2-2015

On Polynomial Roots Approximation Via Dominant Eigenspaces And Isolation Of Real Roots

Omar Ivan Retamoso Urbano
Graduate Center, City University of New York

ON POLYNOMIAL ROOTS APPROXIMATION VIA DOMINANT EIGENSPACES AND ISOLATION OF REAL ROOTS

by

OMAR IVAN RETAMOSO URBANO

A dissertation submitted to the Graduate Faculty in Mathematics in partial fulfillment of the requirements for the degree of Doctor of Philosophy, The City University of New York.

2015

This manuscript has been read and accepted for the Graduate Faculty in Mathematics in satisfaction of the dissertation requirements for the degree of Doctor of Philosophy.

Victor Y. Pan

Date

Chair of Examining Committee

Linda Keen

Date

Executive Officer

Victor Y. Pan

Christina Sormani

Vladimir Shpilrain

Supervisory Committee

THE CITY UNIVERSITY OF NEW YORK

Abstract

ON POLYNOMIAL ROOTS APPROXIMATION VIA DOMINANT EIGENSPACES AND ISOLATION OF REAL ROOTS

by

OMAR IVAN RETAMOSO URBANO

Advisor: Professor Victor Y. Pan

Finding the roots of a given polynomial is a very old and noble problem in mathematics and computational mathematics. For about 4,000 years, various approaches had been proposed to solve this problem (see [FC99]). In 1824, Niels Abel showed that there existed polynomials of degree five, whose roots could not be expressed using radicals and arithmetic operations through their coefficients. Here is an example of such polynomials:

$$x^5 - 4x - 2.$$

Thus we must resort to iterative methods to approximate the roots of a polynomial given with its coefficients.

There are many algorithms that approximate the roots of a polynomial(see [B40], [B68], [MN93], [MN97], [MN99], [MN02], [MN07]). As important examples we cite Quadtree (Weyl's) Construction and Newton's Iteration (see [P00a]). Some of the algorithms have as their goal to output a single root, for example, the absolutely

largest root. Some other algorithms aim to output a subset of all the roots of the given polynomial, for example, all the roots within a fixed region on the complex plane. In many applications (e.g., algebraic geometric optimization), only the real roots are of interest, and they can be much less numerous than all the roots of the polynomial (see [MP13]). Nevertheless, the best numerical subroutines, such as MPSolve 2.0 [BF00], Eigensolve [F02], and MPsolve 3.0 [BR14], approximate all real roots about as fast and as slow as all complex roots.

The purpose of this thesis is to find real roots of a given polynomial effectively and quickly, this is accomplished by separating real roots from the other roots of the given polynomial and by finding roots which are clustered and absolutely dominant. We use matrix functions throughout this thesis to achieve this goal.

One of the approaches is to approximate the roots of a polynomial $p(x)$ by approximating the eigenvalues of its associated companion matrix C_p. This takes advantage of using the well-known numerical matrix methods for the eigenvalues.

This dissertation is organized as follows.

Chapter 1 is devoted to brief history and modern applications of Polynomial root-finding, definitions, preliminary results, basic theorems, and randomized matrix computations.

In Chapter 2, we present our Basic Algorithms and combine them with repeated squaring to approximate the absolutely largest roots as well as the roots closest to a selected complex point. We recall the matrix sign function and apply it to eigen-solving. We cover its computation and adjust it to real eigen-solving.

In Chapter 3, we present a "matrix free" algorithm to isolate and approximate

real roots of a given polynomial. We use a Cayley map followed by Dandelin's (Lobachevsky's, Gräffe's) iteration. This is in part based on the fact that we have at hand good and efficient algorithms to approximate roots of a polynomial having only real roots (for instance the modified Laguerre's algorithm of [DJLZ97]). The idea is to extract (approximately) from the image of the given polynomial (via compositions of rational functions) a factor whose roots are all real, which can be solved using modified Laguerre's algorithm, so we can output good approximations of the real roots of the given polynomial.

In Chapter 4, we present an algorithm based on a matrix version of the Cayley map used in Chapter 3. As our input, we consider the companion matrix of a given polynomial. The Cayley map and selected rational functions are treated as matrix functions. Via composition of matrix functions we generate and approximate the eigenspace associated with the real eigenvalues of the companion matrix, and then we readily approximate the real eigenvalues of the companion matrix of the given polynomial.

To simplify the algorithm and to avoid numerical issues appearing in computation of the high powers of matrices, we use factorization of $P^k - P^{-k}$ as the product $\prod_{i=0}^{k-1}(P - \omega_k^i P^{-1})$ where $\omega_k = \exp(2\pi\sqrt{-1}/k)$ is a primitive kth root of unity.

Acknowledgements

I would like to thank many people for support and encouragement during the completion of this work. I will start expressing my deepest gratitude to my advisor Professor Victor Pan for all his unconditional help and orientation from the beginning until the end of this work.

Sincerely, I want to thank Professors Christina Sormani and Vladimir Shpilrain for being members of the committee.

Also, I am grateful to my mother Bivina, who as an educator, introduced me to the mathematical world at an early age; my wife Laura, for her unconditional love and support; my sister Lena, for believing in my ideas, and yet questioning them often; my brothers Rafael and Rolando, for moral support; and my mother in law Faina, for her kindness and emotional support.

Finally, I wish to dedicate this work to the memory of my father, who passed away two years ago. Father, your presence will stay always with us.

Contents

Chapter 1

Introduction and preliminaries

1.1 Brief History of Polynomial Root-finding

A polynomial is an expression of the form

$$p(x) \;=\; c_n x^n + c_{n-1} x^{n-1} + ... + c_1 x + c_0 \tag{1.1.1}$$

If the highest power of x is x^n in this equation (1.1.1), then the polynomial is said to have degree n. According to the Fundamental Theorem of Algebra, proved by Argand in 1814, every polynomial has at least one zero (that is, a value ζ that makes $p(\zeta)$ equal to zero), and it follows that a polynomial of degree n has n zeros (not necessarily distinct). Proving this theorem was attempted by D'Alembert 1746, Euler 1749, de Foncenex 1759, Lagrange 1772, Laplace 1795, Wood 1798, and Gauss 1799, but by modern standards all these attempted proofs were not complete because of some serious flaws (see [S81]).

Hereafter we often write x for a real variable, and z for a complex. ζ is a zero of a polynomial $p(x)$ and is a "root" of the equation $p(x) \;=\; 0$ if $p(\zeta) \;=\; 0$. A polynomial $p(x)$ of degree n with any complex "coefficients" c_i has at most n complex roots; they can be nonreal even where the c_i are all real. In this case all the nonreal zeros occur

in conjugate pairs $\alpha + i\beta,\ \alpha - i\beta,\ i = \sqrt{-1}$.

The calculation of roots of polynomials is the oldest mathematical problem. The solution of quadratics was known to the ancient Babylonians (about 2000 B.C.) and to the Arab and Persian scholars of the early Middle Ages, the most famous of them being Al Khwarismi (c.780–c.850) and Omar Khayyam (1048–1131), both Persians. In 1545 G. Cardano published his opus Ars Magna containing solutions of cubic and quartic in closed form; the solutions for cubic have been obtained by his predecessors S. del Ferro and N. Tartaglia and for quartic by his disciple L. Ferrari. In 1824, however, N.H. Abel proved that polynomials of degree five or more could not be solved by a formula involving rational expressions in the coefficients and radicals, as those of degree up to four could be. (P. Ruffini came close to proving this result in 1799.) Since then (and for some time before in fact), researchers have concentrated on numerical (iterative) methods such as the Newton's famous method (see [BM91]) of the 17th century, Bernoulli's method of the 18th (see [WR67]), and Graeffe's method (see [BP96]), proposed by Dandelin in 1828. Of course there have been a plethora of new methods in the 20th and early 21st century, especially since the advent of electronic computers. These include the Jenkins-Traub (see [JT70]), Larkin's (see [N85]) and Muller's methods (see [PTVF07]), as well as several methods for simultaneous approximation starting with the Durand-Kerner method (see [V03]), actually traced back to Weierstrass. Recently matrix methods have become popular.

1.2 Modern Applications of Polynomial Root-finding

Polynomial roots have many applications. For one example, in control theory we are led to the equation

$$y(s) \;=\; G(s)u(s) \tag{1.2.1}$$

where $G(s)$ is known as the "transfer function" of the system, $u(s)$ is the Laplace tranform of the input, and $y(s)$ is that of the output. $G(s)$ usually takes the form $\frac{P(s)}{Q(s)}$ where P and Q are polynomials in s. Their zeros may be needed, or we may require not their exact values, but only the knowledge of whether they lie in the left-half of the complex plane, which indicates stability. Sometimes we need the zeros to be inside the unit circle.

Another application arises in certain financial calculations, for example, to compute the rate of return on an investment where a company buys a machine for, (say) \$100,000. Assume that they rent it out for 12 months at \$5000/month, and for a further 12 months at \$4000/month. It is predicted that the machine will be worth \$25,000 at the end of this period. The solution goes as follows: the present value of \$1 received n months from now is $\frac{1}{(1+i)^n}$, where i is the monthly interest rate, as yet unknown. Hence

$$100,000 \;=\; \sum_{j=1}^{12} \frac{5000}{(1+i)^j} + \sum_{j=13}^{24} \frac{4000}{(1+i)^j} + \frac{25,000}{(1+i)^{24}} \tag{1.2.2}$$

Hence

$$100,000(1+i)^{24} - \sum_{j=1}^{12} 5000(1+i)^{24-j} - \sum_{j=13}^{24} 4000(1+i)^{24-j} - 25,000 \;=\; 0, \tag{1.2.3}$$

a polynomial equation in $1+i$ of degree 24. If the term of the lease was many years, as is often the case, the degree of the polynomial could be in the hundreds.

In signal processing one commonly uses a "linear time-invariant discrete" system. Here an input signal $x[n]$ at the nth time-step produces an output signal $y[n]$ at the same instant of time. The latter signal is related to $x[n]$ and previous input signals, as well as previous output signals, by the equation

$$y[n] \;=\; b_0x[n]+b_1x[n-1]+...+b_Nx[n-N]+a_1y[n-1]+...+a_My[n-M] \quad (1.2.4)$$

To solve this equation one often uses the "z-transform" given by:

$$X(z) \;=\; \sum_{n=-\infty}^{\infty} x[n]z^{-n} \quad (1.2.5)$$

A very useful property of this transform is that the transform of $x[n-i]$ is

$$z^{-i}X(z) \quad (1.2.6)$$

Then if we apply (1.2.5) to (1.2.4) using (1.2.6) we obtain

$$Y(z) \;=\; b_0X(z)+b_1z^{-1}X(z)+...+b_Nz^{-N}X(z)+$$

$$a_1z^{-1}Y(z)+...+a_Mz^{-M}Y(z) \quad (1.2.7)$$

and hence

$$Y(z) \;=\; X(z)\frac{[b_0+b_1z^{-1}+...+b_Nz^{-N}]}{[1-a_1z^{-1}-...-a_Mz^{-M}]} \quad (1.2.8)$$

$$\;=\; X(z)z^{M-N}\frac{[b_0z^N+b_1z^{N-1}+...+b_N]}{[z^M-a_1z^{M-1}-...-a_M]} \quad (1.2.9)$$

For stability we must have $M \;\geq\; N$. We can factorize the denominator polynomial in the above (which is closely linked to computing its zeros z_i). Then we may expand

the right-hand-side of (1.2.9) into partial fractions, and finally apply the inverse z-transform to obtain the components of $y[n]$. For example, the inverse transform of $\frac{z}{z-a}$ is

$$a^n u[n] \tag{1.2.10}$$

where $u[n]$ is the discrete step-function, that is,

$$\begin{array}{rcll} u[n] & = & 0 & (n < 0) \\ & = & 1 & (n \geq 0) \end{array} \tag{1.2.11}$$

In the common case that the denominator of the partial fraction is a quadratic (for the zeros occur in conjugate complex pairs), we find that the inverse transform is a sine or cosine function. For more details, see [EV89].

The last but not the least application worth mentioning is the computations for algebraic geometry and geometric modelling, in particular the computation of the intersections of algebraic curves and surfaces, which amounts to the solution of systems of multivariate polynomial equations. The most popular current methods (such as elimination based on Gröbner basis computation) reduce the solution to accurate root-finding for high degree univariate polynomial equations.

1.3 Preliminaries

We will perform all our computations within the fields of real and complex numbers $\mathbb{R}$ and $\mathbb{C}$, respectively. From now on, "flop" will be the short hand for "arithmetic operation". "Is expected" and "is likely" will mean "with a probability near 1", and "small", "large", "close" and "near" are to be considered in the context.

