2013

TR-2013013: New Structured Matrix Methods for Real and Complex Polynomial Root-Finding

Victor Y. Pan

Ai-Long Zheng

New Structured Matrix Methods for Real and Complex Polynomial Root-finding *

Victor Y. Pan[1,2],[a] and Ai-Long Zheng[2],[b]
Supported by NSF Grant CCF-1116736 and PSC CUNY Award 64512–0042

[1] Department of Mathematics and Computer Science
Lehman College of the City University of New York
Bronx, NY 10468 USA
[2] Ph.D. Programs in Mathematics and Computer Science
The Graduate Center of the City University of New York
New York, NY 10036 USA
[a] victor.pan@lehman.cuny.edu
http://comet.lehman.cuny.edu/vpan/
[b] azheng-1999@yahoo.com

Abstract

We combine the known methods for univariate polynomial root-finding and for computations in the Frobenius matrix algebra with our novel techniques to advance numerical solution of a univariate polynomial equation, and in particular numerical approximation of the real roots of a polynomial. Our analysis and experiments show efficiency of the resulting algorithms.

2000 Math. Subject Classification: 65H05, 65F15, 30C15, 26C10, 12Y05

KEYWORDS: Polynomial, Root-finding, Eigen-solving, Companion matrix, Dominant eigenspaces, Real roots, Numerical approximation.

1 Introduction

Polynomial root-finding is the oldest subject of mathematics and computational mathematics and is still an area of intensive research worldwide. The list of hundreds if not thousands algorithms known for this task still grows every year (see the books and articles [2], [3], [14], [39], [40], [30], [31], [32], [48], and the bibliography therein). Many algorithms are directed to computing a single, e.g., absolutely largest root of a polynomial or a subset of all its n roots, e.g., all r its real roots. In some applications, e.g., to algebraic geometric optimization, only the real roots are of interest, and they can be much less numerous than all n complex roots. Nevertheless the best numerical subroutines such as MPSolve approximate all these r real roots about as fast and as slow as all n complex roots.

Root-finding for a polynomial $p(x)$ via eigen-solving for the associated companion matrix C_p is a classical approach recently revived, with the incorporation of the well developed numerical matrix methods (see [6], [20], [54], [59], and the bibliography therein). The QR algorithm, adopted for polynomial root-finding by Matlab, avoids numerical problems, faced by many other companion

*Some results of this paper have been presented at the 14th Annual Conference on Computer Algebra in Scientific Computing (CASC 2012), September 2012, Maribor, Slovenia, and at the 18th Conference of the International Linear Algebra Society (ILAS'2013), Providence, RI, June 2013

matrix methods [20, Section 7.4.6], but is not readily amenable to exploiting the rich structure of the companion matrix. Extensive research toward such exploitation by using QR- and LR-based root-finders has been initiated in the papers [10], [11] and [7] and still goes on (see [5], [57], [62], [1], and the references therein). The QR algorithm is celebrated for its fast empirical convergence, but the Rayleigh Quotient iteration [20, Section 8.2.2] also has very good convergence record, exploits matrix structures even better than the QR algorithm, and unlike that algorithm can be applied concurrently with no communication among the processors that handle distinct initial points. The papers [9], [49] adjust this iteration to polynomial root-finding and perform every iteration step and every deflation step in linear space and linear arithmetic time.

In this paper we explore the somewhat similar approach of Cardinal [13], extended in [12] and [45]. It enhances the Power Method and the method of [50], [51], and [22] by reducing every multiplication in the Frobenius algebra, generated by the companion matrix C_p, to application of a small number of FFTs. By combining these and some other known techniques of polynomial root-finding with our novelties, we achieve substantial progress, in particular for numerical approximation of the real roots. We reduce this task to the approximation of the associated eigenspace of the companion matrix (cf. Theorem 2.2), make this eigenspace dominant by using shifts, inversions and repeated squaring in the Frobenius matrix algebra as well as the approximation of the matrix sign function, and then readily approximate this eigenspace and the associated eigenvalues. Numerically we approximate the r_+ real and nearly roots of the input polynomial, and among them we immediately select all the r real roots (see Remark 3.2 in Section 3). In this way we accelerate the known numerical real root-finders by a factor of n/r_+ for a polynomial of a degree n. We also substantially accelerate the known numerical algorithms for complex roots of polynomials by proposing some novel matrix methods, as we show both formally and empirically.

