if T_r exceeds T in norm

then there is likely to be no benefit to the scaling and squaring method from using T_r in place of T and, moreover, this is only likely to happen when $\mathcal{U}(T)$ is very ill conditioned, in which case the computed T_r may be rather inaccurate.

It is instructive to compare the costs of computing e^A from Algorithm 5.1 and directly from `expm_new`. We note that the scaling and squaring algorithm usually sets the degree of the Padé approximation to 13 and the overall differences between applying the algorithm with and without argument reduction are predominantly determined by the scaling factors. For `expm_new` applied directly to A the cost is $(14 + 2s_1)n^3$ flops, and if a Schur decomposition is computed and `expm_new` applied to the triangular factor the cost is $(30\frac{1}{3} + \frac{s_2}{3})n^3$; here, s_1 and s_2 are the respective scaling parameters chosen by `expm_new`. We see from these figures that for large scaling factors it is more efficient to use the Schur decomposition in the e^A computation, in which case denoting the cost of the reordering in Algorithm 3.26 by θn^3 flops, Algorithm 5.1 will be cheaper if its scaling parameter s satisfies $s < s_2 - 1 - 3\theta$. This inequality is unlikely to be satisfied if θ achieves its maximum value of 20, but in the more typical case of $\theta = 1$ (say), the inequality is

$$s < s_2 - 4, \tag{5.3}$$

which is readily satisfied.

5.2.2 Algorithms for the matrix sine and cosine

We consider argument reduction in computing the matrix sine and cosine functions. Analogous algorithms for the hyperbolic sine and cosine can be obtained since for any $A \in \mathbb{C}^{n\times n}$ the trigonometric functions and their hyperbolic counterparts are related by the identities

$$\sinh A = -i \sin(iA),$$
$$\cosh A = \cos(iA).$$

The following lemma gives the main results required for the argument reduction algorithms for the cosine and sine.

Lemma 5.2. *For all* $A \in \mathbb{C}^{n\times n}$,

$$\sin A = \sin(A - 2\pi\,\mathcal{U}(iA)),$$
$$\cos A = \cos(A - 2\pi\,\mathcal{U}(iA)).$$

Proof. Replacing A by iA in (5.2) gives

$$\mathrm{e}^{iA} = \mathrm{e}^{i(A-2\pi\,\mathcal{U}(iA))}.$$

The result then follows by writing the sine and cosine in their exponential forms, $\sin A = (\mathrm{e}^{iA} - \mathrm{e}^{-iA})/(2i)$ and $\cos A = (\mathrm{e}^{iA} + \mathrm{e}^{-iA})/2$ and on noting that $\mathrm{e}^{\pm 2\pi i\mathcal{U}(iA)} = I$ from Lemma 3.12 applied to iA. □

So, we compute sine and cosine at the reduced argument $A_r = A - 2\pi\,\mathcal{U}(iA)$. Observe that we can use the definition of the unwinding function (3.5) to write

$$A_r := A - 2\pi\,\mathcal{U}(iA) = \log \mathrm{e}^{iA}.$$

Recall that the principal matrix logarithm is the one all of whose eigenvalues have imaginary parts in the interval $(-\pi, \pi]$, and so the spectrum of A_r lies in the horizontal strip of the complex plane between $-i\pi$ and $i\pi$.

The algorithm for computing the matrix unwinding function requires an initial Schur decomposition $A = QTQ^* \in \mathbb{C}^{n\times n}$, where Q is unitary and T is upper triangular. Then, since for all functions f, $f(A) = Qf(T)Q^*$, the problem has been reduced to computing $f(T)$, or in our case $f(A) = Qf(T - 2\pi\,\mathcal{U}(iT))Q^*$, where the function f is the sine or the cosine.

We can now give the algorithms for computing sine and cosine using argument reduction. For completeness we state both algorithms separately.

Algorithm 5.3. *Given* $A \in \mathbb{C}^{n\times n}$ *this algorithm computes* $S = \sin A$ *using argument reduction.*

1 Compute a Schur decomposition $A = QTQ^*$ (Q unitary, T upper triangular).
2 Compute $R = \mathcal{U}(iT)$ using Algorithm 3.26.
3 $T_r = T - 2\pi R$
4 If $\|T_r\|_F > \|T\|_F$, $T_r = T$, end
5 $Y = \sin T_r$
6 $S = QYQ^*$

To compute $\sin T_r$ in line 5 of Algorithm 5.3 we use the algorithm of Al-Mohy, Higham and Relton [9, Alg. 5.2] based on the triple angle formula $\sin 3A = 3\sin A - 4\sin^3 A$ along with rational approximants to the sine function. Invoking the triple angle formula s times translates to performing a multiplicative argument reduction so that $\|3^{-s}A\|_F$ is within some pre-computed tolerance level and $\sin(3^{-s}A)$ can be computed using rational approximation. Note that since all eigenvalues of the reduced argument T_r have real parts in the interval $(-\pi, \pi]$, $\rho(T_r) \leq \rho(T)$. However, $\|T_r\|_F$ is not necessarily smaller than $\|T\|_F$, as we noted for the exponential and we discuss further in Section 5.3.2, which is why we have introduced line 4 in Algorithm 5.3. The cost of the general algorithm for sine applied to a triangular matrix T is $((m + 2s + 1)/3)n^3$, where s is the number of times the triple angle formula is invoked and $mn^3/3$ is the number of floating point operations required for the approximation of $\sin(3^{-s}T)$. The cost of applying Algorithm 5.3 to compute $\sin T_r$ is $((m_1+2s_1+2)/3+\theta)n^3$, where m_1 and s_1 are the corresponding parameters for Algorithm 5.3 and θn^3 is the cost of reordering the Schur form required in Algorithm 3.26. Using argument reduction is therefore more efficient than the existing algorithm for computing $\sin T$ if $2s + m > 2s_1 + m_1 + 3\theta + 1$. In the typical case of $\theta = 1$ (say), this inequality is simplified to

$$2s_1 + m_1 < 2s + m - 4. \tag{5.4}$$

In Section 5.4 we give examples for which the inequality (5.4) is satisfied.

Here is the corresponding algorithm for computing cosine using argument reduction.

Algorithm 5.4. *Given $A \in \mathbb{C}^{n\times n}$, this algorithm computes $C = \cos A$ using argument reduction.*

1 Compute a Schur decomposition $A = QTQ^*$ (Q unitary, T upper triangular).
2 Compute $R = \mathcal{U}(iT)$ using Algorithm 3.26.
3 $T_r = T - 2\pi R$
4 If $\|T_r\|_F > \|T\|_F$, $T_r = T$, end
5 $Y = \cos T_r$
6 $C = QYQ^*$

To compute $\cos T_r$ in line 5 of Algorithm 5.4 we use an algorithm of Al-Mohy, Higham and Relton [9, Alg. 4.2] based on the double-angle formula for the cosine and rational approximants.

We let $\widetilde{s}$ be the number of times the scaling formula is invoked and $\widetilde{m}n^3/3$ be the number of floating point operations required for the approximation of cosine at the scaled argument. We find that Algorithm 5.4 is more efficient than the same algorithm without argument reduction if (again assuming a typical cost for the argument reduction algorithm)

$$\widetilde{s}_1 + \widetilde{m}_1 < \widetilde{s} + \widetilde{m} - 4. \tag{5.5}$$

In Section 5.4 we give examples for which the inequality (5.5) is satisfied.

In some applications both the sine and the cosine are required [9]. Since in Algorithms 5.3 and 5.4 the argument is reduced to $A - 2\pi\,\mathcal{U}(iA)$ for both sine and cosine, we can execute the argument reduction step once and then compute sine and cosine at the reduced argument simultaneously, using [9, Alg. 6.2]. For consistency we state the full algorithm below.

Algorithm 5.5. *Given $A \in \mathbb{C}^{n\times n}$, this algorithm computes $C = \cos A$ and $S = \sin A$ using argument reduction.*

1 Compute a Schur decomposition $A = QTQ^*$ (Q unitary, T upper triangular).
2 Compute $R = \mathcal{U}(iT)$ using Algorithm 3.26.
3 $T_r = T - 2\pi R$
4 If $\|T_r\|_F > \|T\|_F$, $T_r = T$, end
5 $Y = \cos T_r$, $Z = \sin T_r$
6 $C = QYQ^*$
7 $S = QZQ^*$

From the operation counts it is easily checked that applying argument reduction in Algorithm 5.5 brings a computational saving if

$$2\hat{s}_1 + \hat{m}_1 < 2\hat{s} + \hat{m} - 4 \tag{5.6}$$

holds. As before, $\hat{s}$ is the number of times a scaling formula is invoked and $\hat{m}n^3/3$ is the number of floating point operations required for the approximation of cosine and sine at the scaled argument.

5.3 Method for general functions

We consider a variant of additive argument reduction that can be used to compute general periodic functions of matrices. Throughout this work we will assume that f satisfies the conditions of the following assumption.

Assumption 5.6. *The function f is analytic and for some period $p \in \mathbb{C}$ satisfies $f^{-1}(f(x_0)) = x_0 + pk$ for every $x \in \mathbb{C}$ and all integers k.*

The inverse function f^{-1} is multivalued: $f^{-1}(f(x_0)) = x_0 + pk$ for all integers k. The conventional way of dealing with multivalued complex functions is to set a principal branch. From this point on f^{-1} will always denote *the principal branch.* We note that there is no agreed convention on the values the inverse function attains on its branch cut, however the chosen values must be used consistently; more detail on this is given in [95].

We note that all our results apply also for doubly periodic functions, such as Jacobian elliptic functions [119, Chap. 22], for example. For functions with multiple periods argument reduction must be applied separately for each period.

For all the examples we consider, there is exactly one finite branch point associated with each branch cut. The values the inverse function attains on its branch cut are set to the values of f^{-1} on the side of the cut that is approached when the finite branch-point is circled counter-clockwise.

For any $x \in \mathbb{C}$ we define the integer-valued function

$$\mathcal{U}_f(x) = \frac{x - f^{-1}(f(x))}{p}.$$

$\mathcal{U}_f$ counts how many periods x is away from the range of f^{-1}.

We will call $\mathcal{U}_f$ the *generalized unwinding number*, because in the particular case $f(x) = \mathrm{e}^x$ it reduces to

$$\mathcal{U}_{\exp}(x) = \frac{x - \log \mathrm{e}^x}{2\pi i}, \tag{5.7}$$

which is the unwinding number we discussed in Section 3.2.

In additive argument reduction we take $\widetilde{x} = x - p\mathcal{U}_f(x)$ and use the fact that $f(\widetilde{x}) = f(f^{-1}(f(x))) = f(x)$, so that the problem reduces to computing $f(\widetilde{x})$. This

is the procedure commonly used for evaluating elementary functions [29, Sec. 4.3], [114, Sec. 9.1].

We note that "reduction" of the argument should not be interpreted literally, as it is possible that $|\widetilde{x}| > |x|$. The relation between $|x|$ and $|\widetilde{x}| = |f^{-1}(f(x))|$ is determined by the way we define the principal branch f^{-1} of the inverse function of f.

To generalize argument reduction to matrix functions we first define the function

$$\mathcal{U}_f(A) = \frac{A - f^{-1}(f(A))}{p}, \quad A \in \mathbb{C}^{n\times n}, \tag{5.8}$$

where $f(A)$ denotes the matrix function defined in terms of the underlying scalar function f. Then $\widetilde{A} = A - p\mathcal{U}_f(A)$ is the argument-reduced matrix that satisfies $f(\widetilde{A}) = f(A)$. We call $\mathcal{U}_f(A)$ the *generalized matrix unwinding function*.