Throughout this dissertation, we will work with the following univariate polynomial:

$$p(x) = \sum_{i=0}^{n} p_i x^i = p_n \prod_{j=1}^{n} (x - \lambda_j). \tag{1.3.1}$$

It has degree n, real coefficients, r real roots and $2s$ nonreal roots, which are conjugate to each other, $n = r + 2s$.

1.4 Basic Definitions and Theorems

We begin with definitions and basic results.

For $\rho' > \rho > 0$ and a complex c, define the circle $\mathcal{C}_\rho(c) = \{\lambda : \ |\lambda - c| = \rho\}$, the disc $\mathcal{D}_\rho(c) = \{\lambda : \ |\lambda - c| \le \rho\}$, and the annulus $\mathcal{A}_{\rho,\rho'}(c) = \{\lambda : \ \rho \le |\lambda - c| \le \rho'\}$.

$(A_i)_{i=1}^m = (A_1|A_2|...|A_m)$ is the $1 \times m$ block matrix with blocks $A_1, A_2, ..., A_m$ and $\text{diag}(A_i)_{i=1}^m$ =diag$(A_1, A_2, ..., A_m)$ is the $m \times m$ block diagonal matrix with diagonal blocks $A_1, A_2, ..., A_m$.

$I = I_n = (e_1|e_2|...|e_n)$ is the $n \times n$ identity matrix whose columns are the n coordinate vectors $e_1, e_2, ..., e_n$. $J = J_n = (e_n|e_{n-1}|...|e_1)$ is the $n \times n$ reflection matrix, $J^2 = I$. $O_{k,l}$ is the $k \times l$ matrix whose entries are all zeros.

M^T is the transpose of a matrix M.

$\mathcal{R}(M)$ is the range of a matrix M, that is, the linear space generated by its columns.

$\mathcal{N}(M) = \{v : Mv = 0\}$ is the null space of M, rank(M) =dim$(\mathcal{R}(A))$. A matrix of full column rank is a *matrix basis* of its range.

M^+ is the Moore-Penrose pseudo inverse of M [GL96, Section 5.5.4]. An $n \times m$ matrix $X = M^{(I)}$ is a left inverse (respectively right inverse) of an $m \times n$ matrix M

if $XM = I_n$ (respectively, if $MY = I_m$). M^+ is an $M^{(I)}$ for a matrix M of full rank; $M^{(I)} = M^{-1}$ for a nonsingular matrix M.

We use the matrix norms $\|\cdot\|_h$ for $h = 1, 2, \infty$ and write $\|\cdot\| = \|\cdot\|_2$.

We call a matrix U *unitary, orthogonal* and *orthonormal* if $U^T U = I$.

Theorem 1.4.1. *([GL96], Theorem 5.2.2) A matrix M of full column rank has unique QR factorization M=QR, where Q=Q(M) is a unitary matrix and R=R(M) is a square upper triangular matrix with positive diagonal entries.*

1.4.1 Toeplitz Matrices

An $m \times n$ Toeplitz matrix $T = (t_{i-j})_{i,j=1}^{m,n}$ is defined by the $m + n - 1$ entries of its first row and column. In particular, here is an $n \times n$ Toeplitz matrix,

$$T = (t_{i-j})_{i,j=1}^{n,n} = \begin{pmatrix} t_0 & t_{-1} & \ldots & t_{1-n} \\ t_1 & t_0 & \ddots & \vdots \\ \vdots & \ddots & \ddots & t_{-1} \\ t_{n-1} & \cdots & t_1 & t_0 \end{pmatrix}.$$

1.5 Polynomials and Companion Matrices

Given the polynomial of (1.3.1)

$$p(x) = \sum_{i=0}^{n} p_i x^i = p_n \prod_{j=1}^{n} (x - \lambda_j),$$

define

$$C_p = \begin{pmatrix} 0 & & & & -\frac{p_0}{p_n} \\ 1 & \ddots & & & -\frac{p_1}{p_n} \\ & \ddots & \ddots & & \vdots \\ & & \ddots & 0 & -\frac{p_{n-2}}{p_n} \\ & & & 1 & -\frac{p_{n-1}}{p_n} \end{pmatrix}. \tag{1.5.1}$$

C_p is the companion matrix of $p(x)$.

Also we define

$$p_{rev}(x) = x^n p(\frac{1}{x}) = \sum_{i=0}^{n} p_i x^{n-i} = p_n \prod_{j=1}^{n}(1 - x\lambda_j), \tag{1.5.2}$$

the reverse polynomial of $p(x)$.

Note that $C_{p_{rev}} = JC_pJ$.

Without loss of generality for the purpose of approximating the roots of $p(x)$, assume that the polynomial $p(x)$ is monic (i.e., the leading coefficient is 1). Under this assumption we can say that C_p and $C_{p_{rev}}$ are the companion matrices of $p(x) = \det(xI_n - C_p)$ and $p_{rev}(x) = \det(xI_n - C_{p_{rev}})$, respectively.

Theorem 1.5.1. *(See [C96] or [P05]). The companion matrix $C_p \in \mathbb{C}^{n\times n}$ of polynomial p(x) of (1.3.1) generates an algebra $\mathcal{F}$ of matrices with structure of Toeplitz type, called the Frobenius algebra. One needs $O(n)$ flops for addition in $\mathcal{F}$, $O(n \log n)$ flops for multiplication in $\mathcal{F}$, $O(n \log^2 n)$ flops for inversion in $\mathcal{F}$, and $O(n \log n)$ flops for multiplying a matrix from $\mathcal{F}$ by a vector.*

1.6 Eigenspaces and singular value decomposition (SVD)

Definition 1.6.1. *$\mathcal{S}$ is the* invariant subspace *of a square matrix M if $M\mathcal{S} = \{Mv : v \in \mathcal{S}\} \subseteq \mathcal{S}$. A scalar λ is an* eigenvalue *of a matrix M associated with an* eigenvector *$\mathbf{v}$ if $M\mathbf{v} = \lambda\mathbf{v}$ and $\mathbf{v} \neq 0$. All eigenvectors associated with an eigenvalue λ of M form an eigenspace $\mathcal{S}(M, \lambda)$, which is an invariant space. Its dimension d is the* geometric multiplicity *of λ. The eigenvalue is simple if its multiplicity is* 1.

Theorem 1.6.1. *[S01, Theorem 4.1.2]. Let $U \in \mathbb{C}^{n\times r}$ be a matrix basis for an eigenspace $\mathcal{U}$ of a matrix $M \in \mathbb{C}^{n\times n}$. Then the matrix $L = U^{(I)}MU$ is unique (that is, independent of the choice of the left inverse $U^{(I)}$) and satisfies $MU = UL$.*

The above pair $\{L, U\}$ is an *eigenpair* of a matrix M, L is its *eigenblock* and Φ is the *associated eigenspace* of L[S01]. If $L = \lambda I_n$, then $\{\lambda,\mathcal{U}\}$ is also called an *eigenpair* of a matrix M. In this case $\det(\lambda I - M) = 0$ and $\mathcal{N}(M - \lambda I)$ is the eigenspace associated with the *eigenvalue* λ and made up of its *eigenvectors*. $\Lambda(M)$ is the set of all eigenvalues of M, called its *spectrum*. $\rho(M) = max_{\lambda\in\Lambda(M)}\ |\lambda|$ is the *spectral radius* of M. Theorem 1.4.1 implies that $\Lambda(L) \subseteq \Lambda(M)$. For an eigenpair $\{\lambda,$ $\mathcal{U}\}$ let $\psi = min|\frac{\lambda}{\mu}|$ over $\lambda \in \Lambda(L)$ and $\mu \in \Lambda(M) - \Lambda(L)$; we will call the eigenspace $\mathcal{U}$ *dominant* if $\psi > 1$,*dominated* if $\psi < 1$, *strongly dominant* if $\dfrac{1}{\psi} \approx 0$, and *strongly dominated* if $\psi \approx 0$.

A scalar λ is *nearly real* (within $\epsilon > 0$) if $|\Im(\lambda)| \le \epsilon|\lambda|$.

An $n\times n$ matrix M is called *diagonalizable* or *nondefective* if SMS^{-1} is a diagonal matrix for some matrix S, e.g., if M has n distinct eigenvalues. A random real or complex perturbation makes the matrix diagonalizable with probability 1.

In all our algorithms we assume diagonalizable input matrices.

Theorem 1.6.2. *(See [H08, Theorem 1.13].) $\Lambda(F(M)) = F(\Lambda(M))$ for a square matrix M and a function $F(x)$ defined on its spectrum. Furthermore $(F(\lambda),\mathcal{U})$ is an eigenpair of F(M) if M is diagonalizable and has an eigenpairpair $(\lambda, \mathcal{U})$.*

$M = S_M\Sigma_M T_M^T$ is an SVD of an $m \times n$ matrix M of a rank ρ, provided

$S_M S_M^T = S_M^T S_M = I_m$, $T_M T_M^T = T_M^T T_M = I_n$, Σ_M =diag$(\widehat{\Sigma}_M, O_{m-\rho,n-\rho})$, $\widehat{\Sigma}_M$ =diag$(\sigma_j(M))_{j=1}^{\rho}$, $\sigma_j = \sigma_j(M) = \sigma_j(M^T)$ is the jth largest singular value of

a matrix M, and so $S_M \in \mathbb{C}^{m\times m}$, $T_M \in \mathbb{C}^{n\times n}$, $\Sigma_M \in \mathbb{R}^{m\times n}$, and S_M and T_M are real matrices if M is a real matrix. The singular values have the minimax property

$$\sigma_j = \max_{dim(\mathbb{S})=j} \min_{x\in\mathbb{S},\|x\|=1} \|Mx\|,\ j = 1, ..., \rho, \tag{1.6.1}$$

where $\mathbb{S}$ denotes linear spaces [GL96, Theorem 8.6.1]. Note that σ_j^2 is an eigenvalue of $M^TM, \sigma_1 = \|M\|$, $\sigma_\rho = \dfrac{1}{\|M^+\|}$, and $\sigma_j = 0$ for $j > \rho$.

Theorem 1.6.3. *(Cf. [GL96, Corollary 8.6.3].) If A_0 is a submatrix of a matrix A, then $\sigma_j(A) \geq \sigma_j(A_0)$ for all j.*

Let $\sigma_q > \sigma_{q+1}$. Then $q \leq \rho$ and the matrix $T_{q,M} = T(I_q|O_{n-q,q})^T$ generates the right leading singular space $\mathbb{T}_{q,M} = \mathcal{R}(T_{q,M})$ associated with the q largest singular values of the matrix M.

$\kappa(M) = \dfrac{\sigma_1(M)}{\sigma_\rho(M)} = \|M\|\|M^+\| \geq 1$ is the condition number of the matrix M of a rank ρ. Such a matrix is *ill conditioned* if $\sigma_1(M) \gg \sigma_\rho(M)$; otherwise *well conditioned*. $\kappa(M) = \|M\| = \|M^+\| = 1$ for unitary matrices M.

A matrix M has *numerical rank* ρ if the ratio $\dfrac{\sigma_1}{\sigma_\rho}$ is not large but if $\dfrac{\sigma_{\rho+1}}{\sigma_\rho} \ll 1$.

1.7 Ranks and condition numbers of random matrices

1.7.1 Random variables and random matrices

Definition 1.7.1. *$F_\gamma(y) = Probability\{\gamma \leq y\}$ for a real random variable γ is the* cumulative distribution function (cdf) *of γ evaluated at y. $F_{g(\mu,\sigma)}(y) = \frac{1}{\sigma\sqrt{2\pi}} \int_{-\infty}^{y} e^{-\frac{(x-\mu)^2}{2\sigma^2}} dx$ for a Gaussian random variable $g(\mu, \sigma)$ with a mean μ and a possitive variance σ^2, and so*

$$\mu - 4\sigma \leq y \leq \mu + 4\sigma \qquad \text{with a probability near one.} \tag{1.7.1}$$

Definition 1.7.2. *$\mathcal{G}_{\mu,\sigma}^{m\times n}$ is the set of $m \times n$ Gaussian random matrices having a mean μ and a positive variance σ^2, that is, matrices filled with independent Gaussian random variables, all sharing these mean and variance. For $\mu = 0$ and $\sigma^2 = 1$ they are standard Gaussian random matrices. $\mathcal{T}_{\mu,\sigma}^{m\times n}$ is the set $\mathcal{G}_{\mu,\sigma}^{m\times n}$ restricted to Toeplitz matrices.*

Remark 1.7.1. *With probability 1 matrices $G \in \mathcal{G}_{\mu,\sigma}^{m\times n}$ and $T \in \mathcal{T}_{\mu,\sigma}^{m\times n}$ have full rank (cf. [PQa]).*

1.7.2 Condition numbers of random general and Toeplitz matrices

Gaussian random matrices tend to be well conditioned [D88], [E88], [ES05],[CD05], and actually even the sum $W + M$ for any $W \in \mathbb{R}^{m\times n}$ and $M \in \mathcal{G}_{\mu,\sigma}^{m\times n}$ is expected to be well conditioned unless the ratio $\sigma/||W||$ is large or small [SST06].