We organize our presentation as follows. The next section is devoted to definitions and preliminary results. In Section 3 we present our basic algorithms. They reduce the eigenvalue problem to the approximation of the dominant or dominated eigenspaces of the appropriate functions of the input matrix. In the subsequent sections we cover the computation of such matrix functions. In Section 4 we do this by combining repeated squaring, shifts and inversions in the associated matrix algebra, whereas in Section 5 we exploit the approximation of the matrix sign function. Both sections are mostly devoted to the approximation of real eigenvalues, but Subsections 4.1, 5.2 and 5.7 present some novel efficient algorithms that approximate complex eigenvalues of the companion matrix and consequently complex roots of a polynomial. Section 6 covers our numerical tests, which are the contribution of the second author. There are many directions for extending and refining our techniques, and our concluding Section 7 lists some of them. In the Appendix we sketch a dual approach emulating some of our techniques in terms of polynomial computations.

2 Definitions and preliminaries

Hereafter "flop" stands for "arithmetic operation", "is expected" and "is likely" mean "with a probability near 1", and "small", "large", "close", and "near" are meant in the context. We assume computations in the fields of complex and real numbers $\mathbb{C}$ and $\mathbb{R}$, respectively. 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 scalar λ is *nearly real* (within $\epsilon > 0$) if $|\Im(\lambda)| \le \epsilon|\lambda|$.

Matrix computations: fundamentals [20], [53], [58]. $(B_j)_{j=1}^s = (B_1 \mid B_2 \mid \ldots \mid B_s)$ is the $1 \times s$ block matrix with the blocks $B_1, B_2, \ldots, B_s$. $\mathrm{diag}(B_j)_{j=1}^s = \mathrm{diag}(B_1, B_2, \ldots, B_s)$ is the $s \times s$ block diagonal matrix with the diagonal blocks B_1, B_2,..., B_s. 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. $\mathcal{N}(M) = \{\mathbf{v} :\ M\mathbf{v} = \mathbf{0}\}$ is its null space. $\mathrm{rank}(M) = \dim(\mathcal{R}(A))$. A matrix of full column rank is a *matrix basis* of its range. $I = I_n = (\mathbf{e}_1 \mid \mathbf{e}_2 \mid \ldots \mid \mathbf{e}_n)$ is the $n \times n$ identity matrix with columns $\mathbf{e}_1,\ \mathbf{e}_2, \ldots, \mathbf{e}_n$. $J = J_n = (\mathbf{e}_n \mid \mathbf{e}_{n-1} \mid \ldots \mid \mathbf{e}_1)$ is the $n \times n$ reflection matrix, $J^2 = I$. $O_{k,l}$ is the $k \times l$ matrix filled with zeros. A matrix Q is called *orthogonal* (also *unitary* and *orthonormal*) if $Q^TQ = I$ or $QQ^T = I$.

Theorem 2.1. *[20, Theorem 5.2.2]. A matrix M of full column rank has unique QR factorization $M = QR$ where $Q = Q(M)$ is an orthogonal matrix and $R = R(M)$ is a square upper triangular matrix with positive diagonal entries.*

We use the matrix norms $||\cdot||_h$ for $h = 1, 2, \infty$ [20, Section 2.3] and write $||\cdot|| = ||\cdot||_2$. We write $a \approx 0$ and $A \approx O$ if the values $|a|$ and $||A||$ are small in context. We write $a \approx b$ for $b \neq 0$ and $A \approx B$ and $B \neq O$ if the ratios $|a|/|b|$ and $||A||/||B||$ are small.