Since f^{-1} is a single branch of the multivalued inverse of f, it is discontinuous at its branch cuts, so we need to provide more information to clarify the meaning of (5.8). Let A have the Jordan canonical form (1.1) and consider the definition of $\mathcal{U}_f(A)$ via (1.2), so that $\mathcal{U}_f(A) = Z\mathcal{U}_f(J)Z^{-1} = Z\,\mathrm{diag}(\mathcal{U}_f(J_k))Z^{-1}$, where

$$\mathcal{U}_f(J_k) := \begin{bmatrix} \mathcal{U}_f(\lambda_k) & \mathcal{U}_f'(\lambda_k) & \dots & \dfrac{\mathcal{U}_f^{(m_k-1)}(\lambda_k)}{(m_k-1)!} \\ & \mathcal{U}_f(\lambda_k) & \ddots & \vdots \\ & & \ddots & \mathcal{U}_f'(\lambda_k) \\ & & & \mathcal{U}_f(\lambda_k) \end{bmatrix}. \tag{5.9}$$

The derivatives $\mathcal{U}_f'(z), \mathcal{U}_f''(z), \dots$ are necessarily zero for values of z where $f(z)$ is not on a branch cut of f^{-1}, since $\mathcal{U}_f$ is locally constant at such values. On the branch cuts we define the derivatives to be zero. Alternatively, we can define the first derivative of f^{-1} at z on a branch cut as the one-sided limit $\mathrm{d}f^{-1}(z)/\mathrm{d}z = \lim_{h\to 0}[f^{-1}(z+h) - f^{-1}(z)]/h$ and so on for higher derivatives, where $z+h$ tends along a continuous path to z from the counter-clockwise direction, consistent with the counter-clockwise continuity principle used to define the value of f^{-1} on its branch cut [95]. These two approaches yield the same $\mathcal{U}_f(A)$ because the underlying scalar functions take the same values on the spectrum of A [80, Sec. 1.2.2]. We used this argument for the definition of the unwinding function in Section 3.3.

We note that $\mathcal{U}_f(A)$ has integer eigenvalues; it is easy to see from (1.3) that $\mathcal{U}_f(J_k(\lambda_k)) = \mathcal{U}_f(\lambda_k)I_{m_k}$ for any Jordan block $J_k(\lambda_k)$. In terms of the Jordan canonical form (1.1) we have

$$\mathcal{U}_f(A) = Z\,\mathrm{diag}(\mathcal{U}_f(\lambda_k)I_{m_k})Z^{-1}. \tag{5.10}$$

If f is the exponential and f^{-1} the principal logarithm, then $\mathcal{U}_{\exp}$ is the matrix unwinding function. Defining $\mathcal{U}_{\sin}$ and $\mathcal{U}_{\cos}$ is less straightforward. The principal branch of the complex arcsine function [119, Table 4.23.1] is defined as the one whose real parts are in the interval $(-\pi/2, \pi/2)$, or whose real parts are $-\pi/2$ and imaginary parts are nonnegative, or whose real parts are $\pi/2$ and imaginary parts are nonpositive. We denote the principal inverse sine by asin. Since we have the identity $\sin x = \sin(\pi - x)$, the principal branch asin is defined by convention on an interval of length only π, despite the period of the sine function being 2π. The function $\mathcal{U}_{\sin}$ is defined under the assumption that $\mathrm{asin}(\sin x) = x - 2\pi k$ for some $k \in \mathbb{Z}$. It is easy to verify that there exist $x \in \mathbb{C}$ such that the above relation does not hold. For example, letting $x = 3\pi/4$, we have $(x - \mathrm{asin}(\sin x))/(2\pi) = 1/4$, which is clearly not an integer. So, the general definition (5.8) is not applicable to sin. This is remedied by defining

$$\mathcal{U}_{\sin}(A) := \mathcal{U}(iA), \tag{5.11}$$

which we used in Algorithm 5.3. Similarly,

$$\mathcal{U}_{\cos}(A) := \mathcal{U}(iA), \tag{5.12}$$

which corresponds to the argument reduction scheme in Algorithm 5.4. For both functions the spectrum of the reduced argument is contained in the horizontal strip of the complex plane between $-i\pi$ and $i\pi$.

Another popular periodic function is the tangent. The principal branch of the inverse tangent function [119, Sec. 4.23(ii)] is defined as the one all of whose eigenvalues have real parts in the interval $(-\pi/2, \pi/2)$, or have real parts $-\pi/2$ and positive imaginary parts, or have real parts $\pi/2$ and negative imaginary parts. The tangent function satisfies $\tan(x + \pi k) = \tan x$ for all $x \in \mathbb{C}$ and $k \in \mathbb{Z}$, so it

has a period π. We have

$$\mathcal{U}_{\tan}(A) = \frac{A - \operatorname{atan}(\tan A)}{\pi} \tag{5.13}$$

and so $\tan A$ can be computed as $\tan(A - \pi\,\mathcal{U}_{\tan}(A))$, provided a reliable algorithm for the tangent exists.

Some applications require the computation of $f(At)$ for many values of $t \in \mathbb{R}$. The next result shows that when t takes integer values we can perform just one argument reduction and then re-use it for each t.

Lemma 5.7. *For $A \in \mathbb{C}^{n\times n}$ and $t \in \mathbb{Z}$, $f(At) = f((A - p\mathcal{U}_f(A))t)$.*

Proof. Since t is an integer, the matrix $t\mathcal{U}_f(A)$ has only integer eigenvalues and therefore $f(At - pt\mathcal{U}_f(A)) = f(At)$. □

We will make use of this result in Examples 8 and 9 of our numerical experiments.

5.3.1 Norm and conditioning of $\mathcal{U}_f$

We give an upper bound on the norm and a lower bound on the condition number of $\mathcal{U}_f(A)$, generalizing the results for the matrix unwinding function in Section 3.3. The results hold for any consistent matrix norm for which $\|\operatorname{diag}(d_i)\| = \max_i |d_i|$. We denote by $\rho(A)$ the spectral radius of A and by $\kappa(A) = \|A\|\|A^{-1}\|$ the condition number with respect to inversion.

Lemma 5.8. *Let $A \in \mathbb{C}^{n\times n}$ have the Jordan canonical form $A = ZJZ^{-1}$ and assume that $|f^{-1}(f(\lambda_k))| \le |\lambda_k|$ for all eigenvalues λ_k of A. Then*

$$\|\mathcal{U}_f(A)\| \le \frac{2\kappa(Z)\rho(A)}{|p|}.$$

Proof. Using (5.10), $\|\mathcal{U}_f(A)\| \le \kappa(Z)\max_k |\mathcal{U}_f(\lambda_k)|$ and

$$\max_k |\mathcal{U}_f(\lambda_k)| = \max_k \frac{|\lambda_k - f^{-1}(f(\lambda_k))|}{|p|} \le \max_k \frac{2|\lambda_k|}{|p|} = \frac{2\rho(A)}{|p|}.$$ □

We note that for the elementary periodic functions with standard choices for the principal branches of the inverse functions the condition $|f^{-1}(f(\lambda_k))| \le |\lambda_k|$

is satisfied. The bound can be made sharper by exploiting properties of specific functions f and f^{-1}, and simplifying further $f^{-1}(f(\lambda_k))$; see Lemma 3.13 in the case of the unwinding function.

We now turn to the conditioning of $\mathcal{U}_f$. When f has an eigenvalue on a branch cut of f^{-1}, where f^{-1} is discontinuous, $\mathrm{cond}_{\mathcal{U}_f}(A) = \infty$. We next give a lower bound on the condition number of $\mathcal{U}_f(A)$.

Lemma 5.9. *For $A \in \mathbb{C}^{n\times n}$ with Jordan canonical form $A = ZJZ^{-1}$,*

$$\mathrm{cond}_{\mathcal{U}_f}(A) \geq \frac{|p|}{2\kappa(Z)} \max_{\lambda,\mu\in\Lambda(A)} \mathcal{U}_f[\lambda,\mu], \tag{5.14}$$

where $\mathcal{U}_f[\lambda,\mu]$ denotes the divided difference

$$\mathcal{U}_f[\lambda,\mu] = \begin{cases} \dfrac{\mathcal{U}_f(\lambda) - \mathcal{U}_f(\mu)}{\lambda - \mu}, & \lambda \neq \mu, \\ \mathcal{U}_f'(\lambda) = 0, & \lambda = \mu. \end{cases}$$

Proof. By Lemma 5.8, $\|\mathcal{U}_f(A)\| \leq 2\kappa(Z)\rho(A)/|p| \leq 2\kappa(Z)\|A\|/|p|$ and then by [80, Thm. 3.14] we have

$$\mathrm{cond}_{\mathcal{U}_f}(A) \geq \frac{\|A\|}{\|\mathcal{U}_f(A)\|} \max_{\lambda,\mu\in\Lambda(A)} \mathcal{U}_f[\lambda,\mu] \geq \frac{|p|}{2\kappa(Z)} \max_{\lambda,\mu\in\Lambda(A)} \mathcal{U}_f[\lambda,\mu]. \qquad \square$$

We note that Lemma 3.14 gives a special case of the above result for the matrix unwinding function.

5.3.2 Algorithm

Having defined $\mathcal{U}_f$ we can state the three-step scheme that carries out argument reduction for a general function f satisfying Assumption 5.6.

1. Compute $\mathcal{U}_f(A)$.
2. Compute the reduced argument $A_r = A - p\mathcal{U}_f(A)$.
3. Compute $f(A_r)$ using an appropriate algorithm.

If the computation of $f(A)$ is carried out by an (inverse) scaling and squaring-type algorithm whose cost depends on some norm-based function of A then we replace step 3 by

3. If $\|A_r\|_F < \|A\|_F$, compute $f(A_r)$ using an appropriate algorithm, else compute $f(A)$.

Recall that the reason for this test is that even though $\rho(A_r) \leq \rho(A)$ for the elementary periodic functions with the standard choice for the principal branches, a decrease in the spectral radius does not imply a decrease in the norm. We used this test in all of our argument reduction algorithms for the elementary functions.

On the first step we need to compute $\mathcal{U}_f(A)$, and it is clearly not suitable to use the definition (5.8) directly. Instead, we first compute a Schur decomposition $A = QTQ^* \in \mathbb{C}^{n\times n}$, where Q is unitary and T is upper triangular. Then, since $\mathcal{U}_f(A) = Q\mathcal{U}_f(T)Q^*$, the problem has been reduced to computing $\mathcal{U}_f(T)$. Similarly to the matrix unwinding function, we compute the generalized matrix unwinding function using the block variant of the Schur–Parlett method [51], implemented in the MATLAB routine `funm`. However, we will use a non-standard reordering and blocking of T, so that all eigenvalues λ_i which yield identical values of $k_i = \mathcal{U}_f(\lambda_i)$ are placed in the same block. We previously used the same reordering in Algorithm 3.26 to compute the matrix unwinding function. The reordering can be achieved using an algorithm of Bai and Demmel [18], implemented in the MATLAB routine `ordschur`. It computes a unitary V such that $\widetilde{T} = V^*TV$ and the eigenvalues of $\widetilde{T}$ appear in the desired order. The diagonal blocks of $\mathcal{U}_f(\widetilde{T})$ are diagonal and given by $\mathcal{U}_f(\widetilde{T}_{ii}) = k_i I$ for all i. The off-diagonal blocks are obtained from the block Parlett recurrence

$$\widetilde{T}_{ii}F_{ij} - F_{ij}\widetilde{T}_{jj} = (k_i - k_j)\widetilde{T}_{ij} + \sum_{\ell=i+1}^{j-1} (F_{i\ell}\widetilde{T}_{\ell j} - \widetilde{T}_{i\ell}F_{\ell j}), \qquad i < j, \tag{5.15}$$

where $F = \mathcal{U}_f(\widetilde{T})$ and the recurrence is derived using $F\widetilde{T} = \widetilde{T}F$. The equations in (5.15) are nonsingular because the diagonal elements of $\widetilde{T}_{ii}$ are distinct from those of $\widetilde{T}_{jj}$.