Next we recall a relevant theorem from [SST06] for $W = O$. The norms of standard Gaussian random matrices M are readily bounded, and we only cover the estimates for the norms $||M^+||$.

Theorem 1.7.1. *Let $l = \min\{m, n\}$, $y \geq 0$, $M \in \mathcal{G}_{\mu,\sigma}^{m\times n}$. Then M has full rank with probability 1 and $F_{1/||M^+||}(y) \leq 2.35\, y\sqrt{l}/\sigma$, that is Probability$\{||M^+|| \geq 2.35x\sqrt{l}/\sigma\} \leq 1/x$.*

A Gaussian random Toeplitz matrix $T_n = (t_{i-j})_{i,j=1}^{n} \in \mathcal{T}_{\mu,\sigma}^{n\times n}$ is nonsingular with probability 1. Next we recall the probabilistic upper bound of [PQ12] on the norm $||T_n^{-1}||$ (cf. empirical data in [PQZa] Table 1], [PQ12]), which are readily extended in [PQ12] to the estimates on the condition number $\kappa(T_n) = ||T_n||\, ||T_n^{-1}||$. The estimates meet a research challenge from [SST06] and show that this number does not tend to

grow exponentially in n as $n \to \infty$, whereas the opposite behavior has been proved in [BG05] for some large and important special classes of Toeplitz matrices.

Theorem 1.7.2. *Given a matrix $T_n = (t_{i-j})_{i,j=1}^n \in \mathcal{T}_{\mu,\sigma}^{n\times n}$, assumed to be nonsingular (cf. Section 1.4.1), write $p_1 = \mathbf{e}_1^T T_n^{-1}\mathbf{e}_1$. Then $F_{1/||p_1 T_n^{-1}||}(y) \le 2n\alpha\beta$ for two random variables α and β such that*

$$F_\alpha(y) \le \sqrt{\frac{2n}{\pi}}\frac{y}{\sigma} \text{ and } F_\beta(y) \le \sqrt{\frac{2n}{\pi}}\frac{y}{\sigma} \text{ for } y \ge 0. \tag{1.7.2}$$

Next, we observe that $p_1 = |\frac{\det T_n}{\det T_{n+1}}|$ and it complements the latter estimate for the norm $||p_1 T_n^{-1}||$ with the following upper bound on the geometric means of the ratios $|\frac{\det T_{h+1}}{\det T_h}|$ for $h = 1, \dots, n-1$.

Theorem 1.7.3. *Let $T_h \ne O$ denote $h \times h$ matrices for $h = 1, \dots, n$ whose entries have absolute values at most t for a fixed scalar or random variable t, e.g. for $t = ||T||$. Furthermore let $T_1 = (t)$. Then the geometric mean $(\prod_{h=1}^{n-1} |\frac{\det T_{h+1}}{\det T_h}|)^{1/(n-1)} = \frac{1}{t}|\det T_n|^{1/(n-1)}$ is at most $n^{\frac{1}{2}(1+\frac{1}{n-1})}t$.*

Proof. The theorem follows from Hadamard's upper bound $|\det M| \le n^{n/2}t^n$, which holds for any $k \times k$ matrix $M = (m_{i,j})_{i,j=1}^k$ with $\max_{i,j=1}^k |m_{i,j}| \le t$. □

The theorem says that the geometric mean of the ratios $|\det T_{h+1}/\det T_h|$ for $h = 1, \dots, n-1$ is not greater than $n^{0.5+\epsilon(k)}t$ where $\epsilon(n) \to 0$ as $k \to \infty$. Furthermore for $T_n \in \mathcal{T}_{\mu,\sigma}^{n\times n}$ we can write $t = ||T||$ and ready bound the cdf of t (cf. (1.4.1)).

By applying Theorem 1.3.3 we can extend the above results to the case of rectangular Toeplitz matrices.

1.7.3 Condition numbers of randomized matrix products

We wish to bound the condition number $\kappa(MG) = ||MG||\ ||(MG)^+||$ of the matrix products of a fixed matrix M and a Gaussian random matrix G. Since $||MG|| \le ||M||\ ||G||$, we probabilistically estimate from below the smallest singular value of the product of fixed and random matrices.

Theorem 1.7.4. *[PQa]. Suppose $M \in \mathcal{G}_{\mu,\sigma}^{m\times n}$, $r(M) = \text{rank}(M) \ge r$, $G \in \mathcal{G}^{r\times m}$. Then the matrix M has full rank r with probability 1 and*

$$F_{1/||(MG)^+||}(y) \le 2.35y\sqrt{r(M)}/(\sigma_{r(M)}(M)\sigma). \tag{1.7.3}$$

The theorem implies that $\sigma_{\text{rank}(MG)} = 1/||(MG)^+|| \le y$ with a probability of at most the order y, and so it is unlikely that multiplication by a square or rectangular Gaussian random matrix can dramatically decrease the smallest positive singular value of a matrix, although $UV = O$ for some pairs of rectangular unitary matrices U and V.

Remark 1.7.2. *The results of the previous subsection bound $\kappa(G)$ for $G \in \mathcal{T}^{n\times r}$. Such a bound is necessary but not sufficient for proving the extension of Theorem 1.4.4 to the case of Toeplitz matrix G.*

Chapter 2

Polynomial Roots Approximation via Dominant Eigenspaces

Suppose we have computed a matrix basis $U \in \mathbb{C}^{n\times r}$ for an invariant space $\mathcal{U}$ of a matrix function $f(M)$ of an $n \times n$ matrix M. By virtue of Theorem 1.3.1, this is a matrix basis of an invariant space of the matrix M. We can first compute a left inverse $U^{(I)}$ or the orthogonalization $Q = Q(U)$ and then approximate the eigenvalues of M associated with this eigenspace as the eigenvalues of the $r \times r$ matrix $L = U^{(I)}MU = Q^H MQ$ (cf. Theorem 1.3.1).

2.1 The Basic Algorithms

The following algorithm employs Theorems 1.3.1 and 1.3.2 to approximate a specified set $\widehat{\Lambda}$ of the eigenvalues of a matrix (e.g., its absolutely largest eigenvalue or the set of its real eigenvalues).

Algorithm 2.1.1. Reduction of the input size in eigen-solving for a subset of the spectrum.

INPUT: *a diagonalizable matrix $M \in \mathbb{R}^{n\times n}$ and a property specifying a subset $\widehat{\Lambda}$ of its unknown spectrum.*

OUTPUT: *a pair of matrices* $\{\widehat{L}, \widehat{U}\}$ *that closely approximates an eigenpair* $\{L, \mathcal{U}\}$ *of* M *such that* $\Lambda(L) = \widehat{\Lambda}$.

COMPUTATIONS:

1. *Compute a matrix function* $F(M)$ *that has strongly dominant eigenspace* $\mathcal{U}$*, shared with* M.
2. *Compute and output a matrix* $\widehat{U}$ *of full column rank whose range approximates the eigenspace* $\mathcal{U}$.
3. *Compute the left inverse* $\widehat{U}^{(I)}$.
4. *Compute and output the matrix* $\widehat{L} = \widehat{U}^{(I)} M \widehat{U}$.

At Stage 2 of the algorithm one can apply rank revealing QR or LU factorization of the matrix $F(M)$ [GC96],[HP92],[CP00].

Given an upper bound r_+ on the dimension r of the eigenspace $\mathcal{U}$, we can alternatively employ a randomized multiplier as follows (cf. [PQY12],[PQZb]).

Algorithm 2.1.2. Randomized approximation of a dominant eigenspace.

INPUT: *a positive integer* r_+ *and a diagonalizable matrix* $F \in \mathbb{R}^{n \times n}$ *that has numerical rank* $n - r$ *and has strongly dominant eigenspace* $\mathcal{U}$ *of dimension* $r > 0$ *for an unknown* $r \le r_+$.

OUTPUT: *an* $n \times r$ *matrix* $\widehat{U}$ *such that* $\mathcal{R}(\widehat{U}) \approx \mathcal{U}$.

COMPUTATIONS:

1. *Compute the* $n \times r_+$ *matrix* FG *for* $G \in \mathcal{G}_{0,1}^{n \times r_+}$.

2. *Compute its rank revealing QR or LU factorization, which outputs its orthogonal matrix basis* $\widehat{U}$.

The following observations support the algorithm. Clearly, $\text{rank}(FG) = n - r$ with probability 1. Define matrix $\tilde{F}$ by zeroing the r smallest singular values of F. We have $\tilde{F} \approx F$ because $\sigma_{n-r+1}(F)$ is small; therefore $\tilde{F}G \approx FG$ and $\mathcal{R}(\tilde{F}) \approx \mathcal{U}$. Deduce from Theorem 1.4.1 that $\mathcal{R}(\tilde{F}G) \approx \mathcal{R}(\tilde{F})$. Finally combine the latter two relationships and obtain that $\mathcal{R}(\tilde{F}G) \approx \mathcal{U}$.

In some cases we naturally arrive at matrices $\tilde{F}(M)$ having some dominated (rather than dominant) eigenspaces $\mathcal{U}$. If the matrix $\tilde{F}(M)$ is nonsingular, then $\mathcal{U}$ is a dominant eigenspace of the matrix $(\tilde{F}(M))^{-1}$, and we can apply Stages 2–4 of Algorithm 2.1.1 to this eigenspace. Alternatively, we can employ the following variation of Algorithm 2.1.1.

Algorithm 2.1.3. Dual reduction of input size in eigen-solving for a subset of the spectrum.

INPUT, OUTPUT *and Stages 3 and 4 of* COMPUTATIONS *as in Algorithm 2.1.1.*

COMPUTATIONS:

1. *Compute a matrix function* $\tilde{F}(M)$ *that has strongly dominated eigenspace approximating* $\mathcal{U}$.
2. *Apply the Inverse Orthogonal Iteration [GL96, page 339] to the matrix* $\tilde{F}(M)$ *to output a matrix* $\widehat{U}$ *of full column rank whose range approximates the eigenspace* $\mathcal{U}$. *Output* $\widehat{L} = \widehat{U}^{(I)} M \widehat{U}$.

Remark 2.1.1. *Seeking a single eigenvalue of M and having performed Stage 1 of Algorithm 2.1.1 (resp. 2.1.3), we can apply the Power (resp. Inverse Power) Method (cf. [GL96, Sections 7.3.1 and 7.6.1]), to approximate an eigenvector $\mathbf{v}$ of the matrix $F(M)$ in its dominant (resp. dominated) eigenspace $\mathcal{U}$. This eigenvector is shared with M by virtue of Theorem 1.3.2, and we can approximate the associated eigenvalue of M by the Rayleigh quotient $\mathbf{v}^T M\mathbf{v}/\mathbf{v}^T\mathbf{v}$ or a simple quotient $\mathbf{v}^T M\mathbf{e}_j/\mathbf{v}^T\mathbf{e}_j$ for a fixed or random integer j, $1 \le j \le n$, in [BGP04], [PQZC11] and [PZ10/11]. We can employ deflation to approximate other eigenvalues of M.*

In the next sections we describe some algorithms for computing matrix functions $F(M)$ and $\tilde{F}(M)$ at stages 1 of Algorithms 2.1.1 and 2.1.3.

2.2 Repeated squaring

Assume A is a diagonalizable matrix and let $F(A) = A^k$ be a matrix function of A then based on Theorem 1.3.2 for sufficiently large integers k, the matrices A^k have dominant eigenspace $\mathcal{U}$ associated with the set of the absolutely largest eigenvalues of A. For a fixed or random real or complex shift s we can write $A_0 = A - sI$ and compute $A_0^{2^h}$ in h squarings,

$$A_{h+1} = a_h A_h^2 \qquad where \qquad a_h \approx \frac{1}{\|A_h^2\|} \qquad for \qquad h = 0, 1, ... \tag{2.2.1}$$

It is well-known that squaring a matrix squares its eigenvalues, and so for large values of h the absolutely largest eigenvalues of A_h strongly dominate its other eigenvalues.

Suppose A is a real diagonalizable matrix with simple eigenvalues and h is a reasonably large number. Then with probability one the dominant eigenspace $\mathcal{U}$ of

A_h has dimension one for random nonreal shifts s and dimension one or two for a random real number s. If the matrix A has the single absolutely largest eigenvalue of multiplicity m or has a cluster of m simple absolutely largest eigenvalues, then the associated eigenspace of dimension m is dominant for the matrix A_h and a reasonably large number h. In this cases the dimension of the dominant eigenspace of the matrix A_h is equal to the numerical rank of this matrix.