M^+ is the Moore–Penrose pseudo inverse of M [20, Section 5.5.4]. An $n \times m$ matrix $X = M^{(I)}$ is a left (resp. right) inverse of an $m \times n$ matrix M if $XM = I_n$ (resp. 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.

Matrix computations: eigenspaces [20], [54], [58], [59], [6]. $\mathcal{S}$ is an *invariant subspace* or *eigenspace* of a square matrix M if $M\mathcal{S} = \{M\mathbf{v} : \mathbf{v} \in \mathcal{S}\} \subseteq \mathcal{S}$.

Theorem 2.2. *[54, Theorem 4.1.2], [58, Section 6.1], [59, Section 2.1]. 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 $\{L,\mathcal{U}\}$ is an *eigenpair* of a matrix M, L is its *eigenblock*, and $\mathcal{U}$ is the *associated eigenspace* of L [54]. If $L = \lambda I_n$, then also $\{\lambda,\mathcal{U}\}$ is called an *eigenpair* of a matrix M. In this case $\det(\lambda I - M) = 0$, whereas $\mathcal{N}(M - \lambda I)$ 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 2.2 implies that $\Lambda(L) \subseteq \Lambda(M)$. For an eigenpair $\{\lambda,\mathcal{U}\}$ write $\psi = \min\ |\lambda/\mu|$ over $\lambda \in \Lambda(L)$ and $\mu \in \Lambda(M) - \Lambda(L)$. Call the eigenspace $\mathcal{U}$ *dominant* if $\psi > 1$, *dominated* if $\psi < 1$, *strongly dominant* if $1/\psi \approx 0$, and *strongly dominated* if $\psi \approx 0$. 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 2.3. *(See [23, Theorem 1.13].) $\Lambda(\phi(M)) = \phi(\Lambda(M))$ for a square matrix M and a function $\phi(x)$ defined on its spectrum. Furthermore $(\phi(\lambda),\mathcal{U})$ is an eigenpair of the matrix $\phi(M)$ if the matrix M is diagonalizable and has an eigenpair $(\lambda,\mathcal{U})$.*

A nonsingular matrix M is *well conditioned* if its condition number $\kappa(M) = ||M||\ ||M^{-1}|| \geq 1$ is reasonably bounded. This matrix is *ill conditioned* if its condition number is large. $\kappa(M) = ||M|| = ||M^+|| = 1$ for orthogonal matrices M.

Toeplitz matrices [43, Ch. 2]. 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

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

Polynomials and companion matrices. Write

$$p(x) = \sum_{i=0}^{n} p_i x^i = p_n \prod_{j=1}^{n} (x - \lambda_j), \tag{2.1}$$

$$p_{\rm rev}(x) = x^n p(1/x) = \sum_{i=0}^{n} p_i x^{n-i} = p_n \prod_{j=1}^{n} (1 - x\lambda_j), \tag{2.2}$$

$p_{\rm rev}(x)$ is the reverse polynomial of $p(x)$,

$$C_p = \begin{pmatrix} 0 & & & & -p_0/p_n \\ 1 & \ddots & & & -p_1/p_n \\ & \ddots & \ddots & & \vdots \\ & & \ddots & 0 & -p_{n-2}/p_n \\ & & & 1 & -p_{n-1}/p_n \end{pmatrix}, \text{ for } \mathbf{p} = (p_j)_{j=0}^{n-1},$$

and $C_{p_{\rm rev}} = JC_pJ$ are the $n \times n$ companion matrices of the polynomials $p(x) = \det(xI_n - C_p)$ and $p_{\rm rev}(x) = \det(xI_n - C_{p_{\rm rev}})$, respectively.

Fact 2.1. *(See [13] or [45].) The companion matrix $C_p \in \mathbb{C}^{n\times n}$ of a polynomial $p(x)$ of (2.1) generates an algebra $\mathcal{A}_p$ of matrices having structure of Toeplitz type. One needs $O(n)$ flops for addition, $O(n \log n)$ flops for multiplication and $O(n \log^2 n)$ flops for inversion in this algebra and needs $O(n \log n)$ flops for multiplying a matrix from the algebra by a square Toeplitz matrix.*

3 Basic algorithms for approximating selected eigenvalues

The following algorithms employ Theorems 2.2 and 2.3 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. They will serve as the basis for our eigenvalue algorithms, which we will apply to the companion matrices in the subsequent sections.