Algorithm 5.10. *Let T be a triangular matrix from a Schur decomposition $A = QTQ^* \in \mathbb{C}^{n\times n}$ and let f be a periodic function satisfying Assumption 5.6. This algorithm computes $U = \mathcal{U}_f(T)$ using the Schur–Parlett method with a particular reordering.*

1 Assign t_{ii} to set $S_{\mathcal{U}_f(t_{ii})}$, $i = 1\colon n$, and use unitary similarity transformations to reorder T so that all elements belonging to each set $S_{\mathcal{U}_f(t_{ii})}$ are contiguous. Update Q.
2 $u_{ii} = \mathcal{U}_f(t_{ii})$, $i = 1\colon n$
3 for $j = 2\colon n$
4 for $i = j-1\colon -1\colon 1$
5 if $u_{ii} = r_{jj}$
6 $u_{ij} = 0$
7 else
8 $$u_{ij} = \left(t_{ij}(u_{ii} - u_{jj}) + \sum_{\ell=i+1}^{j-1} \left(u_{i\ell} t_{\ell j} - t_{i\ell} u_{\ell j} \right) \right) / \left(t_{ii} - t_{jj} \right)$$
9 end
10 end
11 end

The computational cost of this algorithm is $n^3/3$ flops for U plus the cost of the reordering. This algorithm follows the framework of Algorithm 3.26, where the cost of reordering T was estimated as at most $10n^3$ flops and usually much less.

The complete algorithm for computing a general periodic function of a matrix is as follows.

Algorithm 5.11. *Given $A \in \mathbb{C}^{n\times n}$, and a periodic function f satisfying Assumption 5.6, this algorithm computes $X = f(A)$ using matrix argument reduction.*

1 Compute a Schur decomposition $A = QTQ^*$ (Q unitary, T upper triangular).
2 Compute $U = \mathcal{U}_f(T)$, and update Q, using Algorithm 5.10.
3 $T_r = T - pU$
4 Compute $Y = f(T_r)$ using an appropriate algorithm.
5 $X = QYQ^*$

The computational cost of this algorithm is $25n^3$ for the Schur decomposition plus the cost of computing U and $f(T_r)$.

Algorithm 5.11 is similar to an algorithm of Ng [117] for matrix argument reduction. Ng does not use the function $\mathcal{U}_f$ in (5.8) or make any reference to f^{-1}.

In his algorithm, each eigenvalue λ_i of A is reduced to $\lambda_i - kp$, where the integer k is chosen so that $|\lambda_i - kp|$ is minimized, where we recall that p denotes the period of the function.

5.4 Numerical experiments

We demonstrate numerically the computational savings that can result from using argument reduction for computing the matrix exponential and matrix trigonometric functions. We give five example problems for the matrix exponential in Section 5.4.1 and four example problems for the sine and cosine in Section 5.4.2.

Many applications require the computation of $f(At)$, so in some of our examples we will consider different values of the parameter t. In our examples we use the standard argument reduction and compute $f(At) = f(At - p\mathcal{U}_f(At))$ and we also use Lemma 5.7 to speed up the computation by evaluating $f(At) = f((A - p\mathcal{U}_f(A))t)$ for several integers t.

All numerical experiments were done using MATLAB 2015a, for which the unit roundoff is $u \approx 1.1\times10^{-16}$. Relative errors $\|f(A)-X\|_F/\|f(A)\|_F$, for the computed matrix functions X are measured in the Frobenius norm and for the exact solution $f(A)$ we use a reference result computed at 100-digit precision using the Advanpix Multiprecision Computing Toolbox for MATLAB [3], using the eigendecomposition $A = VDV^{-1}$ and the fact that $f(A) = Vf(D)V^{-1}$.

5.4.1 Matrix exponential

Before describing two particular problems from applications we consider three matrices that demonstrate the reduction in computational cost that can be obtained by using argument reduction for the matrix exponential.

Example 1. The first matrix is a 2×2 matrix with real entries and eigenvalues $1 \pm 500i$:

$$A = \begin{bmatrix} 1 & -500 \\ 500 & 1 \end{bmatrix}, \qquad \mathcal{U}(A) = \begin{bmatrix} 0 & 80i \\ -80i & 0 \end{bmatrix}. \tag{5.16}$$

Matrices like this often appear as diagonal blocks in a quasitriangular Schur form.

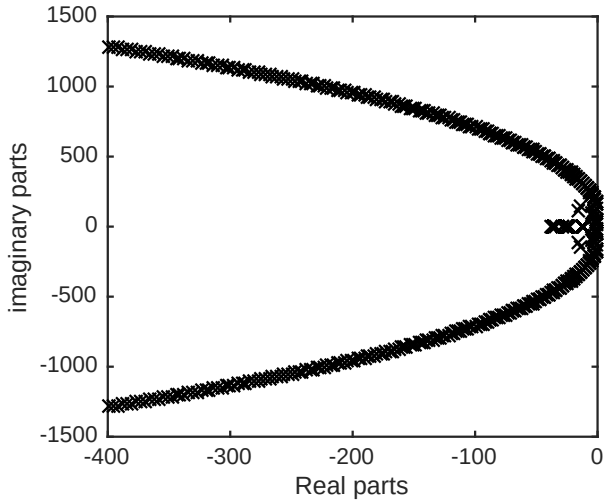


Figure 5.1: Example 2: spectrum of Tolosa matrix of dimension 1090.

Example 2. The second matrix is the Tolosa matrix of dimension 1090 from Matrix Market [109]. It is a sparse matrix arising in the stability analysis of a model of an airplane in flight; it has many eigenvalues with large imaginary part, as shown in Figure 5.1.

Example 3. Our third matrix is the block upper triangular matrix

$$\begin{bmatrix} 0 & 30 & 1 & 1 & 1 & 1 \\ -100 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & -6 & 1 & 1 \\ 0 & 0 & 500 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 200 \\ 0 & 0 & 0 & 0 & -15 & 0 \end{bmatrix} \tag{5.17}$$

from [123, p. 7, Ex I], which has two triple eigenvalues $\pm 10\sqrt{30}i$.

We compute the exponential e^{At} of each of the three matrices for $t = 1$ and $t = 100$, first using the MATLAB function `expm` and `expm_new` from [5], and then by Algorithm 5.1 using `expm` and `expm_new` on line 5. Table 5.1 shows the value of s used in the scaling and squaring method in each case. We see reductions in the value of s for A_r in every case, the amount of reduction varying with A and t, but being as much as 9, and with (5.3) being satisfied in over half of the cases. For `expm_new` the values of s are smaller than for `expm`, since `expm_new` gathers and exploits information about the nonnormality of the matrices, but argument

Table 5.1: Examples 1–3. Scaling parameter s in scaling and squaring method for evaluating e^{At}, with (A_r) and without (A) argument reduction.

	`expm`		`expm_new`	
	A	A_r	A	A_r
Matrix (5.16), $t = 1$	7	0	7	0
Matrix (5.16), $t = 100$	14	5	14	5
Tolosa matrix, $t = 1$	16	10	8	6
Tolosa matrix, $t = 100$	22	14	15	12
Matrix (5.17), $t = 1$	7	3	4	0
Matrix (5.17), $t = 100$	14	9	11	2

reduction still leads to a decrease in s, which is especially notable for (5.16) and (5.17), for which no scaling is needed when $t = 1$. The number of swaps required by the Tolosa matrix is about 250, which yields a value of θ about 0.0084. Recall that θn^3 are the floating point operations required for the swaps in the Schur form.

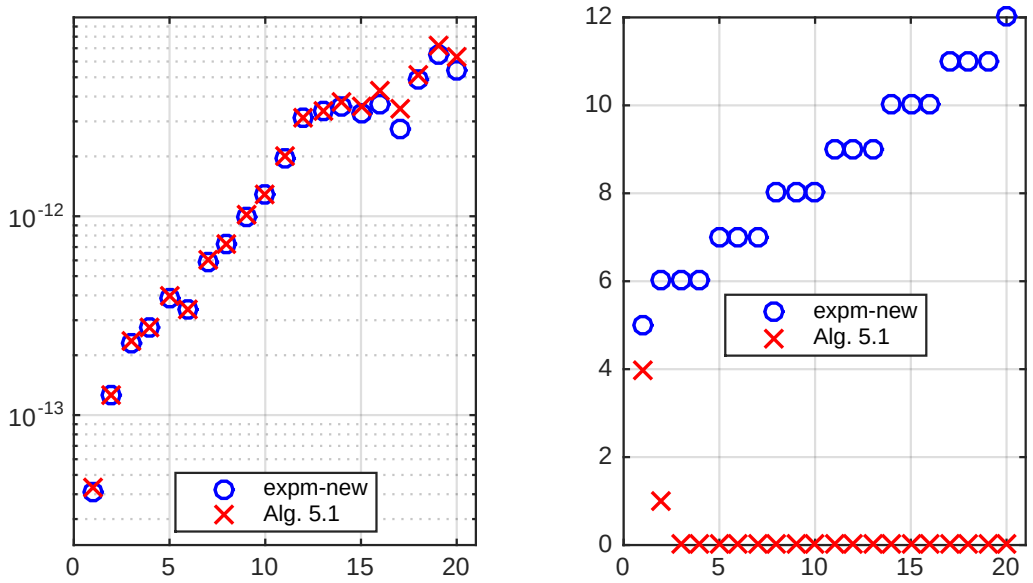
Example 4. Our fourth example is from Physics. Problems in open quantum systems arise from the interaction of a closed quantum system with elements external to it, that is, the environment. The Markovian quantum master equation can be written as

$$\frac{\mathrm{d}}{\mathrm{d}t}\boldsymbol{\rho}(t) = \mathcal{L}\boldsymbol{\rho}(t), \quad \boldsymbol{\rho}(0) = \boldsymbol{\rho_0}. \tag{5.18}$$

Here, $\mathcal{L}$ denotes the Lindbladian super-operator, which is an $n \times n$ skew-Hermitian matrix, perturbed by terms induced from interaction of the system with the environment, for example, from damping. These matrices are characterized by having eigenvalues with large imaginary parts and relatively small real parts [30].

The exact solution to equation (5.18) is $\boldsymbol{\rho}(t) = e^{\mathcal{L}t}\boldsymbol{\rho_0}$, and hence we consider computing the exponential of the Lindbladian. For the example of the quantum damped harmonic oscillator, $\mathcal{L}$ has three nonzero diagonals and bandwidth $2n^{1/2} + 3$. The perturbations induced from damping have a similar sparsity structure, with norms depending on the damping parameters. An explicit form of $\mathcal{L}$ can be found in [68, eqs (3.3), (3.7), (3.12)]. Figure 5.2 shows the relative errors in computing

$\mathrm{e}^{\mathcal{L}}$ using `expm_new` and $\mathrm{e}^{\mathcal{L}_r} = \mathrm{e}^{\mathcal{L}-2\pi i\mathcal{U}(\mathcal{L})}$ using Algorithm 5.1 and a comparison of the scaling parameters s the two approaches require. Twenty 100×100 matrices with different parameters are used; their eigenvalues have real parts of order 1 or less and imaginary parts up to order 10^3. We observe that $\mathrm{e}^{\mathcal{L}_r}$ is computed with very similar levels of accuracy to $\mathrm{e}^{\mathcal{L}}$, but requires a much smaller scaling parameter and in many cases no scaling at all. The replacement of T_r by T in line 4 of Algorithm 5.1 was not carried out for any of these matrices.



(a) (b)

Figure 5.2: Example 4. (a) relative error for using Algorithm 5.1 to compute $\mathrm{e}^{\mathcal{L}}$, and (b) scaling parameters s.