For $A = C_p$ we can follow [C96] and apply the FFT-based algorithms that support Fact 1.2.1 to perform every squaring and every multiplication in $O(n \log n)$ flops. The bottleneck is the recovery of the roots of $p(x)$ at the end of the squaring process where $|\lambda_j| \approx |\lambda_k|$ for $j \neq k$. The paper [P05] takes care of some difficulties by using approximations to the roots of $p'(x), p''(x)$, etc., but the techniques of [P05] are still too close to the symbolic recovery methods of [C96]. In contrast Algorithm 2.1.1 reduces the computations of the r eigenvalues of a selected subset of the spectrum $\Lambda(A)$ to eigen-solving for the $r \times r$ matrix L, which is simple for a small integer r.

Now replace A_0 in (2.1.1) by $A_0 = (A - \sigma I)^{-1}$ for a fixed complex number σ. Then the dominant eigenspace of A_h for large h is associated with the set of the eigenvalues of A that are the nearest to σ, e.g., the absolutely smallest eigenvalues where $\sigma = 0$. For $A = C_p$ we can alternatively write $A_0 = C_{p_{rev}(x-\sigma)}$ in (2.1.1).

2.3 Matrix sign function and dominant eigenspaces

Now we will consider a useful function for generating dominant eigenspaces. It is called the Sign function. At first we define it for complex numbers. Then we extend it to matrices.

Definition 2.3.1. *Given two real numbers a,b where* $a \neq 0$*, write*

$$\text{sign}(a + b\sqrt{-1}) = \begin{cases} 1 & , a > 0 \\ -1 & , a < 0 \end{cases}$$

Definition 2.3.2. *(See [H08].) Let* $A = ZJZ^{-1}$ *be a Jordan canonical decomposition of an* $n \times n$ *matrix* A *where* $J = \text{diag}(J_-, J_+)$*,* J_- *is a* $p \times p$ *matrix, and all its* p *diagonal entries have negative real parts, whereas* J_+ *is a* $q \times q$ *matrix, and all its* q *diagonal entries have positive real parts. Then* $\text{sign}(A) = Z\text{diag}(-I_p, I_q)Z^{-1}$*. Equivalently* $\text{sign}(A) = A(A^2)^{-1/2}$ *or* $\text{sign}(A) = \frac{2}{\pi}A\int_0^\infty (t^2 I_n + A^2)^{-1}dt$*.*

Definition 2.3.3. *Assume the matrices* $A = ZJZ^{-1}$*,* J_- *and* J_+ *above, such that* $n = p + q + r$ *and* $J = \text{diag}(J_-, J_0, J_+)$ *for a* $r \times r$ *matrix* J_0 *whose all* r *diagonal entries have real parts 0. Then fix some* $r \times r$ *real diagonal matrix* Dr*, e.g.,* $D_r = O_{r,r}$*, and define a generalized matrix sign function* $\text{sign}(A)$ *by writing* $\text{sign}(A) = Z\text{diag}(-I_p, D_r\sqrt{-1}, I_q)Z^{-1}$*.*

We have the following simple results.

Theorem 2.3.1. *Let us consider the generalized matrix sign function* $\text{sign}(A)$ *defined for an* $n \times n$ *matrix* $A = ZJZ^{-1}$*. Then for some real* $r \times r$ *diagonal matrix* D_r *we have*

$$I_n - \text{sign}(A) = Z\text{diag}(2I_p, I_r - D_r\sqrt{-1}, O_{q,q})Z^{-1},$$

$$I_n + \text{sign}(A) = Z\text{diag}(O_{p,p}, I_r + D_r\sqrt{-1}, 2I_q)Z^{-1},$$

$$I_n - \text{sign}(A)^2 = Z\text{diag}(O_{p,p}, I_r + D_r^2, O_{q,q})Z^{-1}.$$

Corollary 2.3.1. *Under the assumptions of Theorem 2.3.1 the matrix* $I_n - \text{sign}(A)^2$ *has dominant eigenspace of dimension* r *associated with the eigenvalues of the matrix*

A that lie on the imaginary axis

$$\mathcal{IA} = \{\lambda : \Re(\lambda) = 0\}. \tag{2.3.1}$$

The matrices $I_n - \text{sign}(A)$ *(respectively* $I_n + \text{sign}(A)$*) have dominant eigenspaces associated with the eigenvalues of A that either lie on the left (respectively, on the right) of the axis* $\mathcal{IA}$ *or lie on this axis and have nonzero images in* $I_n - \text{sign}(A)$ *(respectively,* $I_n + \text{sign}(A)$*).*

2.4 Eigen-solving via matrix sign computation

If we have at hand the matrices A and $F(A) = I_n - \text{sign}(A)^2$ available, we can apply Algorithm 2.1.1 to approximate the imaginary eigenvalues of A. In the next sections we devise real eigensolvers for a real $n \times n$ matrix M, based on applying these techniques to the matrix $A = M\sqrt{-1}$. Likewise, having the matrices A and $F(A) = I_n - \text{sign}(A)$ (respectively, $F(A) = I_n + \text{sign}(A)$) available, we can apply Algorithm 2.1.1 to approximate all eigenvalues of A that lie either on the axis $\mathcal{IA}$ or on the left (respectively, right) from it. The computed square matrices L have dimensions p_+ and q_+, respectively, where $p \le p_+ \le p + r$ and $q \le q_+ \le q + r$. If $M = C_p$ and if the integer p_+ or q_+ is large, we split out a high degree factor of the polynomial $p(x)$. This can lead to dramatic growth of the coefficients, for example, in the case where we split the polynomial $x^n + 1$ into the product of two high degree factors, such that all roots of one of them have positive real parts. The subdivision techniques (cf. [P00b]) based on the following simple fact, however, lead us to a universal remedy, unlike the limited remedies in the paper [C96].

Theorem 2.4.1. *Suppose* $\mathcal{U}$ *and* $\mathcal{V}$ *are two eigenspaces of A and* $\Lambda(\mathcal{U})$ *and* $\Lambda(\mathcal{V})$ *are*

the sets of the associated eigenvalues. Then $\Lambda(\mathcal{U}) \cap \Lambda(\mathcal{V})$ *is the set of the eigenvalues of* A *associated with the eigenspace* $\mathcal{U} \cap \mathcal{V}$.

By computing the matrix sign function of the matrices $\alpha A - \sigma I$ for various selected pairs of complex scalars α and σ, we can define the eigenspace of A associated with the eigenvalues lying in a selected region of the complex plane bounded by straight lines, e.g., in any fixed rectangle with four vertices defined by the pairs $\{\alpha, \sigma\}$ where α equals 1 and $\sqrt{-1}$ and where $\sigma = k2^l$ for proper integers k and l. By including matrix inversions into this game, we define the eigenvalue regions bounded by straight lines, their segments, circles and their arcs.

2.5 Computation of the matrix sign function

In [H08, equations (6.17)–(6.20)] Higham defines effective iterative algorithms for the square root function $B^{1/2}$; one can readily extend them to $\text{sign}(A) = A(A^2)^{-1/2}$.Then in [H80, Chapter 5], he presents a number of effective algorithms devised directly for the matrix sign function. Among them we recall Newton's iteration

$$N_0 = A, \qquad N_{i+1} = \frac{1}{2}(N_i + N_i^{-1}), \qquad i = 0, 1, ..., \tag{2.5.1}$$

based on the Möbius transform $x \to (x + 1/x)/2$, and the [2/0] Padé iteration

$$N_0 = A, \qquad N_{i+1} = \frac{1}{8}(15I_n - 10N_i^2 + 3N_i^4)N_i, \qquad i = 0, 1, ... \tag{2.5.2}$$

Theorem 1.3.2 implies the following simple corollary.

Corollary 2.5.1. *Suppose throughout iterations (2.5.1) and (2.5.2) none of the matrices* N_i *is singular.*

Let $\lambda = \lambda^{(0)}$ be an eigenvalue of the matrix N_0 ($\lambda^{(0)} \neq 0$) and define

$$\lambda^{(i+1)} = \frac{1}{2}(\lambda^{(i)} + (\lambda^{(i)})^{-1}), \qquad i = 0, 1, ..., \tag{2.5.3}$$

$$\lambda^{(i+1)} = \frac{1}{8}\lambda^{(i)}(15 - 10(\lambda^{(i)})^2 + 3(\lambda^{(i)})^4), \qquad i = 0, 1, ... \tag{2.5.4}$$

Then $\lambda^{(i)} \in \Lambda(N_i)$ for i=1,2,... provided the pairs $\{N_i, \lambda^{(i)}\}$ are defined by the pairs of equations (2.5.1), (2.5.2) or (2.5.3), (2.5.4), respectively.

Corollary 2.5.2. *Assume the iterations (2.5.3) and (2.5.4) and let λ be purely imaginary. Then the images $\lambda^{(i)}$ are also purely imaginary.*

By virtue of the following theorems, the sequences $\{\lambda^{(0)}, \lambda^{(1)}, ...\}$ defined by the equations (2.5.3) and (2.5.4) converge to ± 1 exponentialy fast, right from the start. The convergence is quadratic for the sequence (2.5.3) provided $\Re(\lambda) \neq 0$ and cubic for the sequence (2.5.4) provided $|\lambda - \text{sign}(\lambda)| \leq \frac{1}{2}$.

Theorem 2.5.1. *(See [H08], [BP96, page 500]). Let $\lambda = \lambda^{(0)}, \delta = \text{sign}(\lambda)$ and $\gamma = |\frac{\lambda-\delta}{\lambda+\delta}|$. Consider the iteration on (2.5.3) with $\Re(\lambda) \neq 0$. Then*

$$|\lambda^{(i)} - \delta| \leq \frac{2\gamma^{2^i}}{1 - \gamma^{2^i}} \ \ for \ \ i = 0, 1, ... \tag{2.5.5}$$

Theorem 2.5.2. *Write $\delta_i = \text{sign}(\lambda^{(i)})$ and $\gamma_i = |\lambda^{(i)} - \delta_i|$ for $i = 0, 1, ...$ Assume (2.5.4) and suppose $\gamma_0 \leq \frac{1}{2}$. Then*

$$\gamma_i \leq \frac{32}{113}\left(\frac{113}{128}\right)^{3^i} for \ \ i = 1, 2, ... \tag{2.5.6}$$

Proof. We clarify the proof of [BP96, Proposition 4.1]. First verify that

$$\gamma_{i+1} = \frac{1}{8}\gamma_i^3|3(\lambda^{(i)})^2 + 9\lambda^{(i)} + 8|. \tag{2.5.7}$$

And therefore

$$\gamma_{i+1} \le \frac{113}{32}\gamma_i^3 \ \ for \ \ i = 0, 1, ... \tag{2.5.8}$$

Now the claimed bounds follow by induction on i because $\gamma_0 \le \frac{1}{2}$. □

2.6 Variants for real eigen-solving

As we mentioned we can reduce real eigen-solving for a real matrix M to the matrix sign computation for $A = M\sqrt{-1}$, but next we substitute $N_0 = M$ instead of $N_0 = A$ into matrix sign iterations (2.5.1) and (2.5.2) and equivalently rewrite them to avoid involving nonreal values,

$$N_0 = M, \ N_{i+1} = 0.5(N_i - N_i^{-1}) \text{ for } i = 0, 1, \dots, \tag{2.6.1}$$

$$N_0 = M, \ N_{i+1} = -(3N_i^5 + 10N_i^3 + 15N_i)/8 \text{ for } i = 0, 1, \dots. \tag{2.6.2}$$

The matrices N_i and the images $\lambda^{(i)}$ of every real eigenvalue λ of M are real for all i, thus the results of Theorems 2.5.1 and 2.5.2 are immediately extended. The images of every nonreal λ converge to $\text{sign}(\lambda)\sqrt{-1}$ quadratically under (2.6.1) if $\Re(\lambda) \neq 0$ and cubically under (2.6.2) if $\lambda \in \mathcal{D}_{1/2}(\text{sign}(\lambda)\sqrt{-1})$.

Under the maps $M \to I_n + N_i^2$ for N_i in the above iterations, the images $1 + (\lambda^{(i)})^2$ of nonreal eigenvalues λ of M in the respective basins of convergence converge to 0, whereas for real λ the images are real and are at least 1 for all i. Thus for sufficiently large integers i we yield strong domination of the eigenspace of N_i associated with the images of real eigenvalues of M.