Algorithm 3.1. Reduction of the input size for eigen-solving.

INPUT: *a diagonalizable matrix $M \in \mathbb{R}^{n\times n}$ and a property that specifies a subset Λ of its unknown spectrum associated with an unknown eigenspace $\mathcal{U}$.*

OUTPUT: *two matrices $\widehat{L}$ and $\widehat{U}$ such that the pair $\{\Lambda(\widehat{L}), \mathcal{R}(\mathcal{U})\}$ closely approximates the eigenpair $\{\Lambda, \mathcal{U}\}$ of the matrix M.*

COMPUTATIONS:

1. *Compute a matrix function $\phi(M)$ for which the linear space $\mathcal{U}$ is a strongly dominant eigenspace.*
2. *Compute and output a matrix $\widehat{U}$ of full column rank whose range approximates the eigenspace $\mathcal{U}$.*
3. *Compute the left inverse $\widehat{U}^{(I)}$ of the matrix $\widehat{U}$.*
4. *Compute and output the matrix $\widehat{L} = \widehat{U}^{(I)} M \widehat{U}$.*

At Stage 2 of the algorithm, one can apply a rank revealing QR or LU factorization of the matrix $\phi(M)$ [19], [24], [42]. Given a reasonably close upper bound r_+ on the dimension r of the eigenspace $\mathcal{U}$, we can alternatively employ a randomized multiplier as follows.

Algorithm 3.2. Approximation of a dominant eigenspace.

INPUT: *a positive integer r_+ and a diagonalizable matrix $W \in \mathbb{R}^{n\times n}$ 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 WG for a well conditioned random $n \times r_+$ matrix G.*

2. *Compute the rank revealing QR factorization of the matrix WG and output an orthogonal matrix basis $\widehat{U}$ of this matrix.*

The algorithm amounts to a single iteration of the Power Method [20], [54]. This is expected to be sufficient where the matrix W has a strongly dominant eigenspace. By virtue of Fact 2.1 we would benefit from choosing a random Toeplitz multiplier G where the matrix W belongs to the matrix algebra $\mathcal{A}_p$, generated by the companion matrix C_p of a polynomial $p(x)$. According to the study in [46] Gaussian random Toeplitz matrices are likely to be reasonably well conditioned under both standard Gaussian and uniform probability distribution.

Now assume a nonsingular matrix $\tilde{\phi}(M)$ with a dominated (rather than dominant) eigenspace $\mathcal{U}$. Then this is a dominant eigenspace of the matrix $(\tilde{\phi}(M))^{-1}$. We can apply Stages 2–4 of Algorithm 3.1 to this eigenspace or, alternatively, apply the following variation of Algorithm 3.1.

Algorithm 3.3. Dual reduction of the input size for eigen-solving.

INPUT, OUTPUT *and Stages 3 and 4 of* COMPUTATIONS *as in Algorithm 3.1.*

COMPUTATIONS:

1. *Compute a matrix function $\phi(M)$ having strongly dominated eigenspace $\mathcal{U}$.*
2. *Apply the Inverse Orthogonal Iteration [20, page 339] to the matrix $\phi(M)$ to output a matrix $\widehat{U}$ of full column rank whose range approximates the eigenspace $\mathcal{U}$. Output the matrix $\widehat{L} = \widehat{U}^{(I)} M \widehat{U}$.*

Remark 3.1. *Seeking a single eigenvalue of a matrix M and having performed Stage 1 of Algorithm 3.1 (resp. 3.3), we can apply the Power (resp. Inverse Power) Method (cf. [20, Sections 7.3.1 and 7.6.1], [9]) to approximate an eigenvector $\mathbf{v}$ of the matrix $\phi(M)$ in its dominant (resp. dominated) eigenspace $\mathcal{U}$. This eigenvector is shared with the matrix M by virtue of Theorem 2.3, and we can approximate the associated eigenvalue 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 [9], [47] and [49]. We can employ deflation or reapply our algorithm for other initial approximations (cf. our Section 5.5 and [26]) to approximate other eigenvalues of the matrix M.*