Example 5. We consider the convection–diffusion equation

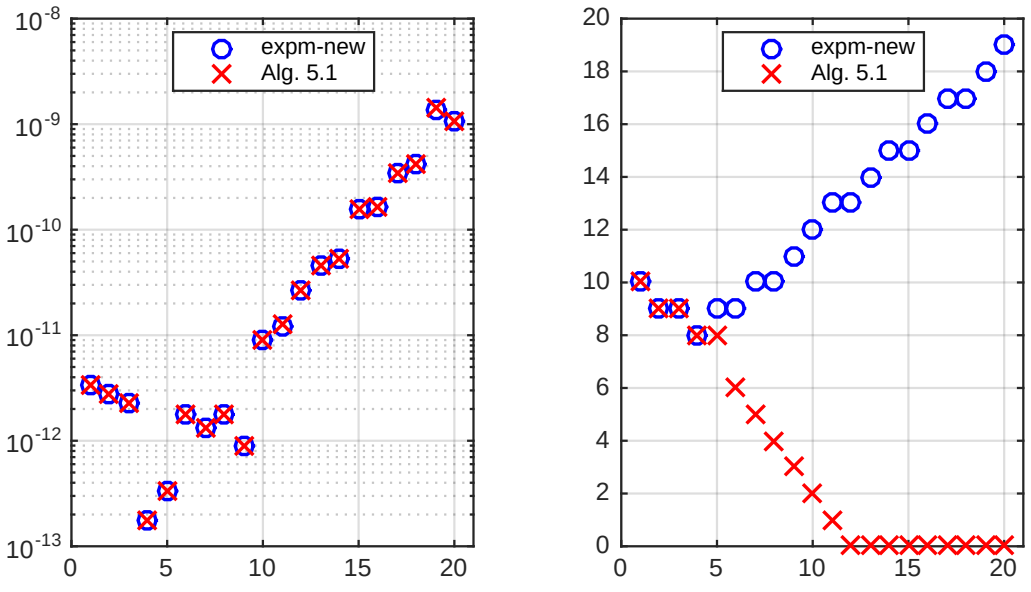
$$u_t + cu_x = du_{xx}, \tag{5.19}$$

where c and $d > 0$ are constants [91, Sec. 3.4]. Assuming homogeneous boundary conditions, spatial discretization using second order central differences yields $\boldsymbol{u}_t = A\boldsymbol{u}$ with a tridiagonal A, so again the solution is given in terms of the matrix exponential.

When the system is dominated by the convection term, i.e., $d \ll |c|$, the matrix A has eigenvalues with small real and large imaginary parts.

We constructed a set of 20 discretization matrices of dimension 100, arranged such that the convection coefficient is increasing and the diffusion coefficient is decreasing: $c = (1.6)^k$ and $d = 0.2(0.5)^k$ for $k = 1\colon 20$. Figure 5.3 shows the relative errors in computing e^A by `expm_new` and e^{A_r} by Algorithm 5.1, and a comparison of the scaling factors the two methods employ. We see that e^{A_r} is computed with the same accuracy as e^A. For the first five matrices the eigenvalues have imaginary parts smaller or not much larger than the real parts and about the same amount of scaling is required to compute e^A and e^{A_r}; thereafter the imaginary parts of the eigenvalues dominate and the use of the matrix unwinding function results in smaller scaling parameters. No scaling is required for matrices indexed 12–20. The replacement of T_r by T in line 4 of Algorithm 5.1 was used for matrices indexed 4 and 5.

Finally, we note that for both of these examples inequality (5.3) is satisfied in most cases (and the underlying quantity θ is less than 1).



(a) (b)

Figure 5.3: Example 5. (a) relative error for using Algorithm 5.1 to compute e^A, and (b) scaling parameters s.

5.4.2 Matrix sine and cosine

We give four example problems—the first one is a 2×2 matrix and aims to illustrate explicitly how argument reduction works, the second example uses a benchmark collection of matrices, and the third and fourth examples arise from wave equations. For all examples we report the relative errors in computing the functions and for Examples 6 and 7 we also report the relative condition numbers $\mathrm{cond}_{\cos} A$ and $\mathrm{cond}_{\sin} A$ multiplied by the unit roundoff. The condition numbers were estimated using the algorithm `funm_condest_fro` from the Matrix Function Toolbox [77], which implements [80, Alg. 3.20]. How to obtain the Fréchet derivatives is explained in Section 1.2.

Example 6. The first matrix we consider is 2×2 and has eigenvalues $500 \pm i$:

$$A = \begin{bmatrix} 500 & -1 \\ 1 & 500 \end{bmatrix}, \quad \mathcal{U}_{\sin}(A) = \mathcal{U}_{\cos}(A) = \begin{bmatrix} 80 & 0 \\ 0 & 80 \end{bmatrix}. \tag{5.20}$$

Matrices with such structure may arise as diagonal blocks in a quasi-triangular Schur form. We consider both $t = 1$ and $t = 100$.

Table 5.2 gives the scaling parameters s and the total number of matrix multiplications `matmults` required for both the approximation stage of Algorithm 5.11 and the algorithms `cosm` for the matrix cosine [9, Alg. 4.2] and `sinm` for the matrix sine [9, Alg. 5.2], with and without argument reduction. Recall that the scaling parameter indicates the number of times a multiplicative reduction formula has been invoked. For both $t = 1$ and $t = 100$ inequalities (5.4) and (5.5) are satisfied (for $t = 1$ reading $7 < 15$ and $8 < 17$, respectively and for $t = 100$ reading $13 < 22$ and $15 < 25$, respectively) and it is more efficient to use argument reduction. Table 5.3 gives the relative errors for `cosm` and `sinm` with and without argument reduction.

The algorithm using argument reduction performs in a forward-stable manner, though yielding errors larger than those without argument reduction. We also note that reducing the argument resulted in a significant reduction in norm. We observed that $\|T\|_F \approx 707$, $\|T - 2\pi\,\mathcal{U}(iT)\|_F \approx 4$, and $\|100T\|_F \approx 70711$, $\|100T - 2\pi\,\mathcal{U}(i100T)\|_F \approx 141$.

Table 5.2: Example 6. Scaling parameter s and number of matrix multiplications required to compute $\cos(At)$ and $\sin(At)$, with (A_r) and without (A) argument reduction.

	s				matmults			
	cosm		sinm		cosm		sinm	
	A	A_r	A	A_r	A	A_r	A	A_r
$t = 1$	7	1	6	2	15	7	17	8
$t = 100$	13	4	10	3	22	13	25	15

Table 5.3: Example 6. Relative errors in the computation of $\cos(At)$ and $\sin(At)$, with (A_r) and without (A) argument reduction.

	cosm			sinm		
	A	A_r	$u\,\mathrm{cond}_{\cos} A$	A	A_r	$u\,\mathrm{cond}_{\sin} A$
$t = 1$	1.51e-16	8.09e-14	4.51e-14	6.61e-14	1.09e-14	5.10e-14
$t = 100$	1.84e-16	3.78e-12	6.43e-12	1.84e-16	3.78e-12	4.27e-12

Example 7. We constructed a set of 30 test matrices, collected from `gallery`, the Matrix Computation Toolbox [76], and test problems provided with EigTool [147]. Most of the test matrices are small, of size 10×10, and we have scaled them by factors ranging from 100 to 10000 to have nonzero generalized unwinding term. Figures 5.4 and 5.5 show the relative errors and the scaling parameters s. The matrices are arranged by decreasing condition numbers $\mathrm{cond}_{\sin} A$ and $\mathrm{cond}_{\cos} A$. We also give the number of matrix multiplications required to form the rational approximation. For one of the matrices, reducing the argument increased its norm, so it was more efficient to compute the functions of the original matrix.

In this example argument reduction brings a reduction in cost (inequalities (5.4) and (5.5) were satisfied for all but four of the matrices), with forward stability at least as good as without argument reduction.

Example 8. Our next example arises from the wave equation [62, Prob. 4]

$$\begin{aligned} &u_{tt} - a(x)u_{xx} + 92u = f(t,x,u), \quad 0 < x < 1, \quad t > 0, \\ &u(t,0) = 0, \quad u(t,1) = 0, \quad u(0,x) = a(x), \quad u_t(0,x) = 0, \end{aligned} \tag{5.21}$$

with $f(t,x,u) = u^5 - a(x) - a(x)^2u^3 + \frac{a(x)^5}{4}\sin^2(20t)\cos(10t)$ and $a(x) = 4x(1-x)$. Performing a second-order symmetric semidiscretization of the spatial variable, we obtain the matrix

$$A = n^2 \begin{bmatrix} 2a(x_1) & -a(x_1) & & & & \\ -a(x_2) & 2a(x_2) & -a(x_2) & & & \\ & \ddots & \ddots & \ddots & & \\ & & -a(x_{n-2}) & 2a(x_{n-2}) & -a(x_{n-2}) & \\ & & & -a(x_{n-1}) & 2a(x_{n-1}) \end{bmatrix} + 92I, \tag{5.22}$$

where $x_i = i/n$, $i = 1 : n-1$ and we have taken $n = 101$. This problem has been identified as difficult for the cosine algorithm [9, Sec. 7.5]. The eigenvalues of A are real and range between 100 and 40492. Figure 5.6 shows the results for using Algorithm 5.4 to compute $\cos At$, for $t = 10, 20, \ldots, 100$. Inequality (5.5) is satisfied for all values of t. Observe that for all values of t, Algorithm 5.4 required no scaling of the argument at all and computing cosine at the original argument required a scaling parameter as high as 20.

Computing $\cos Tt$, with $t = 1, 2, \ldots, 100$, using `cosm` required 5.57 seconds and computing first $T_r = T - 2\pi\mathcal{U}_{\cos}(T)$ and then $\cos(T_r t)$ required 2.20 seconds in total—a speedup by a factor of about 2. The computation was averaged over ten runs, and performed under Mac OS X Yosemite on a machine with an Intel Core i7 2.6 Ghz 4-core processor. These timings do not include the initial Schur decomposition.

Example 9. Our final example is the wave equation

$$u_{tt} = u_{xx} \tag{5.23}$$

with Dirichlet boundary conditions on $[-1, 1]$, $u(t,-1) = u(t,1) = 0$, and initial conditions $u(0,x) =: \boldsymbol{u_0}$ and $u_t(0,x) =: \boldsymbol{v_0}$. After spatial semi-discretization the solution of (5.23) is

$$\boldsymbol{u}(t) = \cos((-A)^{1/2}t)\boldsymbol{u_0} - (-A)^{-1/2}\sin((-A)^{1/2}t)\boldsymbol{v_0}. \tag{5.24}$$

More on spectral discretization matrices can be found in [143]. We take A to be the 20×20 Chebyshev spectral discretization of u_{xx} and compute the sine and

cosine required in the solution (5.24) for $t = 10, 20, \ldots, 100$. For this problem it is most appropriate to use Algorithm 5.5 in conjunction with algorithm `cosmsinm` [9, Alg. 6.2], which computes sine and cosine simultaneously. Figure 5.7 shows the results. Inequality (5.6) is satisfied for all values of t, and hence for this problem it is more economical to perform argument reduction. We note that Algorithm 5.5 required only one scaling for computing sine and cosine at the reduced arguments and as many as 12 at the original arguments. We further speed up the computation by using Lemma 5.7 and reducing the argument only once. The total time taken by `cosmsinm` to compute $\cos Tt$ and $\sin Tt$ for $t = 1, 2, \ldots, 100$ was 0.85 seconds. The time taken to compute $T_r = T - 2\pi\mathcal{U}_{\cos}(T) = T - 2\pi\mathcal{U}_{\sin}(T)$, and $\cos T_r t$ and $\sin T_r t$ for $t = 1 : 1 : 100$ was 0.41 seconds, and so we achieve speedup by a factor of about 2. These timings do not include the initial Schur decomposition, or the square root term required for the solution (5.24) of the wave equation. As before the timings were averaged over ten runs.