2.6.1 Newton's iteration with shifts for real matrix sign function

Iteration (2.6.1) fails where for some i the matrix N_i is singular or nearly singular, that is has eigenvalue 0 or near 0. In that case, we can approximate this eigenvalue by applying the Rayleigh Quotient Iteration [GL96, Section 8.2.3], [BGP02/04] or the Inverse Orthogonal Iteration [GL96, page 339].

If we seek other real eigenvalues as well, we can deflate the matrix M and apply Algorithm 2.1.1 to the resulting matrix of a smaller size. Alternatively we can apply it to the matrix $N_i + \rho_i I_n$ for a shift ρ_i randomly generated in the range $-r \le \rho_i \le r$ for a positive r. We choose r reasonably small and then can expect that we will both avoid degeneracy and, by virtue of Theorems 2.3.1 and 2.5.2, have the images of all nonreal eigenvalues of M still rapidly converging to a small neighborhood of the points $\pm\sqrt{-1}$, thus ensuring their isolation from the images of real eigenvalues.

2.6.2 Controlling the norms in the $[2/0]$ Padé iterations

We have no singularity problem with iteration (2.6.2), but have numerical problems where the norms $||N_i||$ grow large. If the nonreal eigenvalues of the matrix N_0 lie in the two discs $\mathcal{D}_{1/2}(\pm\sqrt{-1})$, then their images also stay there by virtue of extension of Theorem 2.5.2, and then the norms $||N_i||$ can be large only where some real eigenvalues of the matrices N_i are absolutely large.

Now suppose the nonreal eigenvalues of M have been mapped into the two discs $\mathcal{D}_{y_i}(\pm\sqrt{-1})$ for $0 < y_i < 0.1$. (One or two steps (2.6.2) move every $\mu \in \mathcal{D}_{1/2}(\pm\sqrt{-1})$ into the discs $\mathcal{D}_{y_i}(\pm\sqrt{-1})$, cf. Theorem 2.5.2.) Then the transformation $N_i \to N_i(N_i^2 + 2I_n)^{-1}$ confronts excessive norm growth by mapping all real eigenvalues of

N_i into the range $[-\frac{1}{4}\sqrt{2}, \frac{1}{4}\sqrt{2}]$ and mapping all nonreal eigenvalues of N_i into the discs $\mathcal{D}_{w_i}(\pm\sqrt{-1})$ for $w_i \le \frac{1+y_i}{1-2y_i-y_i^2}$. E.g., $w_i < 0.4$ for $y_i = 0.1$, whereas $w_i < 0.17$ for $y_i = 0.05$, and then single step (2.6.2) would more than compensate for such a minor dilation of the discs $\mathcal{D}_{y_i}(\pm\sqrt{-1})$ (see Theorem 2.5.2).

2.7 Modifications with fewer matrix inversions

We should apply iteration (2.6.2) rather than (2.6.1) to exploit its cubic convergence and to avoid matrix inversions as soon as the images of the targeted eigenvalues λ of M have been moved into the discs $\mathcal{D}_{1/2}(\pm\sqrt{-1})$. Our goal is to achieve this in fewer steps (2.6.1) based on nonreal computations and repeated squaring (2.2.1) for appropriate matrices M_0.

2.7.1 Mapping the real line onto unit circle and repeated squaring

Next we incorporate repeated squaring of a matrix between its back and forth transforms defined by the maps of the complex plane $\mu \to \lambda$ and $\lambda \to \mu$ below.

Theorem 2.7.1. *Write* $\lambda = u + v\sqrt{-1}$,

$$\mu = (a\lambda + \sqrt{-1})(a\lambda - \sqrt{-1})^{-1},\ \beta_k = \frac{\sqrt{-1}(\mu^k + 1)}{a(\mu^k - 1)} \tag{2.7.1}$$

for a positive integer k *and a real* $a \neq 0$ *(one can simply choose* $a = 1$*, but other choices can be more effective). Then*

(a) $\lambda = \frac{\sqrt{-1}(\mu+1)}{a(\mu-1)}$,

(b) $\mu = \frac{n(\lambda)}{d(\lambda)}$ *for* $n(\lambda) = u^2 + v^2 - a^2 = 2au\sqrt{-1}$ *and* $d(\lambda) = u^2 + (v-a)^2)^2$, *and consequently*

(c) $|\mu|^2 = \frac{u^2+(v+a)^2}{u^2+(v-a)^2} = 1 + \frac{4av}{u^2+(v-a)^2}$,

(d) $|\mu| = 1$ if and only if λ is real.

Furthermore

(e) $\beta_k = \frac{n_k(\lambda)}{d_k(\lambda)}$ *for* $n_k(\lambda) = \sum_{g=0}^{\lfloor k/2 \rfloor}(-1)^g \binom{k}{2g} (a\lambda)^{k-2g}$ *and*

$$d_k(\lambda) = a \sum_{g=0}^{\lfloor k/2 \rfloor}(-1)^{g+1} \binom{k}{2g+1} (a\lambda)^{h-2g-1}.$$

Theorem 2.7.1 implies that the transform $\lambda \to \mu$ maps the real line onto the unit circle $\mathcal{C}_1 = \{\mu : |\mu| = 1\}$, whereas the transform $\lambda \to \beta_k$ maps the real line into itself. Clearly, powering of μ keeps the unit circle $\mathcal{C}_1$ in place, whereas the values $|\mu|^k$ converge to 0 for $|\mu| < 1$ and to $+\infty$ for $|\mu| > 1$ as $k \to \infty$; thus for large k the transform $\lambda \to \beta_k$ isolates the images of the sets of real and nonreal values λ from one another.

Corollary 2.7.1. *Suppose that an $n \times n$ matrix M has exactly s eigenpairs $\{\lambda_j, \mathcal{U}_j\}$, $j = 1, \dots, s$, and does not have eigenvalues $\pm\sqrt{-1}/a$. By extending the equations of Theorem 2.7.1, write*

$$P = (aM + I_n\sqrt{-1})(aM - I_n\sqrt{-1})^{-1}, \tag{2.7.2}$$

$$M_k = \frac{\sqrt{-1}}{a}(P^k + 1)(P^k - 1)^{-1}, \tag{2.7.3}$$

$$\mu_j = (a\lambda_j + \sqrt{-1})(a\lambda_j - \sqrt{-1})^{-1},$$

$$\beta_{j,k} = \frac{n(\lambda_{j,k})}{d(\lambda_{j,k})}, \quad n(\lambda_{j,k}) = \sum_{g=0}^{\lfloor k/2 \rfloor}(-1)^g \binom{k}{2g} (a\lambda_j)^{k-2g},$$

$$d(\lambda_{j,k}) = a\sum_{g=0}^{\lfloor k/2\rfloor}(-1)^{g+1}\binom{k}{2g+1}(a\lambda_j)^{k-2g-1},$$

$j = 1, \dots, s$. *(In particular* $M_1 = M$, *whereas* $2M_2 = M - (aM)^{-1}$.*) Then* $M_k = n_k(M)(d_k(M))^{-1}$ *where*

$$n_k(M) = \sum_{g=0}^{\lfloor k/2\rfloor}(-1)^{g}\binom{k}{2g}(aM)^{k-2g},$$

$$d_k(M) = a\sum_{g=0}^{\lfloor k/2\rfloor}(-1)^{g+1}\binom{k}{2g+1}(aM)^{h-2g-1},$$

and the matrices M_k *have the eigenpairs* $\{\{\beta_{j,k}, \mathcal{U}_j\},\ j = 1, \dots, s\}$ *where* $\beta_{j,k}$ *are real if* λ_j *is real,* $|\beta_{j,k}| + 1/|\beta_{j,k}| \to \infty$ *as* $k \to \infty$ *unless* λ_j *is real.*

The corollary implies that for sufficiently large integers k we can set $F(M) = M_k$ in Algorithm 2.1.1.

We can apply repeated squaring to compute high powers P^k. In numerical implementation we must avoid involving large norms $||P^k||_q$. We can readily estimate them based on the Power Method [B74], [D83]. Also note that $(\rho(P))^k = \rho(P^k) \le ||P^k||_q \le ||P||_q^k$ for the spectral radii $\rho(P)$ and $\rho(P^k)$, $q = 1, 2, \infty$ and all k (cf. [S01, Theorems 1.2.7 and 1.2.9]).

Below is an algorithm that implements this approach by using only two matrix inversions; this is much less than in iteration (2.6.1). The algorithm works for a large class of inputs M, although it fails for harder inputs M, which have many real and nearly real eigenvalues, and also other nonreal eigenvalues. The heuristic choice

$$v = 0,\ w = 1\ t \approx -\Re(\text{trace}(M)),\ a = \frac{t}{n},\ \text{and}\ \widehat{M} = M + tI_n \tag{2.7.4}$$

tends to push the values $|\mu|$ away from 1 on the average input although can strongly push such a value toward 1 for the worst case input.

Algorithm 2.7.1. Mapping the real line onto the unit circle and repeated squaring *(cf. Remark 2.7.1).*

INPUT: *a real $n \times n$ matrix M, whose real and nearly real eigenvalues are associated with an unknown eigenspace $\mathcal{U}_+$ having an unknown dimension $r_+ \ll n$.*

OUTPUT: *a matrix $\widehat{U}$ such that $\mathcal{R}(\widehat{U}) \approx \mathcal{U}_+$ or FAILURE.*

INITIALIZATION: *Fix sufficiently large tolerances τ and h_+, fix real a, t, v, and w and matrix $\widehat{M}$ of (2.7.4).*

COMPUTATIONS:

1. *Compute the matrices $P = (a\widehat{M} + I_n\sqrt{-1})(a\widehat{M} - I_n\sqrt{-1})^{-1}$ (cf. Corollary 2.7.1) and P^{2^g} for $g = 1, 2, \dots, h+1$ until $||P^{2^{h+1}}||_q > \tau$ for a fixed q (e.g., for $q = 1$ or $q = \infty$) or until $h \geq h_+$.*
2. *Compute matrix M_k of Corollary 2.7.1 for $k = 2^{h_+}$.*
3. *Apply Algorithm 2.1.2 to the matrix $F = M_k$ and the integer $r = n$ to output an $n \times r$ matrix basis for the strongly dominant eigenspace $\widehat{U}$ of F.*
4. *Output FAILURE if Algorithm 2.1.2 fails, which would mean that the matrix $F = M_k$ has no strongly dominant eigenspace of dimension $r_+ < n$.*

One can modify Stage 4 to compute an integer h_+ iteratively, according to a fixed policy: one can begin with a small h_+, then increase it and reapply the algorithm if the computations fail (see Stage 4 and see further variations in Sections 2.7.2).

Remark 2.7.1. *(a) One can extend Stage 2 by setting $N_0 = M_k$ and applying iteration (2.6.2). In this case cubic convergence would be exploited and we could proceed with smaller values of h_+.*

(b) In another variant one computes the matrix P^s for a sufficiently large integer s to ensure isolation of the images of real and nearly real eigenvalues of M from the images of its other eigenvalues and then applies the Rayleigh Quotient Iteration to this matrix at sufficiently many points of the unit circle $\mathcal{C}_1(0)$.

2.7.2 Further variations of matrix sign iterations

Let us comment on some promising variations of the matrix sign iteration.

1. We first examine how the map $M \to P$ for the matrices M and P of Corollary 2.7.1 transforms the basin of convergence of iteration (2.6.2), given by the discs $\mathcal{D}_{1/2}(\pm\sqrt{-1})$. We observe that their complement is mapped into the annulus $\mathcal{A}_{1/5,5}(0) = \{x : \ 1/5 \le |x| \le 5\}$. Conversely, suppose that under the map $M \to P^k$ the images of all nonreal eigenvalues of M lie outside this annulus, then iteration (2.6.2) cubically converges when it is applied to the matrix M_k of Corollary 2.7.1. We can estimate the integer k supporting the cubic convergence if we know the absolute values of all eigenvalues of the matrix P, that is, their distances from the origin. By virtue of part (d) of Theorem 2.7.1 the distance is 1 if and only if an eigenvalue of P is the image of a real eigenvalue of M. For a large class of matrices M we can readily estimate the absolute values of the eigenvalues of the matrix P by using Gerschgörin discs [GL96, page 320], [S01, page 39].

2. Next we recall some relevant techniques used for polynomial root-finding. Suppose that we are given the coefficients of the characteristic polynomial $c_P(x) =$

$\det(xI_n - P)$. Then we can approximate all these values with relative errors of at most 1% by using $O(n \log n)$ flops (see the effective techniques of [B96], [BF00], [P00b], [P01], [S82]). Furthermore we can apply our algorithms to the companion matrix P to compute its eigenvalues μ lying on the unit circle $\mathcal{C}_1$, and then recover the real eigenvalues $\lambda = \frac{\sqrt{-1}(\mu+1)}{a(\mu-1)}$ of M (see part (a) of Theorem 2.7.1).