Remark 3.2. *In numerical implementation of the algorithms of this section one should compute a matrix basis L_+ for the dominant (resp. dominated) eigenspace $\mathcal{U}_+$ of the matrix $\phi_+(M)$ (resp. $\tilde{\phi}_+(M)$) such that $\mathcal{U}_+ \supseteq \mathcal{U}$ and has a dimension $r_+ \ge r$. Then the matrix L_+ has the size $r_+ \times r_+$ and shares r desired and $r_+ - r$ extraneous eigenvalues with the matrix M. For example, in numerical real eigen-solving the eigenspace $\mathcal{U}_+$ is associated with all real and nearly real eigenvalues of M, and having them approximated we can readily select among them the r real eigenvalues.*

In the next sections we describe some algorithms for computing the matrix functions $\phi(M)$ and $\tilde{\phi}(M)$ at Stages 1 of Algorithms 3.1 and 3.3.

4 The computation of the dominant eigenspaces by means of repeated squaring, shifts and inversions

4.1 Repeated squaring in the Frobenius algebra with simplified recovery of the eigenvalues

Theorem 2.3 for $\phi(M) = M^k$ implies that for a diagonalizable matrix M and sufficiently large integers k, the matrices M^k have dominant eigenspace $\mathcal{U}$ associated with the set of the absolutely largest eigenvalues of M. For a fixed or random real or complex shift s we can write $M_0 = M - sI$ and compute $M_0^{2^h}$ in h squarings,

$$M_{h+1} = a_h M_h^2,\ a_h \approx 1/||M_h||^2 \text{ for } h = 0, 1, \dots \tag{4.1}$$

Suppose M is a real diagonalizable matrix with simple eigenvalues and h is a reasonably large integer. Then with probability 1 the dominant eigenspace $\mathcal{U}$ of M_h has dimension 1 for random nonreal shifts s and has dimension 1 or 2 for a random real s. If the matrix M has a 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 M_h and a reasonably large integer h. As in the case of Algorithm 3.2, the column space of the product M_hG for a random well conditioned $n \times m$ matrix G is expected to approximate this eigenspace.

For $M = C_p$ we can follow [13] and apply the FFT-based algorithms that support Fact 2.1 to perform every squaring and every multiplication in $O(n \log n)$ flops. The bottleneck of that paper and its amelioration in [45] 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 [45] relieves some difficulties of [13] by employing approximations to the roots of $p'(x)$, $p''(x)$, etc., but these techniques are still too close to the symbolic recovery methods of the paper [13], which operates with polynomials and does not employ numerical linear algebra. In contrast Algorithms 3.1 and 3.3 reduce the computation of the r eigenvalues of a selected subset of the spectrum $\Lambda(M)$ to eigen-solving for the $r \times r$ matrix L, and this is simple where r is a small integer. Now replace M_0 in (4.1) by $M_0 = (M - \sigma I)^{-1}$ for a fixed complex σ. Then the above algorithms approximate the dominant eigenspace of the matrix M_h for a large integer h and the associated set of the eigenvalues of M, which are the nearest to the point σ. E.g., this is the set of the absolutely smallest eigenvalues where $\sigma = 0$. For $M = C_p$ we can alternatively write $M_0 = C_{p_{\mathrm{rev}}(x-\sigma)}$ in (4.1) to replace the inversion of the shifted companion matrix with Taylor's shift of the variable x of the polynomial $p(x)$ and the reversion of the order of its coefficients.

4.2 Approximation of the real eigenvalues: basic results

Next we map the complex plane to transform the real line into the unit circle $\{z : |z| = 1\}$ and then apply repeated squaring, which maps the unit circle into itself and sends the image of any nonreal eigenvalue of the input matrix towards 0 or ∞, thus ensuring desired isolation of the images.