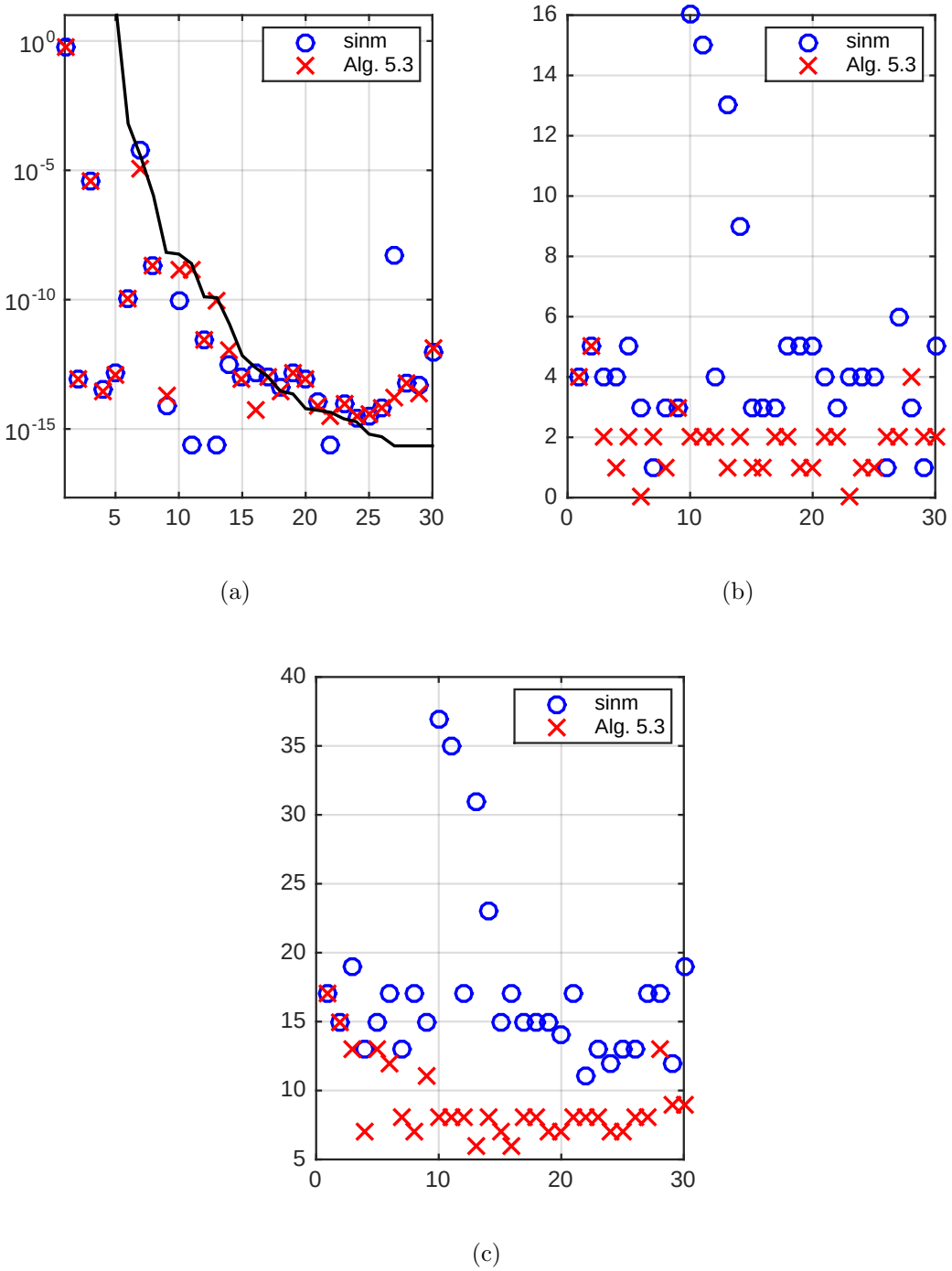


Figure 5.4: Example 7: (a) relative error for using Algorithm 5.3 to compute $\sin A$; the solid line is $u\,\mathrm{cond}_{\sin} A$, (b) scaling parameters s, and (c) total number of matrix multiplications.

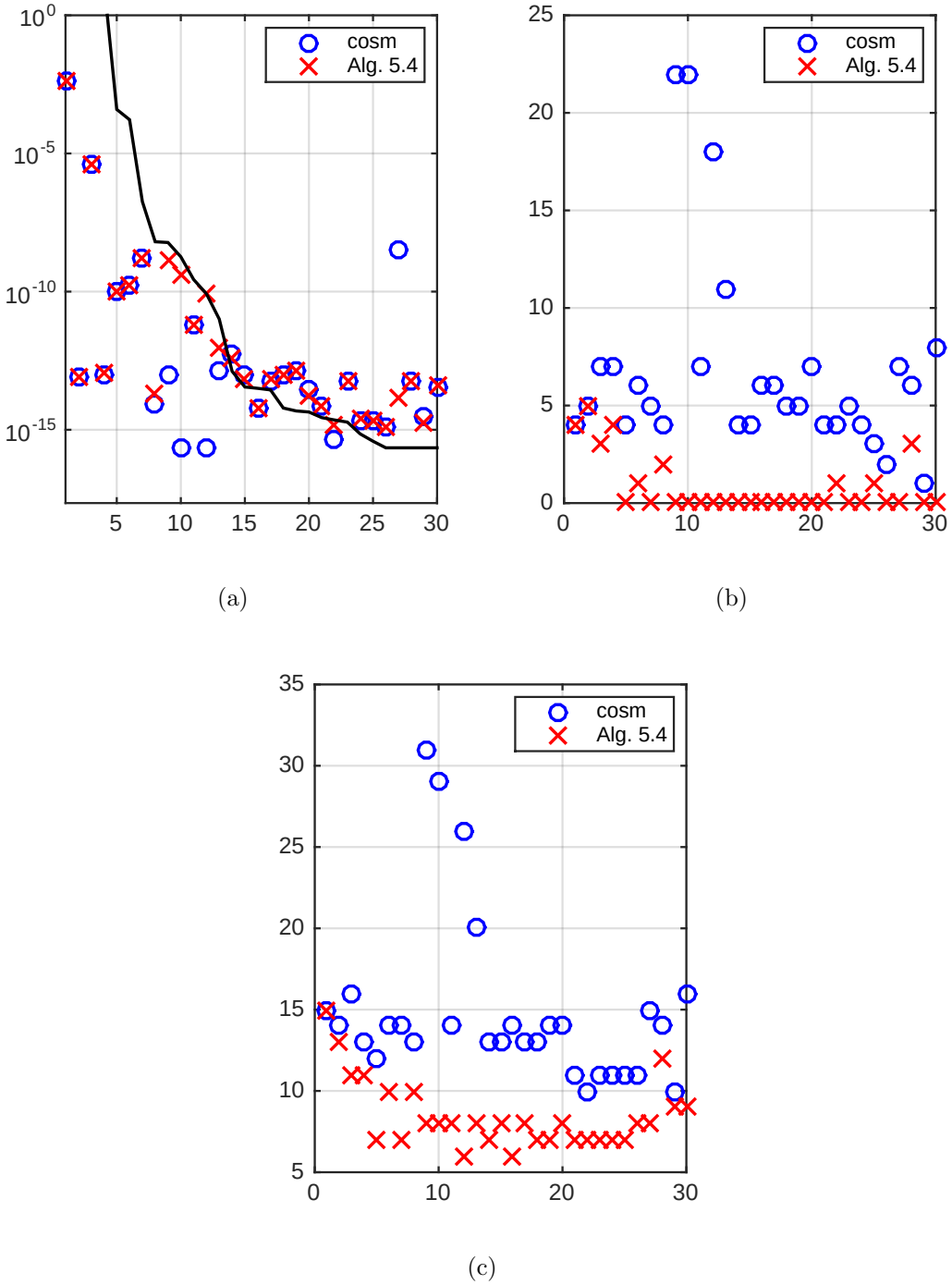
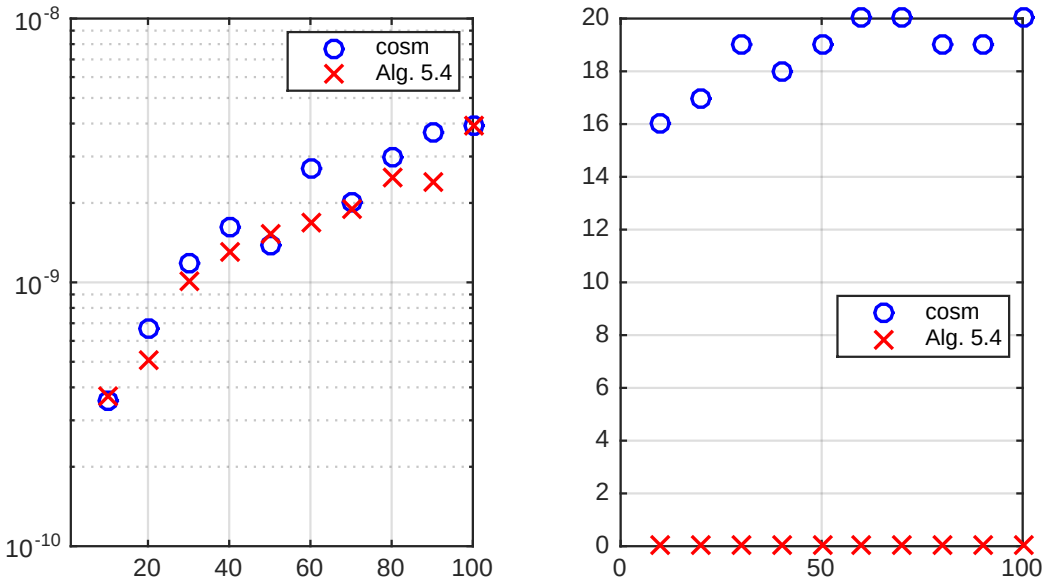
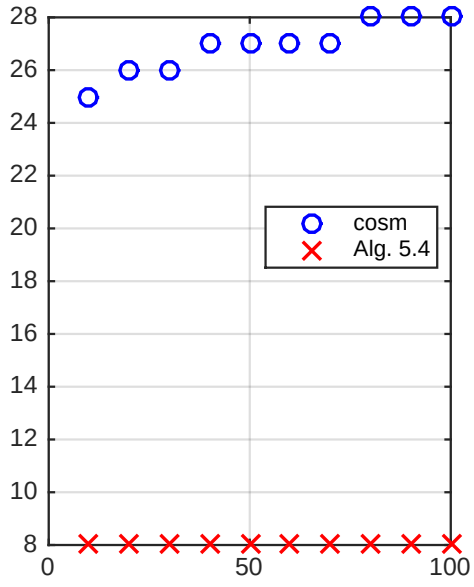


Figure 5.5: Example 7: (a) relative error for using Algorithm 5.4 to compute $\cos A$; the solid line is $u\,\mathrm{cond}_{\cos} A$, (b) scaling parameters s, and (c) total number of matrix multiplications.



(a) (b)



(c)

Figure 5.6: Example 8. (a) Relative error for using Algorithm 5.4 to compute $\cos At$, $t = 10, 20, \ldots, 100$, (b) scaling parameters s, and (c) total number of matrix multiplications.