3. In the case where $M = C_p$ is the companion matrix of a polynomial $p(x)$, the monic characteristic polynomial $C_p(x)$ equals $\gamma(x-1)^n p(\frac{x+1}{x-1}\frac{\sqrt{-1}}{a})$, that is, $\gamma(x-1)^n p(1 - \frac{2}{x-1}\frac{\sqrt{-1}}{a})$ for a scalar γ.

We can compute its coefficients by performing two shifts of the variables and the reversion of the polynomial coefficients (see [P01, Chapter 2] on these polynomial operations).

4. We can replace repeated squaring of the matrix P with k steps of the Dandelin's root-squaring iteration, also attributed to Lobachevsky and Gräffe (see [H59]),

$$p_{i+1}(x) = (-1)^n p_i(\sqrt{x}) p_i(-\sqrt{x}), \quad i = 0, 1, \ldots, k-1 \tag{2.7.5}$$

for $p_0(x) = c_P(x)$. We have $p_i(x) = \prod_{j=1}^{n}(x - \lambda_j^{2^i})$, so that the ith iteration step squares the roots of the polynomial $p_{i-1}(x)$ for every i. Every root-squaring step (2.7.5) essentially amounts to polynomial multiplication and can be performed in $O(n \log n)$ flops; one can improve numerical stability by increasing this count to order n^2 [MZ01]. Having computed the polynomial $p_k(x)$, for a sufficiently large integer k, we have its roots on the unit circle sufficiently well isolated from its other roots. The application of the algorithm in the next section to C_{p_k}, the companion matrix of this polynomial, yields its roots lying on $\mathcal{C}_1$ (they are the eigenvalues of C_{p_k}). From these roots we can recover the roots μ of the circle $c_P(x) = p_0(x)$ by means of the

descending techniques of [P95] (applied also in [P96], [P97], [P01a], and [PZ10/11] Stage 8 of Algorithm 9.1]), and then can recover the real roots λ of $p(x)$ from the values μ by applying the expression in part (a) of Theorem 2.7.1.

Remark 2.7.2. *Having isolated the roots of $p_k(x)$ on the circle $\mathcal{C}_1$ from its other roots, we can apply the algorithms of [K98], [P95], [P96], [P01a], [S82] to split out the factor $f(x)$ of this polynomial sharing with $p_k(x)$ precisely all the roots on the circle $\mathcal{C}_1$. Then these roots can be readily approximated based on the Laguerre or modified Laguerre algorithms. Numerical problems can be caused by potentially dramatic growth of the coefficients of $p_k(x)$ in the transition to the factor $f(x)$ unless its degree is small.*

2.8 Repeated squaring and the Möbius transform

Our next iteration begins in the same way as Algorithm 2.7.1, but we interrupt repeated squaring by applying the scaled Möbius transform $x \to x + 1/x$, instead of the map $P \to M_k$ of (2.7.3). The scaled Möbius transform moves the images of all real eigenvalues of the matrix M from the unit circle $\mathcal{C}_1$ into the real line interval $[-2, 2]$; furthermore under this transform of the matrix N_i the images of all its eigenvalues lying outside the annulus $\mathcal{A}_{1/3,3}(0) = \{x : \ 1/3 \le |x| \le 3\}$ are moved into the exterior of the disc $D_{8/3}(0)$. Recall that the basin of convergence of iteration (2.6.2) preceded by the map $x \to \frac{\sqrt{-1}}{a}\frac{1+x}{1-x}$ was the exterior of the slightly larger annulus $\mathcal{A}_{1/5,5}(0) = \{x : \ 1/5 \le |x| \le 5\}$; furthermore the Möbius transform numerically stabilizes the computations for a large class of inputs.

Next we comment on combining the maps of Theorem 2.7.1, repeated squaring, and the Möbius transform; we observe some pitfalls and propose remedies.

Theorem 2.8.1. *(Cf. Theorem 2.7.1 for $a = 1$.) Write*

$$\mu = (\lambda + \sqrt{-1})(\lambda - \sqrt{-1})^{-1}. \tag{2.8.1}$$

Then

(a) $\lambda = \sqrt{-1}(\mu - 1)/(\mu + 1)$,

(b) $|\mu| = 1$ *if and only if* λ *is real and*

(c) $\mu_k = \mu^k + \mu^{-k} = \sum_{g=0}^{k}(-1)^g \binom{2k}{2g} \lambda^{2k-2g}(\lambda^2+1)^{-k}$ *for* $k = 1, 2, \dots$. *(In particular* $\mu_1 = \frac{\lambda^2-1}{\lambda^2+1}$, *whereas* $\mu_2 = \frac{\lambda^4-6\lambda^2+1}{(\lambda^2+1)^2}$.*)*

Theorem 2.8.2. *Assume* μ *of (2.8.1) and a nonnegative integer* k. *Then* $|\mu| = 1$ *and* $-2 \le \mu_k = \mu^k + \mu^{-k} \le 2$ *if* λ *is real, whereas* $|\mu^k + \mu^{-k}| \to \infty$ *as* $k \to \infty$ *otherwise.*

Corollary 2.8.1. *Let an* $n \times n$ *matrix* M *have exactly* s *eigenpairs* $\{\lambda_j, \mathcal{U}_j\}$, $j = 1, \dots, s$, *and not have eigenvalues* $\pm\sqrt{-1}$. *By extending (2.7.2) for* $a = 1$ *and (2.8.1), write*

$$P = (M + I_n\sqrt{-1})(M - I_n\sqrt{-1})^{-1} = (M - I_n\sqrt{-1})^{-1}(M + I_n\sqrt{-1}),$$

$$T_k = P^k + P^{-k} = \sum_{g=0}^{k}(-1)^g \binom{2k}{2g} M^{k-2g}(M^2+1)^{-k}, \tag{2.8.2}$$

$$\mu_j = (\lambda_j + \sqrt{-1})(\lambda_j - \sqrt{-1})^{-1},$$

$$\mu_{j,k} = \mu_j^k + \mu_j^{-k} = \sum_{g=0}^{k}(-1)^g \binom{2k}{2g} \lambda_j^{k-2g}(\lambda_j^2+1)^{-k}$$

for $k = 1, 2, \dots$ *(In particular* $T_1 = 2(I_n - M^2)(I_n + M^2)^{-1} = 2I_n - 4(I_n + M^2)^{-1}$, *whereas* $T_2 = (M^4 - 6M^2 + I_n)(M^2 + I_n)^{-2} = (M^2 + I_n)^{-2}(M^4 - 6M^2 + I_n)$.*) Then* $M = \sqrt{-1}(P - I_n)(P + I_n)^{-1} = \sqrt{-1}(P + I_n)^{-1}(P - I_n)$, $\lambda_j = \sqrt{-1}(\mu_j - 1)/(\mu_j + 1)$ *for* $j = 1, \dots, s$, *and the matrices* T_k *have the eigenpairs* $\{\{\mu_{j,k}, \mathcal{U}_j\},\ j = 1, \dots, s\}$ *where* $-2 \le \mu_{j,k} \le 2$ *if* λ_j *is real,* $|\mu_{j,k}| \to \infty$ *as* $h \to \infty$ *unless* λ_j *is real.*

Instead of the map $P^k \to M_k$ and equation (2.7.3) of Corollary 2.7.1 we employ the map $P^k \to T_k$ and equation (2.8.2). This complicates the isolation of the images of real eigenvalues of the matrix M from the images of its nonreal eigenvalues provided that we rely on the respective map of the eigenvalues $\lambda = \lambda(P^k) \to \lambda(T_k) = \lambda + 1/\lambda$. Indeed the unit circle $\{\lambda = \lambda(P^k) : |\lambda| = 1\}$ is still mapped onto the line segment $[-2, 2]$, but also the imaginary line $\{\lambda = \lambda(P^k) : \Re(\lambda) = 0\}$ is mapped into the real line.

The problem disappears, however, where $\max\{|\lambda|, 1/|\lambda|\} > 3$, because in this domain the value $|\lambda + 1/\lambda|$ exceeds $8/3$, whereas this value is small near the points $\lambda = \pm\sqrt{-1}$. Therefore we can safely apply the map $P^k \to T_k$ provided that the images of nonreal eigenvalues of M in the map $M \to P^k$ do not lie in the annulus $\mathcal{A}_{1/3,3}(0) = \{x : 1/3 \le |x| \le 3\}$.

This map does not generally increase the minimum ratio of the absolute values of the images of real and nonreal eigenvalues of M, but it brings the images of all nonreal eigenvalues of M into the exterior of the disc $D_{8/3}(0)$, while sending the images of all real eigenvalues of M into the real line interval $[-2, 2]$. If at this stage we can afford a reasonably large number of squarings of the resulting matrix (resp. its inverse), then the eigenspace associated with real eigenvalues of M becomes dominated (resp. dominant), and we can approximate them by applying Algorithm 2.1.3 (resp. 2.1.1).

Chapter 3

Approximating real roots of a polynomial

In this chapter we present a "matrix free" algorithm that avoids matrix inversion.

3.1 Some Basic Results for Polynomial Computations

3.1.1 Maps of the Variables and the Roots

Some important maps of the roots of a polynomial can be computed at a linear or nearly linear cost.

Theorem 3.1.1. *(*Root Inversion, Shift and Scaling*, cf. [P01].) Given a polynomial $p(x)$ of (1.3.1) and two scalars a and b, one can compute the coefficients of the polynomial $q(x) = p(ax + b)$ by using $O(n \log(n))$ flops. This bound decreases to $2n - 1$ multiplications if $b = 0$. Reversing a polynomial inverts all its roots involving no flops, that is, $p_{\rm rev}(x) = x^n p(1/x) = \sum_{i=0}^{n} p_i x^{n-i} = p_n \prod_{j=1}^{n}(1 - xx_j)$.*

Theorem 3.1.2. *(*Root Squaring*, cf. [H59].) (i) Let a polynomial $p(x)$ of (1.3.1) be monic. Then $q(x) = (-1)^n p(\sqrt{x})p(-\sqrt{x}) = \prod_{j=1}^{n}(x - x_j^2)$, and (ii) one can evaluate $p(x)$ at the k-th roots of unity for $k > 2n$ and then interpolate to $q(x)$ by using*

$O(n\log(n))$ *flops.*

Recursive root-squaring is prone to numerical problems because the coefficients of the iterated polynomials very quickly span many orders of magnitude. One can overcome this deficiency by applying a special tangential representation of the coefficients and intermediate results and by using $O(n^2)$ flops per iteration (cf. [MZ01]).

Theorem 3.1.3. *(*The Cayley Maps*, cf. [GL96].) The maps* $y = (x+\sqrt{-1})/(x-\sqrt{-1})$ *and* $x = \sqrt{-1}(y+1)/(y-1)$ *send the real axis* $\{x:\ x \text{ is real}\}$ *onto the unit circle* $C(0,1) = \{y:\ |y| = 1\}$*, and vice versa.*

3.1.2 Auxiliary algorithm

Theorem 3.1.4. *(*Root-finding Where All Roots Are Real*). The modified Laguerre algorithm of [DJLZ97] converges to all roots of a polynomial* $p(x)$ *of (1.3.1) right from the start, uses* $O(n)$ *flops per iteration, and therefore approximates all* n *roots within* $\epsilon = 1/2^b$ *by using* $O(\log(b))$ *iterations and performing* $\tilde{O}(n\log(b))$ *flops overall. This asymptotic cost bound is optimal and is also supported by the alternative algorithms of [BT90] and [BP98].*

3.2 Cayley Map and Root-squaring

The algorithm below summarizes the following ideas: Given a polynomial $p(x)$ if we are interested only in approximations of the real roots of $p(x)$ we can map the roots of $p(x)$ to the unit circle via a Cayley map, after that repeatedly square the roots of the new polynomial, dividing the resulting polynomial by its largest coefficient. Then we can eventually extract a factor of an image of $p(x)$ that we can map back to the real

line via the inverse of the Cayley map and recursive descending. Then by applying Algorithms 2.1.1 and 2.1.2 we can output approximations to the real roots of $p(x)$.

Algorithm 3.2.1. Real root-finding with Cayley map and repeated root-squaring.