Fact 4.1. *Write* $\lambda = u + v\sqrt{-1}$,

$$\mu = (\lambda + \sqrt{-1})(\lambda - \sqrt{-1})^{-1}, \ \beta_k = \frac{\sqrt{-1}(\mu^k + 1)}{\mu^k - 1} \tag{4.2}$$

for a positive integer k. *Then*

(a) $\beta_0 = \lambda = \frac{\sqrt{-1}(\mu+1)}{\mu-1}$,

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

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

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

Furthermore

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

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

Fact 4.1 implies that the transform $\lambda \to \mu$ maps the real line onto the unit circle $\mathcal{C}_1 = \{\mu : |\mu| = 1\}$. Powering of the value μ keeps this circle in place, whereas the transform $\mu^k \to \beta_k$ moves it back to the real line. Furthermore values $|\mu|^k$ converge to 0 for $|\mu| < 1$ and to $+\infty$ for $|\mu| > 1$ as $k \to \infty$. Therefore for large k the transform $\mu^k \to \beta_k$ sends the images of the nonreal values λ into some neubourhood of the values $\sqrt{-1}$ and $-\sqrt{-1}$. Then the transform $\beta_k \to \gamma_k = \beta_k^2 + 1$ sends these images into the neighborhood of the origin, whereas the real eigenvalues β_k are moved into

the real values $\gamma_k \geq 1$. This enables the desired domination of the images of the real eigenvalues of the matrix M over the images of its nonreal eigenvalues. We can recover the eigenvalues λ_k of the matrix M as soon as we approximate their eigenspaces shared with the eigenspaces associated with the eigenvalues γ_k of the matrices

$$Q_k = M_k^2 + I_n \tag{4.3}$$

where

$$P = (M + I_n\sqrt{-1})(M - I_n\sqrt{-1})^{-1}, \tag{4.4}$$

$$M_k = \sqrt{-1}(P^k + I_n)(P^k - I_n)^{-1}, \tag{4.5}$$

and in particular $M_1 = M$, whereas $2M_2 = M - M^{-1}$.

Corollary 4.1. *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 $\pm\sqrt{-1}$. Assume the equations of Fact 4.1 as well as equations (4.3)–(4.5). Furthermore write*

$$\beta_{j,k} = \frac{n_k(\lambda_j)}{d_k(\lambda_j)}, \quad n_k(\lambda_j) = \sum_{g=0}^{\lfloor k/2 \rfloor} (-1)^g \binom{k}{2g} \lambda_j^{k-2g}, \; d_k(\lambda_j) = \sum_{g=0}^{\lfloor k/2 \rfloor} (-1)^{g+1} \binom{k}{2g+1} \lambda_j^{k-2g-1},$$

and $\mu_j = (\lambda_j + \sqrt{-1})(\lambda_j - \sqrt{-1})^{-1}$ for $j = 1, \dots, s$. Then $M_k = n_k(M)(d_k(M))^{-1}$ where

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

and the matrices $Q_k = M_k^2 + I_n$ of (4.3) have the eigenpairs $\{\{\beta_{j,k}, \mathcal{U}_j\}, \; j = 1, \dots, s\}$ where $\beta_{j,k}$ are real and $\beta_k \geq 1$ if λ_j is real, $\beta_{j,k} \to 0$ as $k \to \infty$ unless λ_j is real.

4.3 Approximation of the real eigenvalues: the algorithm

The corollary suggests setting $\phi(M) = Q_k$ in Algorithm 3.1 where the integers k are sufficiently large. We can apply repeated squaring to compute high powers P^k. In numerical implementation we should apply scaling to avoid large norms $||P^k||_q$.