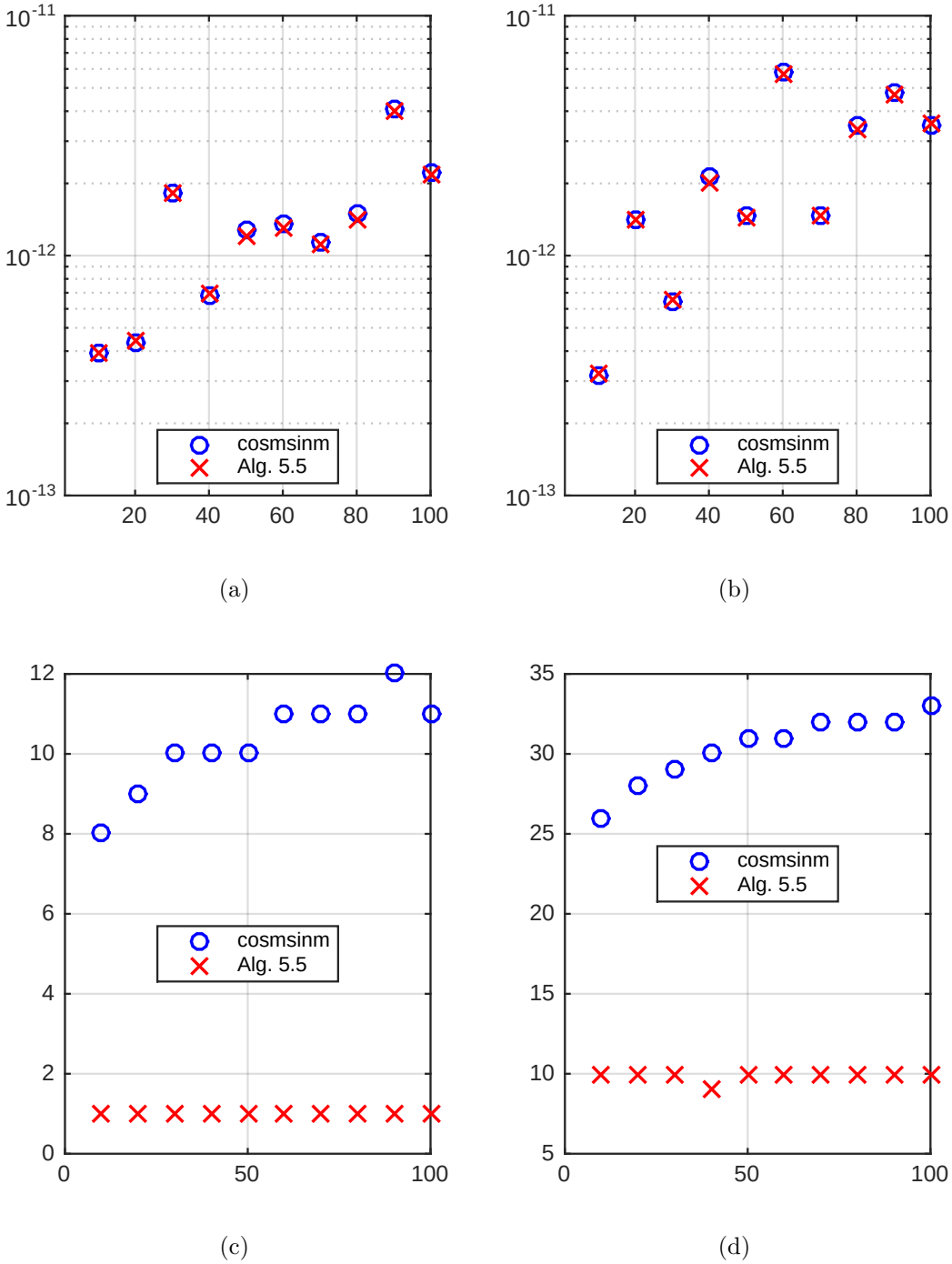


Figure 5.7: Example 9. (a) Relative error for using Algorithm 5.5 to compute $\cos At$, $t = 10, 20, \ldots, 100$, (b) relative error for using Algorithm 5.5 to compute $\sin At$, $t = 10, 20, \ldots, 100$, (c) scaling parameters s, and (d) total number of matrix multiplications required to form both approximations of sine and cosine.

CHAPTER 6

Conclusions

We summarize the material from the previous chapters, give some remarks and identify directions for future research.

In Chapter 2 we considered an application of matrix functions in network analysis. Our aim in this work was to find a value for the damping parameter α such that the centrality scores obtained from the Katz centrality vector $(I - \alpha A)^{-1}\,\mathbf{1}$ and the action of the exponential of the adjacency matrix $\mathrm{e}^{A}\,\mathbf{1}$ are similar. By considering an upper bound on the distance between the two vectors we obtained a value for the Katz parameter $\alpha = (1 - \mathrm{e}^{-\lambda_1})/\lambda_1$ that performs substantially better in our tests than alternatives that have been previously proposed. The new Katz parameter leads to linear systems that are potentially much more ill conditioned than those corresponding to existing choices, but ill conditioning has essentially no effect on the suitability of the computed centralities for ranking. A natural extension of this idea, which is the subject of ongoing research, is to automate the choice of Katz parameter in the case of temporally evolving networks [72].

In Chapter 3 we introduced a new primary matrix function, the matrix unwinding function, and showed that it is an elegant theoretical tool for working with multivalued matrix functions. We derived identities involving the matrix logarithm and the matrix roots and fractional powers.

The scalar unwinding number provides means for the Wright ω function [46] to be defined in terms of the Lambert W function [40], [47]. With recent developments in the theory and algorithms for the matrix Lambert W function [44], [59], in future

it is natural to consider using the matrix unwinding function to define and study the matrix counterpart of the Wright ω function.

In this chapter we also gave an algorithm for computing the matrix unwinding function using a block variant of the Schur–Parlett method with a non-standard reordering of the Schur form.

The goal of Chapter 4 was to study matrix inverse trigonometric and inverse hyperbolic functions and derive algorithms for their computation. We defined the principal branches of the four most common functions asin, acos, asinh, and acosh, paying special attention to the values these functions attain on their respective branch cuts. We showed that many identities known to hold for real scalars can be extended to complex matrices with appropriate use of the matrix unwinding function and the matrix sign function. We also derived some new identities that are not already known in the scalar case. Our new Schur–Padé algorithm performs in a forward stable fashion in our experiments and is superior in accuracy to algorithms based on the logarithm, which have the disadvantage of being susceptible to the sensitivity of the logarithm near the origin. Together with variants of the Schur–Padé algorithm for asin, acosh, and asinh, these are the first numerically reliable algorithms for computing these functions.

The new algorithm for the inverse cosine can be optimized further to perform all computation entirely in real arithmetic. We also remark that using ideas from Chapter 4 it may be possible to derive an algorithm specifically for the inverse hyperbolic cosine function, which would avoid computing the matrix sign function. Similarly to the derivation of the Schur–Padé algorithm, we could scale the argument using (4.18) and then use Padé approximants of a carefully selected function, related to acosh. The study of this algorithm is postponed until demand arises.

In Chapter 5 we introduced argument reduction algorithms for computing the matrix sine and the matrix cosine. We showed that using argument reduction is more economical for some problems. We showed how the matrix unwinding function can be used in conjunction with the inverse scaling and squaring algorithm to compute the matrix exponential using argument reduction. We also showed how it can be used for sine and cosine. But it is not applicable to all periodic functions

and so we introduced the generalized matrix unwinding function and showed how it can be used in argument reduction for periodic functions. Testing the performance of the new argument reduction algorithm in computing periodic functions, other than the exponential, trigonometric and hyperbolic functions, remains the subject of future work. A manuscript on the topic of matrix argument reduction is currently in preparation.

Bibliography

[1] IEEE standard for floating-point arithmetic. *IEEE Std 754-2008*, pages x + 58, 2008.

[2] Milton Abramowitz and Irene A. Stegun, editors. *Handbook of Mathematical Functions with Formulas, Graphs and Mathematical Tables*, volume 55 of *Applied Mathematics Series*. National Bureau of Standards, Washington, D.C., 1964. Reprinted by Dover, New York.

[3] *Multiprecision Computing Toolbox*. Advanpix, Tokyo. `http://www.advanpix.com`.

[4] M. Afanasjew, M. Eiermann, O. G. Ernst, and Stefan Güttel. Implementation of a restarted Krylov subspace method for the evaluation of matrix functions. *Linear Algebra Appl.*, 429:2293–2314, 2008.

[5] Awad H. Al-Mohy and Nicholas J. Higham. A new scaling and squaring algorithm for the matrix exponential. *SIAM J. Matrix Anal. Appl.*, 31(3): 970–989, 2009.

[6] Awad H. Al-Mohy and Nicholas J. Higham. Computing the action of the matrix exponential, with an application to exponential integrators. *SIAM J. Sci. Comput.*, 33(2):488–511, 2011.

[7] Awad H. Al-Mohy and Nicholas J. Higham. Improved inverse scaling and squaring algorithms for the matrix logarithm. *SIAM J. Sci. Comput.*, 34(4): C153–C169, 2012.

[8] Awad H. Al-Mohy, Nicholas J. Higham, and Samuel D. Relton. Computing

the Fréchet derivative of the matrix logarithm and estimating the condition number. *SIAM J. Sci. Comput.*, 35(4):C394–C410, 2013.

[9] Awad H. Al-Mohy, Nicholas J. Higham, and Samuel D. Relton. New algorithms for computing the matrix sine and cosine separately or simultaneously. *SIAM J. Sci. Comput.*, 37(1):A456–A487, 2015.

[10] R. Albert, H. Jeong, and A.-L. Barabási. Internet: Diameter of the world-wide web. *Nature*, 401:130–131, 1999.

[11] Diego R. Amancio, Osvaldo N. Oliveira, Jr., and Luciano da F. Costa. Structure-semantics interplay in complex networks and its effects on the predictability of similarity in texts. *Physica A*, 391:4406–4419, 2012.

[12] Tom M. Apostol. *Mathematical Analysis*. Second edition, Addison-Wesley, Reading, MA, USA, 1974. xvii+492 pp.

[13] Mary Aprahamian, Desmond J. Higham, and Nicholas J. Higham. Matching exponential-based and resolvent-based centrality measures. *Journal of Complex Networks*, 2015. Advance Access published June 29, 2015.

[14] Mary Aprahamian and Nicholas J. Higham. The matrix unwinding function, with an application to computing the matrix exponential. *SIAM J. Matrix Anal. Appl.*, 35(1):88–109, 2014.

[15] Mary Aprahamian and Nicholas J. Higham. Matrix inverse trigonometric and inverse hyperbolic functions: Theory and algorithms. MIMS EPrint 2016.4, Manchester Institute for Mathematical Sciences, The University of Manchester, UK, January 2016. 25 pp.

[16] Helmer Aslaksen. Multiple-valued complex functions and computer algebra. *SIGSAM Bulletin*, 30(2):12–20, 1996.

[17] Zhaojun Bai, James Demmel, Jack Dongarra, Axel Ruhe, and Henk van der Vorst. *Templates for the Solution of Algebraic Eigenvalue Problems: A Practical Guide*. Society for Industrial and Applied Mathematics, 2000.

[18] Zhaojun Bai and James W. Demmel. On swapping diagonal blocks in real Schur form. *Linear Algebra Appl.*, 186:73–95, 1993.

[19] Richard Bellman. *Introduction to Matrix Analysis*. Second edition, McGraw-Hill, New York, 1970. xxiii+403 pp. Reprinted by Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 1997. ISBN 0-89871-399-4.

[20] Michele Benzi and Christine Klymko. Total communicability as a centrality measure. *Journal of Complex Networks*, 1(2):124–149, 2013.

[21] Michele Benzi and Christine Klymko. On the limiting behavior of parameter-dependent network centrality measures. *SIAM J. Matrix Anal. Appl.*, 36(2): 686–706, 2015.

[22] Abraham Berman and Robert J. Plemmons. *Nonnegative Matrices in the Mathematical Sciences*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 1994. xx+340 pp. Corrected republication, with supplement, of work first published in 1979 by Academic Press. ISBN 0-89871-321-8.

[23] Norman Biggs. *Algebraic Graph Theory*. Second edition, Cambridge University Press, 1993.

[24] Åke Björck and Sven Hammarling. A Schur method for the square root of a matrix. *Linear Algebra Appl.*, 52/53:127–140, 1983.

[25] Stephen P. Borgatti and Xun Li. On social network analysis in a supply chain context. *Journal of Supply Chain Management*, 45(2):5–22, 2009.