INPUT: *two integers n and r, $0 < r < n$, and the coefficients of a polynomial $p(x)$ of equation (1.3.1) where $p(0) \neq 0$ and $p(1)p(\sqrt{-1}) \neq 0$.*

OUTPUT: *Approximations of the real roots $x_1, ..., x_r$ of the polynomial $p(x)$.*

COMPUTATIONS:

1. *Compute the polynomial $q(x) = (x-1)^n p(\sqrt{-1}\ \frac{x+1}{x-1}) = \sum_{i=0}^{n} q_i x^i$. (This is the Cayley map of Theorem 3.1.3. It moves the real axis, in particular the real roots of $p(x)$, onto the unit circle $C(0,1)$.)*

2. *Write $q_0(x) = q(x)/q_n$, fix a sufficiently large integer k, and apply the k squaring steps of Theorem 3.1.3, $q_{h+1}(x) = (-1)^n q_h(\sqrt{x}) q_h(-\sqrt{x})$ followed by division of the resulting polynomial by the largest norm of its coefficients, for $h = 0, 1, \ldots, k-1$. (These steps keep the images of the real roots of $p(x)$ on the circle $C(0,1)$ for all k, while sending the images of every other root of $p(x)$ toward either the origin or the infinity.)*

3. *For a sufficiently large integer k, the polynomial $q_k(x)$ approximates the polynomial $x^s u_k(x)$ where $u_k(x) = \sum_{i=0}^{r} u_i x^i$ and has all roots lying on the unit circle $C(0,1)$. Extract the approximation to this polynomial $u_k(x)$ from the coefficients of the polynomial $q_k(x)$.*

4. *Compute the polynomial* $w_k(x) = u_k(\frac{x+\sqrt{-1}}{x-\sqrt{-1}})$. *(This Cayley map sends the images of the real roots of the polynomial* $p(x)$ *lying on the unit circle* $C(0,1)$ *back to the real line.)*

5. *Apply one of the algorithms of [BT90], [BP98], and [DJLZ97] to approximate the* r *real roots* $z_1, \dots, z_r$ *of the polynomial* $w_k(x)$ *(cf. Theorem 3.1.4).*

6. *Apply the Cayley map* $w_j^{(k)} = (z_j + \sqrt{-1})/(z_j - \sqrt{-1})$ *for* $j = 1, \dots, r$ *to extend Stage 5 to approximating the* r *roots* $x_1^{(k)}, \dots, x_r^{(k)}$ *of the polynomials* $u_k(x)$ *and* $y_k(x) = x^s u_k(x)$ *lying on the unit circle* $C(0,1)$.

7. *Apply the descending process (similar to the ones of [P95] and [P02])) to approximate the* r *roots* $x_1^{(h)}, \dots, x_r^{(h)}$ *of the polynomials* $q_h(x)$ *lying on the unit circle* $C(0,1)$ *for* $h = k-1, \dots, 0$.

8. *Approximate the* r *real roots* $x_j = \sqrt{-1}(x_j^{(0)} + 1)/(x_j^{(0)} - 1)$, $j = 1, \dots, r$, *of the polynomial* $p(x)$.

The overall cost of this algorithm is $O(kn \log n)$ flops.

Remark 3.2.1. *(*Refinement by means of Newton's iteration.*) For every* h, $h = k, k-1, \dots, 0$, *one can apply Newton's iteration* $x_{j,i+1}^{(h)} = x^{(h)} - p(x_{j,i}^{(h)})/p'(x_{j,i}^{(h)})$, $h = 0, 1, \dots,$ $i = 0, 1, \dots, l$, *concurrently at the* r *approximations* $x_j^{(h)}$, $j = 1, \dots, r$, *to the* r *real roots of the polynomial* $p_h(x)$. *We can perform* l *iteration loops by using* $O(nl \log^2(r))$ *flops, that is* $O(n \log^2(r))$ *flops per loop (cf. [P01, Section 3.1]).This can be added to the overall arithmetic cost of order* $kn \log(n)$ *for performing the other stages of the algorithm. We can perform the proximity tests of Stage 7 of the algorithm by applying Newton's iteration at all* $2r$ *candidate approximation points.*

Having selected r of them, we can continue applying the iteration at these points, to refine the approximations.

Remark 3.2.2. *(*Countering Degeneracy.*) If $p(0) = p_0 = \cdots = p_m = 0 \neq p_{m+1}$, then we should output the real root $x_0 = 0$ of multiplicity m and apply the algorithm to the polynomial $p(x)/x^m$ to approximate the other real roots.*

Remark 3.2.3. *(*The Number of Real Roots.*) We assume that we know the number r of the real roots (e.g., supplied by algorithms of computer algebra), but we can compute this number as by-product of Stage 3 of Algorithm 3.2.1.*

Chapter 4

Matrix version of Cayley map for Real eigen-solving

4.1 Preliminaries

Given polynomial $p(x)$ of equation (1.3.1). Suppose we are only looking for real roots of $p(x)$. This is equivalent to look for the real eigenvalues of its companion matrix C_p. If we let $M = C_p$ and apply to M the following matrix function (Cayley map matrix version)

$$H(M) = (M + \sqrt{-1}I)(M - \sqrt{-1}I)^{-1}.$$

Then based on Theorem 1.3.2 we will send all real eigenvalues of the companion matrix to the unit circle $C(0,1)$ (they have norms 1). Half of the nonreal eigenvalues will go inside the unit circle $C(0,1)$ (they have norms less than 1) the other half of nonreal eigenvalues will go to the exterior of the unit circle $C(0,1)$ (they have norms greater than 1).

Now Let $P = H(M)$. Apply to P the matrix function, $(P^k - P^{-k})^{-1}$ where k is a positive integer. For a sufficiently large integer k the images of the real eigenvalues of the input matrix C_p strongly dominate all other eigenvalues of the matrix $(P^k - P^{-k})^{-1}$ and so we generate a dominant eigenspace. Therefore we can apply

Algorithms 2.1.1 and 2.1.2 and output the approximations of all real roots of $p(x)$. However, since high powers of the input and auxiliary matrices rapidly increase the magnitude of its entries, they tend to have numerical rank 1, so we must consider an alternative of computing $P^k - P^{-k}$. Let $\omega_k = \exp(2\pi\sqrt{-1}/k)$ be a primitive kth root of unity. Then

$$P^k - P^{-k} = (P^{2k} - 1)P^{-k} = (\prod_{i=0}^{k-1}(P^2 - \omega_k^i))(P^{-k}) = \prod_{i=0}^{k-1}(P - \omega_k^i P^{-1}).$$

4.2 Real eigen-solving by means of factorization

Now consider extension of Algorithm 3.2.1 to real eigen-solving. We must avoid using high powers of the input and auxiliary matrices. The following algorithm involves such powers implicitly by computing the auxiliary matrices $P^k - P^{-k}$ for reasonably large integers k as the product $\prod_{i=0}^{k-1}(P - \omega_k^i P^{-1})$ where $\omega_k = \exp(2\pi\sqrt{-1}/k)$ is a primitive kth root of unity.

Algorithm 4.2.1. Real eigen-solving by means of factorization.

INPUT: *a real $n \times n$ matrix M having r real eigenvalues and $s = (n - r)/2$ pairs of nonreal complex conjugate eigenvalues, neither of them equal to $\sqrt{-1}$.*

OUTPUT: *approximations to the real eigenvalues $x_1, \ldots, x_r$ of the matrix M.*

COMPUTATIONS:

1. *Compute the matrix $P = (M + \sqrt{-1}\ I)(M - \sqrt{-1}\ I)^{-1}$. (This is the matrix version of a Cayley map of Theorem 3.1.3. It moves the real and only the real eigenvalues of the matrix M into the eigenvalues of the matrix P lying on the unit circle $C(0, 1)$.)*

2. *Fix a sufficiently large integer k and compute the matrix $Y = (P^k - P^{-k})^{-1}$ in the following factorized form $\prod_{i=0}^{k-1}(P - \omega_k^i P^{-1})^{-1}$ where $\omega_k = \exp(2\pi\sqrt{-1}/k)$. (For any integer k the images of all real eigenvalues of the matrix M have absolute values at least 2, whereas the images of all nonreal eigenvalues of that matrix converge to 0 as $k \to \infty$.)*

3. *Apply the randomized algorithms of [HMT11] to compute the numerical rank of the matrix Y. The rank is at least r, and if it exceeds r, then go back to Stage 1. If it is equal to r, then generate a standard Gaussian random $n \times r$ matrix G and compute the matrices $H = YQ(G)$ and $Q = Q(H)$. (The analysis of preprocessing with Gaussian random multipliers in [HMT11],[PQY14] shows that, with a probability close to 1, the columns of the matrix Q closely approximate a unitary basis of the invariant space of the matrix Y associated with its r absolutely largest eigenvalues, which are the images of the real eigenvalues of the matrix C_p. Having this approximation is equivalent to having a small upper bound on the residual norm $||Y - QQ^HY||$ ([HMT11], [PQY14]). Verify the latter bound. If the verification fails (which is unlikely), output FAILURE and stop the computations.*

4. *Otherwise compute and output approximations to the r eigenvalues of the $r \times r$ matrix $L = Q^H C_p Q$. They approximate the real roots of the polynomial $p(x)$. (Indeed, by virtue of Theorem 1.3.2, Q is an approximate matrix basis for the invariant space of the matrix C_p associated with its r real eigenvalues. Therefore, by virtue of Theorem 1.3.1, the r eigenvalues of the matrix L approximate the r real eigenvalues of the matrix C_p.)*

The arithmetic complexity of performing the algorithm is $O(kn^3)$ ops for general matrix M, but it decreases to $O(kn^2)$ if M is a Hessenberg matrix or if the rank of all its subdiagonal blocks is bounded by a constant. For $M = C_p$ the complexity decreases to $O(kn)$, which makes the algorithm attractive for real polynomial root-finding, as long as the algorithm converges for a reasonably small integer k.

Remark 4.2.1. *(*Scaling and simplified factorizations.*) One can apply the algorithm to a scaled matrix $\theta M/||M||$ for a fixed matrix norm $||\cdot||$ and a fixed scalar θ, $0 < \theta < 1$, say, for $\theta = 0.5$. In this case the inversion at Stage 1 is applied to a diagonally dominant matrix. Towards more radical simplification of the algorithm, one can avoid computing and inverting the matrix P and can instead compute the matrix Y in one of the following equivalent factorized forms,*

$$Y = \prod_{i=0}^{k-1}((M^2 + I)\ F_i(M)^{-1}G_i(M)^{-1}) = \prod_{i=0}^{k-1}(\alpha_i F_i(M)^{-1} + \beta_i G_i(M)^{-1})$$

for

$$F_i(M) = M + \sqrt{-1}\ I + \omega_{2k}^i(M - \sqrt{-1}\ I) = (1 + \omega_{2k}^i)M + \sqrt{-1}(1 - \omega_{2k}^i)I,$$

$$G_i(M) = M + \sqrt{-1}\ I - \omega_{2k}^i(M - \sqrt{-1}\ I) = (1 - \omega_{2k}^i)M + \sqrt{-1}(1 + \omega_{2k}^i)I,$$

and appropriate complex scalars α_i and β_i, $i = 0, \dots, k-1$. Then again, one can apply the algorithm to a scaled matrix γM for an appropriate scalar γ to simplify the solution of linear systems of equations with the matrices $F_i(M)$ and $G_i(M)$.

Remark 4.2.2. *One can adapt the integer k by doubling it if the computations show that it is not large enough to produce the desired eigenvalues. The previously computed matrices $F_i(M)$ and $G_i(M)$ can be reused.*

Bibliography

[A73] O. Aberth, Iteration Methods For Finding All Zeros of a Polynomial Simultaneously, *Mathematics of Computation*, **27**, **122**, 339–344, 1973.

[B40] E. T. Bell, *The Development of Mathematics*, McGraw-Hill, New York, 1940.

[B68] C. A. Boyer, *A History of Mathematics*, Wiley, New York, 1968.

[B74] D. W. Boyd, The Power Method for l^p Norms, *Linear Algebra and Its Applications*, **9**, 95–101, 1974.

[B96] D. A. Bini, Numerical Computation of Polynomial Zeros by Means of Aberth's Method, *Numerical Algorithms*, **13**, 179–200, 1996.

[BF00] D. A. Bini, G. Fiorentino, Design, Analysis, and Implementation of a Multiprecision Polynomial Rootfinder, *Numerical Algorithms*, **23**, 127–173, 2000.

[BG05] A. Böttcher, S. M. Grudsky, *Spectral Properties of Banded Toeplitz Matrices*, SIAM Publications, Philadelphia, 2005.

[BGP02/04] D. A. Bini, L. Gemignani, V. Y. Pan, Inverse Power and Durand/Kerner Iteration for Univariate Polynomial Root-finding, *Computers and Mathematics(with Applications)*, **47**, **2/3**, 447-459, 2004.

[BGP05] D. A. Bini, L. Gemignani, V. Y. Pan, Fast and Stable QR Eigenvalue Algorithms for Generalized Companion Matrices and Secular Equation, *Numerische Math.*, **3**, 373–408, 2005. (Also Technical Report 1470, *Department of Math., University of Pisa*, Pisa, Italy, July 2003.)