[26] Russell Bradford. Algebraic simplification of multiple-valued functions. In *Design and Implementation of Symbolic Computation Systems*, John Fitch, editor, volume 721 of *Lecture Notes in Computer Science*, Springer-Verlag, Berlin, 1993, pages 13–21.

[27] Russell Bradford, Robert M. Corless, James H. Davenport, David J. Jeffrey, and Stephen M. Watt. Reasoning about the elementary functions of complex analysis. *Annals of Mathematics and Artificial Intelligence*, 36:303–318, 2002.

[28] Ulrik Brandes and Thomas Erlebach, editors. *Network Analysis: Methodological Foundation*. Springer-Verlag, 2005.

[29] Richard P. Brent and Paul Zimmermann. *Modern Computer Arithmetic*. Cambridge University Press, Cambridge, UK, 2010. ISBN 9780521194693.

[30] Heinz-Peter Breuer and Francesco Petruccione. *The Theory of Open Quantum Systems*. 2002. xxi+625 pp. ISBN 0-19-852063-8.

[31] Nicholas J. Bryan and Gen Wang. Musical influence network analysis and rank of sampled-based music. In *Proceedings of the 12th International Society for Music Information Retrieval Conference (ISMIR)*, 2011, pages 329–334.

[32] A. Buchheim. On the theory of matrices. *Proc. London Math. Soc.*, 16:63–82, 1884.

[33] A. Buchheim. An extension of a theorem of Professor Sylvester's relating to matrices. *Phil. Mag.*, 22(135):173–174, 1886. Fifth series.

[34] Marco Caliari, Peter Kandolf, Alexander Ostermann, and Stefan Rainer. Comparison of software for computing the action of the matrix exponential. *BIT*, 54:113–128, 2014.

[35] João R. Cardoso and F. Silva Leite. The Moser–Veselov equation. *Linear Algebra Appl.*, 360:237–248, 2003.

[36] João R. Cardoso and F. Silva Leite. Computing the inverse matrix hyperbolic sine. In *Numerical Analysis and Its Applications*, Lubin Vulkov, Jerzy Waśniewski, and Plamen Yalamov, editors, volume 1988 of *Lecture Notes in Computer Science*, Springer-Verlag, Berlin, 2001, pages 160–169.

[37] J.R. Cardoso and F. Silva Leite. Computing the inverse matrix hyperbolic sine. In *Numerical Analysis and Its Applications*, Lubin Vulkov, Plamen Yalamov, and Jerzy Waśniewski, editors, volume 1988 of *Lecture Notes in Computer Science*, Springer Berlin Heidelberg, 2001, pages 160–169.

[38] Arthur Cayley. A memoir on the theory of matrices. *Philos. Trans. Roy. Soc. London*, 148:17–37, 1858.

[39] Sheung Hun Cheng, Nicholas J. Higham, Charles S. Kenney, and Alan J. Laub. Approximating the logarithm of a matrix to specified accuracy. *SIAM J. Matrix Anal. Appl.*, 22(4):1112–1125, 2001.

[40] R.M. Corless, G.H. Gonnet, D.E.G. Hare, D.J. Jeffrey, and D.E. Knuth. On the Lambert W function. *Adv. in Comput. Math.*, 5(1):329–359, 1996.

[41] Robert M. Corless. *Essential Maple 7: An Introduction for Scientific Programmers*. Springer-Verlag, New York, 2002. xv+282 pp. ISBN 0-387-95352-3.

[42] Robert M. Corless, James H. Davenport, David J. Jeffrey, Gurjeet Litt, and Stephen M. Watt. *Reasoning about the Elementary Functions of Complex Analysis*, volume 1930 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, 2001. 115-126 pp. ISBN 978-3-540-42071-2.

[43] Robert M. Corless, James H. Davenport, David J. Jeffrey, and Stephen M. Watt. "According to Abramowitz and Stegun" or arccoth needn't be uncouth. *ACM SIGSAM Bulletin*, 34(2):58–65, 2000.

[44] Robert M. Corless, Hui Ding, Nicholas J. Higham, and David J. Jeffrey. The solution of $S\exp(S) = A$ is not always the Lambert W function of A. In *ISSAC '07: Proceedings of the 2007 International Symposium on Symbolic and Algebraic Computation*, New York, 2007, pages 116–121. ACM Press.

[45] Robert M. Corless and David J. Jeffrey. The unwinding number. *ACM SIGSAM Bulletin*, 30(2):28–35, 1996.

[46] Robert M Corless and David J Jeffrey. The Wright ω function. In *Artificial intelligence, automated reasoning, and symbolic computation*, Springer, 2002, pages 76–89.

[47] Robert M. Corless and David J. Jeffrey. The Lambert W function. In *The Princeton Companion to Applied Mathematics*, Nicholas J. Higham, Mark R. Dennis, Paul Glendinning, Paul A. Martin, Fadil Santosa, and Jared Tanner,

editors, Princeton University Press, Princeton, NJ, USA, 2015, pages 151–155.

[48] G. W. Cross and P. Lancaster. Square roots of complex matrices. *Linear and Multilinear Algebra*, 1:289–293, 1974.

[49] Dragoş M. Cvetković, Michael Doob, and Horst Sachs. *Spectra of Graphs—Theory and Applications*. Third edition, Johann Ambrosius Barth Verlag, Heidelberg - Leipzig, 1995.

[50] Michel Daune. *Molecular Biophysics. Structures in Motion*. Oxford University Press, 1999.

[51] Philip I. Davies and Nicholas J. Higham. A Schur–Parlett algorithm for computing matrix functions. *SIAM J. Matrix Anal. Appl.*, 25(2):464–485, 2003.

[52] Timothy A. Davis. *Direct Methods for Sparse Linear Systems*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2006.

[53] L. Dieci, B. Morini, and A. Papini. Computational techniques for real logarithms of matrices. *SIAM J. Matrix Anal. Appl.*, 17(3):570–593, 1996.

[54] E. Estrada, M. Fox, D. J. Higham, and G.-L. Oppo, editors. *Network Science: Complexity in Nature and Technology*. Springer-Verlag, 2010. 245 pp.

[55] Ernesto Estrada. Virtual identification of essential proteins within the protein interaction network of yeast. *Proteomics*, 6(1):35–40, 2006.

[56] Ernesto Estrada. *The Structure of Complex Networks*. Oxford University Press, 2011. 480 pp.

[57] Ernesto Estrada, Naomichi Hatano, and Michele Benzi. The physics of communicability in complex networks. *Physics Reports*, 514 (3):89–119, 2012.

[58] Ernesto Estrada and Juan Alberto Rodrígues-Velázquez. Subgraph centrality in complex networks. *Phys. Rev. E*, 71(5), 056103, 2005.

[59] Massimiliano Fasi, Nicholas J. Higham, and Bruno Iannazzo. An algorithm for the matrix Lambert W function. *SIAM J. Matrix Anal. Appl.*, 36(2): 669–685, 2015.

[60] Leon Festinger. The analysis of sociograms using matrix algebra. *Human Relations*, 2(2):153–158, 1949.

[61] Kurt C. Foster, Stephen Q. Muth, John J. Potterat, and Richard B. Rothenberg. A faster Katz status score algorithm. *Computational & Mathematical Organization Theory*, 7(4):275–285, 2001.

[62] J. M. Franco. New methods for oscillatory systems based on ARKN methods. *Appl. Numer. Math.*, 56(8):1040–1053, 2006.

[63] Linton C. Freeman. A set of measures of centrality based on betweenness. *Sociometry*, 40 (1):35–41, 1977.

[64] Linton C. Freeman. Centrality in social networks conceptual clarification. *Social Networks*, 1:215 – 239, 1978/79.

[65] Linton C. Freeman. *The development of social network analysis*. Empirical Press, Vancouver, 2004.

[66] Andreas Frommer, Stefan Güttel, and Marcel Schweitzer. Efficient and stable Arnoldi restarts for matrix functions based on quadrature. *SIAM J. Matrix Anal. Appl.*, 35:661–683, 2014.

[67] Andreas Frommer, Thomas Lippert, Björn Medeke, and Klaus Schilling, editors. *Numerical Challenges in Lattice Quantum Chromodynamics*, volume 15 of *Lecture Notes in Computational Science and Engineering*. Springer-Verlag, Berlin, 2000. viii+184 pp. ISBN 3-540-67732-1.

[68] Kazuyuki Fujii. Quantum damped harmonic oscillator. In *Advances in Quantum Mechanics*, Paul Bracken, editor, InTech, Rijeka, Croatia, 2013, pages 133–156.

[69] F. R. Gantmacher. *The Theory of Matrices*, volume one. Chelsea, New York, 1959. x+374 pp. ISBN 0-8284-0131-4.

[70] GNU Octave. http://www.octave.org.

[71] S. K. Godunov. *Ordinary Differential Equations with Constant Coefficient*, volume 169 of *Translations of Mathematical Monographs*. American Mathematical Society, Providence, RI, USA, 1997. ix+282 pp. ISBN 0-8218-0656-4.

[72] P. Grindrod, D. J. Higham, M. C. Parsons, and E. Estrada. Communicability across evolving networks. *Physical Review E*, 83:046120, 2011.

[73] Peter Grindrod and Desmond J. Higham. A matrix iteration for dynamic network summaries. *SIAM Rev.*, 55:118–128, 2013.

[74] Stefan Güttel and Yuji Nakatsukasa. Scaled and squared subdiagonal Padé approximation for the matrix exponential. MIMS EPrint 2015.46, Manchester Institute for Mathematical Sciences, The University of Manchester, UK, June 2015. 25 pp.

[75] D. J. Higham, P. Grindrod, A. V. Mantzaris, A. Otley, and P. Laflin. Anticipating activity in social media spikes. In *Proceedings of Modelling and Mining Temporal Interactions, Workshop of the 9th International AAAI Conference on Web and Social Media*, 2015.

[76] Nicholas J. Higham. The Matrix Computation Toolbox. http://www.maths.manchester.ac.uk/~higham/mctoolbox.

[77] Nicholas J. Higham. The Matrix Function Toolbox. http://www.maths.manchester.ac.uk/~higham/mftoolbox.

[78] Nicholas J. Higham. *Accuracy and Stability of Numerical Algorithms*. Second edition, Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2002. xxx+680 pp. ISBN 0-89871-521-0.

[79] Nicholas J. Higham. The scaling and squaring method for the matrix exponential revisited. *SIAM J. Matrix Anal. Appl.*, 26(4):1179–1193, 2005.

[80] Nicholas J. Higham. *Functions of Matrices: Theory and Computation*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2008. xx+425 pp. ISBN 978-0-898716-46-7.

[81] Nicholas J. Higham. The scaling and squaring method for the matrix exponential revisited. *SIAM Rev.*, 51(4):747–764, 2009.

[82] Nicholas J. Higham and Edvin Deadman. A catalogue of software for matrix functions. Version 1.0. MIMS EPrint 2014.8, Manchester Institute for Mathematical Sciences, The University of Manchester, UK, February 2014. 19 pp.

[83] Nicholas J. Higham and Edvin Deadman. A catalogue of software for matrix functions. Version 2.0. MIMS EPrint 2016.3, Manchester Institute for Mathematical Sciences, The University of Manchester, UK, January 2016. 22 pp. Updated March 2016.

[84] Nicholas J. Higham and Lijing Lin. A Schur–Padé algorithm for fractional powers of a matrix. *SIAM J. Matrix Anal. Appl.*, 32(3):1056–1078, 2011.

[85] Nicholas J. Higham and Lijing Lin. An improved Schur–Padé algorithm for fractional powers of a matrix and their Fréchet derivatives. *SIAM J. Matrix Anal. Appl.*, 34(3):1341–1360, 2013.