[BM91] C. B. Boyer, U. C. Merzbacher, *A History of Mathematics*, 2nd ed. New York: Wiley, 1991.

[BP94] D. A. Bini, V. Y. Pan, *Polynomial and Matrix Computations,* volume 1: *Fundamental Algorithms*, Birkhäuser, Boston, 1994.

[BP96] D. A. Bini, V. Y. Pan, Graeffe's, Chebyshev, and Cardinal's Processes for Splitting a Polynomial into Factors, *J. Complexity*, **12**, 492–511, 1996.

[BP98] D. A. Bini, V. Y. Pan, Computing Matrix Eigenvalues and Polynomial Zeros Where the Output Is Real, *SIAM Journal on Computing*, **27**, **4**, 1099–1115, 1998.

[BR14] D. A. Bini, L. Robol, Solving secular and polynomial equations: A multiprecision algorithm. *J. Computational and Applied Mathematics*, **272**, 276 - 292, 2014 .

[BT90] M. Ben-Or, P. Tiwari, Simple algorithms for approximating all roots of a polynomial with real roots. *J. Complexity*, **6, 4**, 417–442, 1990.

[C96] J. P. Cardinal, On Two Iterative Methods for Approximating the Roots of a Polynomial, *Lectures in Applied Mathematics*, **32** (*Proceedings of AMS-SIAM Summer Seminar: Mathematics of Numerical Analysis: Real Number Algorithms* (J. Renegar, M. Shub, and S. Smale, editors), Park City, Utah, 1995), 165–188, American Mathematical Society, Providence, Rhode Island, 1996.

[CD05] Z. Chen, J. J. Dongarra, Condition Numbers of Gaussian Random Matrices, *SIAM. J. on Matrix Analysis and Applications*, **27**, 603–620, 2005.

[CP00] C.-T. Pan, On the Existence and Computation of Rank-revealing LU Factorization, *Linear Algebra and Its Applications*, **316**, 199-222, 2000.

[D60] E. Durand, Solutions numériques des équations algébriques, *Tome 1: Equations du type F(X)=0; Racines d'un polynôme*, Masson, Paris, 1960.

[D83] J. D. Dixon, Estimating Extremal Eigenvalues and Condition Numbers of Matrices, *SIAM J. on Numerical Analysis*, **20, 4**, 812–814, 1983.

[D88] J. Demmel, The Probability That a Numerical Analysis Problem Is Difficult, *Math. of Computation*, **50**, 449–480, 1988.

[DJLZ97] Du, Q., Jin, M., Li, T.Y., Zeng, Z.: The quasi-Laguerre iteration. Math. Computation, 66(217), 345–361, 1997.

[E67] L. W. Ehrlich, A Modified Newton Method for Polynomials, *Comm. of ACM*, **10**, 107–108, 1967.

[E88] A. Eldelman, Eigenvalues and Condition Numbers of Random Matrices, SIAM J.*on Matrix Analysis and Applications*, **9**, **4**, 543-560, 1988.

[ES05] A. Edelman, B. D. Sutton, Tails of Condition Number Distributions, *SIAM J. on Matrix Analysis and Applications*, **27**, **2**, 547–560, 2005.

[EV89] van den Emden, A.W.M. and Verhoeckx, N.A.M. *Discrete-Time Signal Processing*, Prentice-Hall, New York, 1989.

[F02] J. C. Faugére, A New Efficient Algorithm for Computing Gröbner Bases without Reduction to Zero (F5), in *Proceedings of the International Symposium on Symbolic and Algebraic Computation (ISSAC '02)*, 75–83, ACM Press, New York, 2002.

[FC99] F. Cajori, *A History of Mathematics* 5/E, AMS Chelsea Publ., Providence, Rhode Island, 1999.

[GC96] M. Gu, S. C. Eisenstat, Efficient Algorithms For Computing a Strong Rank-Revealing QR Factorization, *SIAM Journal on scientific Computing*, **17**, 848-869, 1996.

[GL13] G. H. Golub, C. F. Van Loan, *Matrix Computations*, Johns Hopkins University Press, Baltimore, Maryland, 2013 (fourth edition).

[GL96] G. H. Golub, C. F. Van Loan, *Matrix Computation*, Johns Hopkins University Press, Baltimore, Maryland, 1996 (third edition).

[H08] N. J. Higham,*Functions of Matrices*, SIAM, Philadelphia, 2008.

[H59] A. S. Householder, Dandelin, Lobachevskii, or Graeffe. *Amer. Math. Monthly* **66**, 464-466, 1959.

[HMT11] N. Halko, P. G. Martinsson, J. A. Tropp, Finding structure with randomness: probabilistic algorithms for constructing approximate matrix decompositions. *SIAM Review*, **53(2)**, 217-288, 2011.

[HP92] Y. P. Hong, C.-T. Pan, Rank-Revealing QR Factorizations and the Singular Value Decomposition, *Mathematics of Computation*, **58**, **197**, 213-232, 1992.

[JT70] M. A. Jenkins, J. F. Traub, A Three-Stage Algorithm for Real Polynomials Using Quadratic Iteration, *SIAM J. Numer. Anal.*, **7(4)**, 545-566, 1970.

[K66] I. O. Kerner, Ein Gesamtschrittverfahren zur Berechung der Nullstellen von Polynomen, *Numerische Math.*, **8**, 290–294, 1966.

[K98] P. Kirrinnis, Polynomial Factorization and Partial Fraction Decomposition by Simultaneous Newton's Iteration, *J. of Complexity*, **14**, 378–444, 1998.

[MN93] J. M. McNamee, Bibliography on Roots of Polynomials, *J. Computational and Applied Mathematics*, **47**, 391–394, 1993.

[MN97] J. M. McNamee, A Supplementary Bibliography on Roots of Polynomials, *J. Computational and Applied Mathematics*, **78**, 1, 1997.

[MN99] J. M. McNamee, An Updated Supplementary Bibliography on Roots of Polynomials, *J. Computational and Applied Mathematics*, **110**, 305–306, 1999.

[MN02] J. M. McNamee, A 2002 Updated Supplementary Bibliography on Roots of Polynomials, *J. Computational and Applied Mathematics*, **142**, 433–434, 2002.

[MN07] J. M. McNamee, *Numerical Methods for Roots of Polynomials*, Elsevier, 2007.

[MP13] J. M. McNamee, V. Y. Pan, *Numerical Methods for Roots of Polynomials*, part2(XXII+718 pages), Elsevier, 2013.

[MZ01] G. Malajovich, J.P. Zubelli, J. P.: Tangent Graeffe Iteration. *Numerische Mathematik*, 89(4), 749-782, 2001.

[N85] V. Norton, Algorithm 631: Finding a bracketed zero by Larkin's method of rational interpolation, *ACM Trans. Math. Softw.***11**, 120-134, 1985.

[P95] V. Y. Pan, Optimal (up to polylog factors) sequential and parallel algorithms for approximating complex polynomial zeros. In: *Proc. 27th Ann. ACM Symp. on Theory of Computing*, pp. 741-750. ACM Press, New York, 1995.

[P96] V. Y. Pan, Optimal and Nearly Optimal Algorithms for Approximating Polynomial Zeros, *Computers and Math. (with Applications)* **31, 12**, 97–138, 1996.

[P97] V. Y. Pan, Solving a Polynomial Equation: Some History and Recent Progress, *SIAM Review*, **39, 2**, 187–220, 1997.

[P98] V. Y. Pan, Solving Polynomials with Computers, *American Scientist*, **86**, 62-69, (January-February 1998).

[P00] V. Y. Pan, Nearly Optimal Computations with Structured Matrices, *Proc. of 11th Ann. ACM-SIAM Symposium on Discrete Algorithms, (SODA 00)*, pages 953–962, ACM Press, New York, and SIAM Publications, Philadelphia, 2000.

[P00a] V. Y. Pan, Approximating Complex Polynomial Zeros: Modified Quadtree (Weyl's) Construction and Improved Newton's Iteration, *J. of Complexity*, **16**, **1**, 213–264, 2000.

[P01] V. Y. Pan, *Structured Matrices and Polynomials: Unified Superfast Algorithms*, Birkhäuser/Springer, Boston/New York, 2001.

[P01a] V. Y. Pan, Univariate Polynomials: Nearly Optimal Algorithms for Factorization and Rootfinding, *Journal of Symbolic Computations*, **33, 5**, 701–733, 2002. Proc. version in *Proc. International Symp. on Symbolic and Algebraic Computation (ISSAC 01),* 253–267, ACM Press, New York, 2001.

[P02] V. Y. Pan, Univariate Polynomials: Nearly Optimal Algorithms for Factorization and Rootfinding, *Journal of Symbolic Computations*, **33, 5**, 701–733, 2002.

[P05] V. Y. Pan, Amended DSeSC Power Method for Polynomial Rootfinding, *Computers and Math. (with Applications)*, **49**, **9**–**10**, 1515–1524, 2005.

[PQa] V. Y. Pan, G. Qian, Condition Numbers of Random Toeplitz and Circulant Matrices, Tech. Report TR 2012013, *PhD Program in Comp. Sci., Graduate Center, CUNY*, 2012. Available at

http://www.cs.gc.cuny.edu/tr/techreport.php?id=442

[PQY12] V. Y. Pan, G. Qian, Yan, X, More on the Power of Randomized Matrix Multiplication, Tech. Report TR 2012014, *PhD Program im Comp. Sci., Graduate Center, CUNY*, 2012. Available at

http://www.cs.gc.cuny.edu/tr/techreport.php?id=443

[PQY14] V. Y. Pan, G. Qian, X. Yan, Supporting GENP and low-rank Approximation with Ramdom Multipliers. Technical Report TR 2014008, *PhD Program in Computer Science. Graduate Center, CUNY*, 2014. Available at

http://www.cs.gc.cuny.edu/tr/techreport.php?id=472

[PQZa] V. Y. Pan, G. Qian, A. Zheng, Randomized Preconditioning of the MBA Algorithm, *Proc. International Symp. on Symbolic and Algebraic*

Computation (ISSAC 2011), (San Jose, California, June 2011), (edited by Anton Leykin), 281-288, ACM Press, New York, 2011.

[PQZb] V. Y. Pan, G. Qian, A. Zheng, Randomized Preprocessing versus Pivoting, *Linear Algebra and Its Applications*, 2013.

[PQZC11] V. Y. Pan, G. Qian, A. Zheng, Z. Chen, Matrix Computations and Polynomial Root-finding with Preprocessing, *Linear Algebra and Its Applications*, **434**, 854–879, 2011.

[PTVF07] W. H. Press, S. A. Teukolsky, W. T. Vetterling, B. P. Flannery, "Section 9.5.2. Muller's Method". Numerical Recipes: *The Art of Scientific Computing (3rd ed.)*. New York: Cambridge University Press. ISBN 978-0-521-88068-8, 2007.

[PZ10/11] V. Y. Pan, A. Zheng, New Progress in Real and Complex Polynomial Root-Finding, *Computers and Math. (with Applications)* 61, 1305-1334. Proceedings version: Real and Complex Polynomial Root-Finding with Eigen-Solving and Preprocessing, in *Proc. International Symp. on Symbolic and Algebraic Computation (ISSAC 2010)*, pages 219-226, ACM Press, New York, 2010.

[S01] G. W. Stewart, *Matrix Algorithms, Vol II: Eigensystems*, SIAM, Philadelphia, 2001 (second edition).

[S81] S. Smale, The Fundamental Theorem of Algebra and Complexity Theory, *Bull. American Math. Soc.* **4, 1**, 1–36, 1981

[S82] A. Schönhage, The Fundamental Theorem of Algebra in Terms of Computational Complexity, *Mathematics Department, University of Tübingen*, Germany, 1982.

[SST06] A. Sankar, D. Spielman, S.-H. Teng, Smoothed Analysis of the Condition Numbers and Growth Factors of Matrices, *SIAM Journal on Matrix Analysis*, **28**, **2**, 446–476, 2006.

[V03] J. Verschelde, *The method of Weierstrass (also known as the Durand-Kerner method)*, 2003.

[W02] D. S. Watkins *Fundamentals of Matrix Computations*, Wiley-Interscience, New York, 2002.

[W03] K. Weierstrass, Neuer Beweis des Fundamentalsatzes der Algebra, *Mathematische Werker*, Tome **III**, Mayer und Müller, Berlin, 251–269, 1903.

[WR67] E. T. Whittaker, G. Robinson, "A Method of Daniel Bernoulli." §52 in *The Calculus of Observations: A Treatise on Numerical Mathematics, 4th ed.* New York: Dover, pp. 98-99, 1967.