[86] Nicholas J. Higham, D. Steven Mackey, Niloufer Mackey, and Françoise Tisseur. Functions preserving matrix groups and iterations for the matrix square root. *SIAM J. Matrix Anal. Appl.*, 26(3):849–877, 2005.

[87] Nicholas J. Higham and Françoise Tisseur. A block algorithm for matrix 1-norm estimation, with an application to 1-norm pseudospectra. *SIAM J. Matrix Anal. Appl.*, 21(4):1185–1201, 2000.

[88] Marlis Hochbruck and Christian Lubich. On Krylov subspace approximations to the matrix exponential operator. *SIAM J. Numer. Anal.*, 34(5):1911–1925, 2006.

[89] Roger A. Horn and Charles R. Johnson. *Topics in Matrix Analysis*. Cambridge University Press, Cambridge, UK, 1991. viii+607 pp. ISBN 0-521-30587-X.

[90] Roger A. Horn and Charles R. Johnson. *Matrix Analysis*. Second edition, Cambridge University Press, Cambridge, UK, 2013. xviii+643 pp. ISBN 978-0-521-83940-2.

[91] Willem Hundsdorfer and Jan Verwer. *Numerical Solution of Time-Dependent Advection-Diffusion-Reaction Equations*. Springer-Verlag, Berlin, 2003. x+471 pp. ISBN 3-540-03440-4.

[92] Bruno Iannazzo. *Numerical Solution of Certain Nonlinear Matrix Equations*. PhD thesis, Università degli studi di Pisa, Pisa, Italy, 2007. 180 pp.

[93] Ilse C. F. Ipsen. Computing an eigenvector with inverse iteration. *SIAM Rev.*, 39(2):254–291, 1997.

[94] David J. Jeffrey, D. E. G. Hare, and Robert M. Corless. Unwinding the branches of the Lambert W function. *Math. Scientist*, 21:1–7, 1996.

[95] W. Kahan. Branch cuts for complex elementary functions or much ado about nothing's sign bit. In *The State of the Art in Numerical Analysis*, A. Iserles and M. J. D. Powell, editors, Oxford University Press, 1987, pages 165–211.

[96] Leo Katz. A new status index derived from sociometric data analysis. *Psychometrika*, 18:39–43, 1953.

[97] Maurice G. Kendall. A new measure of rank correlation. *Biometrika*, 30 (1–2):81–93, 1938.

[98] Charles S. Kenney and Alan J. Laub. Condition estimates for matrix functions. *SIAM J. Matrix Anal. Appl.*, 10(2):191–209, 1989.

[99] Charles S. Kenney and Alan J. Laub. The matrix sign function. *IEEE Trans. Automat. Control*, 40(8):1330–1348, 1995.

[100] Jon M. Kleinberg. Authoritative sources in a hyperlinked environment. *J. Assoc. Comput. Mach.*, 46 (5):604–632, 1999.

[101] Edmond Nicolas Laguerre. Le calcul des systèmes linéaires, extrait d'une lettre adressé à M. Hermite. In *Oeuvres de Laguerre*, Ch. Hermite, H. Poincaré,

and E. Rouché, editors, volume 1, Gauthier–Villars, Paris, 1898, pages 221–267. The article is dated 1867 and is "Extrait du Journal de l'École Polytechnique, LXII^{e} Cahier".

[102] Peter Lancaster and Leiba Rodman. *Algebraic Riccati Equations*. Oxford University Press, 1995. xvii+480 pp. ISBN 0-19-853795-6.

[103] Amy N. Langville and Carl D. Meyer. *Google's PageRank and Beyond: The Science of Search Engine Rankings*. Princeton University Press, Princeton, NJ, USA, 2006. x+224 pp. ISBN 0-691-12202-4.

[104] Amy N. Langville and Carl D. Meyer. *Who's #1?: The Science of Rating and Ranking*. Princeton University Press, 2012.

[105] Alan J. Laub. Invariant subspace methods for the numerical solution of Riccati equations. In *The Riccati Equation*, Sergio Bittanti, Alan J. Laub, and Jan C. Willems, editors, Springer-Verlag, Berlin, 1991, pages 163–196.

[106] Jure Leskovec. Stanford network analysis project. `http://snap.stanford.edu`.

[107] Jure Leskovec, Jon Kleinberg, and Christos Faloutsos. Graph evolution: Densification and shrinking diameters. *ACM Transactions on Knowledge Discovery from Data*, 1(1):2:1–2:41, 2007.

[108] Roy Mathias. A chain rule for matrix functions and applications. *SIAM J. Matrix Anal. Appl.*, 17(3):610–620, 1996.

[109] Matrix Market. `http://math.nist.gov/MatrixMarket/`.

[110] A. McCurdy, K. C. Ng, and B. N. Parlett. Accurate computation of divided differences of the exponential function. *Math. Comp.*, 43(168):501–528, 1984.

[111] W. H. Metzler. On the roots of matrices. *Amer. J. Math.*, 14(4):326–377, 1892.

[112] Cleve B. Moler and Charles F. Van Loan. Nineteen dubious ways to compute the exponential of a matrix. *SIAM Rev.*, 20(4):801–836, 1978.

[113] Cleve B. Moler and Charles F. Van Loan. Nineteen dubious ways to compute the exponential of a matrix, twenty-five years later. *SIAM Rev.*, 45(1):3–49, 2003.

[114] Jean-Michel Muller. *Elementary Functions: Algorithms and Implementation*. Second edition, Birkhäuser, Boston, MA, USA, 2006. xxii+265 pp. ISBN 978-0-8176-4372-0.

[115] Yuji Nakatsukasa and Roland Freund. Using Zolotarev's rational approximation for computing the polar, symmetric eigenvalue, and singular value decompositions. *SIAM Rev.* To appear.

[116] M. E. J. Newman. *Networks: An Introduction*. Cambridge University Press, 2010. 784 pp.

[117] Kwok Choi Ng. Contributions to the computation of the matrix exponential. Technical Report PAM-212, Center for Pure and Applied Mathematics, University of California, Berkeley, February 1984. 72 pp. PhD thesis.

[118] Juhani Nieminen. On the centrality in a graph. *Scand. J. Psychol.*, 15:332 – 336, 1974.

[119] Frank W. J. Olver, Daniel W. Lozier, Ronald. F. Boisvert, and Charles W. Clark, editors. *NIST Handbook of Mathematical Functions*. Cambridge University Press, Cambridge, UK, 2010. xv+951 pp. `http://dlmf.nist.gov`. ISBN 978-0-521-14063-8.

[120] L. Page, S. Brin, R. Motwani, and T. Winograd. The PageRank citation ranking: Bringing order to the web. Technical report, Standford Digital Libraries Technology Project, 1998.

[121] Juyong Park and M. E. J. Newman. A network-based ranking system for US college football. (2005) P10014, 2005.

[122] Beresford N. Parlett. *The Symmetric Eigenvalue Problem*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 1998. xxiv+398

pp. Unabridged, amended version of book first published by Prentice-Hall in 1980. ISBN 0-89871-402-8.

[123] Beresford N. Parlett and Kwok Choi Ng. Development of an accurate algorithm for exp(Bt). Technical Report PAM-294, Center for Pure and Applied Mathematics, University of California, Berkeley, August 1985. 23 pp.

[124] Michael S. Paterson and Larry J. Stockmeyer. On the number of nonscalar multiplications necessary to evaluate polynomials. *SIAM J. Comput.*, 2(1): 60–66, 1973.

[125] Charles M. Patton. A representation of branch-cut information. *SIGSAM Bulletin*, 30(2):21–24, 1996.

[126] G. Peano. Intégration par Séries des équations différentielles linéaires. *Math. Annalen*, 32:450–456, 1888.

[127] Karl Pearson. LIII. On lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 2(11):559–572, 1901.

[128] Paul Penfield, Jr. Principal values and branch cuts in complex APL. *SIGAPL APL Quote Quad*, 12(1):248–256, 1981.

[129] G. Peters and J. H. Wilkinson. Inverse iteration, ill-conditioned equations and Newton's method. *SIAM Rev.*, 21(3):339–360, 1979.

[130] George Pólya and Gabor Szegö. *Problems and Theorems in Analysis II. Theory of Functions. Zeros. Polynomials. Determinants. Number Theory. Geometry*. Springer-Verlag, New York, 1998. xi+392 pp. Reprint of the 1976 edition. ISBN 3-540-63686-2.

[131] William H. Press, Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery. *Numerical Recipes: The Art of Scientific Computing*. Third edition, Cambridge University Press, Cambridge, UK, 2007. xxi+1235 pp. ISBN 978-0-521-88068-8.

[132] Juan G. Restrepo, Edward Ott, and Brian R. Hunt. Approximating the largest eigenvalue of network adjacency matrices. *Phys. Rev. E*, 76:056119, 2007.

[133] John R. Rice. A theory of condition. *SIAM J. Numer. Anal.*, 3(2):287–310, 1966.

[134] Klaus Ruedenberg. Free-electron network model for conjugated systems. V. Energies and electron distributions in the FE MO model and in the LCAO MO model. *The Journal of Chemical Physics*, 22(11):1878–1894, 1954.

[135] Steven M. Serbin. Rational approximations of trigonometric matrices with application to second-order systems of differential equations. *Appl. Math. Comput.*, 5(1):75–92, 1979.

[136] L. S. Shieh, Y. T. Tsay, and C. T. Wang. Matrix sector functions and their applications to system theory. *IEE Proc.*, 131(5):171–181, 1984.

[137] Roger B. Sidje. Expokit: a software package for computing matrix exponentials. *ACM Trans. Math. Software*, 24:130–156, 1998.

[138] Charles Spearman. The proof and measurement of association between two things. *Amer. J. Psychol.*, 15(1):72–101, 1904.

[139] J. J. Sylvester. On the equation to the secular inequalities in the planetary theory. *Philosophical Magazine*, 16:267–269, 1883. Reprinted in [140, pp. 110–111].

[140] *The Collected Mathematical Papers of James Joseph Sylvester*, volume IV (1882–1897). Chelsea, New York, 1973. xxxvii+756 pp. ISBN 0-8284-0253-1.

[141] Alan Taylor and Desmond J. Higham. NESSIE: Network example source supporting innovative experimentation. In *Network Science: Complexity in Nature and Technology*, Ernesto Estrada, Maria Fox, Desmond J. Higham, and Gian-Luca Oppo, editors, Springer-Verlag, 2010, pages 85–106.

[142] Sivan Toledo. A high performance algorithm for the matrix sign function, 2015. Talk given at at SIAM Conference on Applied Linear Algebra, Atlanta, USA.

[143] Lloyd N. Trefethen. *Spectral Methods in MATLAB*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2000. xvii + 163 pp. ISBN 978-0-89871-465-4.

[144] J. Keith Vass, Desmond J. Higham, Manikhandan A. V. Mudaliar, Xuerong Mao, and Daniel J. Crowther. Discretization provides a conceptually simple tool to build expression networks. *PLoS ONE*, 6(4):e18634, 2011.

[145] Stanley Wasserman and Katherine Faust. *Social Network Analysis: Methods and Applications*, volume 8. Cambridge University Press, 1994.

[146] Duncan J. Watts. *Small Worlds: The Dynamics of Networks Between Order and Randomness*. Princeton University Press, 1999.

[147] Thomas G. Wright. Eigtool, 2002. `http://www.comlab.ox.ac.uk/pseudospectra/eigtool/`.

[148] W. W. Zachary. An information flow model for conflict and fission in small groups. *J. Anthropol. Res.*, 33:452–473, 1977.

[149] Aidong Zhang. *Protein interaction networks. Computational analysis*. Cambridge University Press, 2009. 278 pp.

[150] Jing Zhao, Ting-Hong Yang, Yongxu Huang, and Petter Holme. Ranking candidate disease genes from gene expression and protein interaction: A Katz-centrality based approach. *PLoS ONE*, 6(9), 2011.