Below is an algorithm that implements this approach by using only two matrix inversions; this is much less than in iteration (5.5). The algorithm works for a large class of inputs M, although it can fail for harder inputs M, which have many real and nearly real eigenvalues, but also have some other nonreal eigenvalues. The heuristic choice

$$v = 0, \; w = 1, \; t \approx -\Re(\text{trace}(M)), \; a = \frac{t}{n}, \text{ and } \widehat{M} = M + tI_n \tag{4.6}$$

tends to push the values $|\mu|$ away from 1 on the average input, motivating application of the algorithm to the input matrix $\widehat{M}$ rather than M, although this shift can strongly push the value $|\mu|$ toward 1 for the worst case input. Note that trace(M) is a real value where M is a real matrix.

Algorithm 4.1. Mapping the real line onto the unit circle and repeated squaring

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

INITIALIZATION: *Fix sufficiently large tolerances τ and h_+, fix real a, t, v, and w and the matrix $\widehat{M}$ of (4.6).*

COMPUTATIONS:

1. *Compute the matrices* $P = (a\widehat{M} + I_n\sqrt{-1})(a\widehat{M} - I_n\sqrt{-1})^{-1}$ *(cf. Corollary 4.1) 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 4.1 for* $k = 2^{h_+}$.
3. *Apply Algorithm 3.2 to the matrix* $\phi = Q_k$ *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 3.2 fails, which would mean that the matrix* $\phi = Q_k$ *has no strongly dominant eigenspace of dimension* $r_+ < n$.

Remark 4.1. *We can compute the matrix* P^k *for a sufficiently large integer* $k = 2^{h_+}$ *to ensure isolation of the images of real and nearly real eigenvalues of* M *from the images of its other eigenvalues and then, as an alternative to the application of Algorithm 3.2 at Stage 3, we can apply the Rayleigh Quotient Iteration to the matrix* P^k *to approximate the associated eigenspace shared by the matrices* P^k *and* M.

Remark 4.2. *We can modify Stage 4 to compute an integer* h_+ *iteratively, according to a fixed policy: we can begin with a small* h_+*, then increase it, and reapply the algorithm if the computations fail. Alternatively we can estimate the integer* h_+ *a priori if we estimate the absolute values of all eigenvalues of the matrix* P *by computing its Gerschgörin discs [20, page 320], [54, page 39] (see also the end of the Appendix).*

4.4 Modification by using the Möbius transform

In an alternative iteration we begin in the same way as Algorithm 4.1 but interrupt repeated squaring by applying the scaled Möbius transform $P^k \to P^k + P^{-k}$ instead of the maps $P \to M_k$ of (4.5) and $M_k \to Q_k = M_k^2 + I_n$ of (4.3). 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 $[-2, 2]$, whereas for reasonably large integers k it moves the other eigenvalues into the exterior of the disc $D_{8/3}(0)$. (Namely the map $M \to P^k$ moves the nonreal eigenvalues of the matrix M towards 0 or ∞ and thus for reasonably large integers k moves them into the exterior of the annulus $\mathcal{A}_{1/3,3}(0) = \{x : 1/3 \leq |x| \leq 3\}$, which the scaled Möbius transform $P^k \to P^k + P^{-k}$ moves into the exterior of the disc $D_{8/3}(0)$.) Consequently by using the map $M \to P^k + P^{-k}$ we isolate from one another the two sets of the real and nonreal eigenvalues of the input companion matrix M. Then we make the eigenspace associated with real eigenvalues of the matrix M dominated or dominant simply by squaring reasonably many times the matrix $P^k + P^{-k}$ or its inverse, respectively, and then it remains to apply Algorithm 3.3 (respectively 3.1) to approximate these eigenvalues. The images of some real eigenvalues of the matrix M dominated by the images of other of them would be lost numerically due to rounding errors unless we apply orthogonalization or deflation. Next we prove the stated properties of this combination of the maps of Fact 4.1, repeated squaring, and the Möbius transform.

Fact 4.2. *(Cf. Fact 4.1 for* $a = 1$*.) Write*

$$\mu = (\lambda + \sqrt{-1})(\lambda - \sqrt{-1})^{-1}. \tag{4.7}$$

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* $\mu_2 = \frac{\lambda^4 - 6\lambda^2 + 1}{(\lambda^2+1)^2}$*.)*

Fact 4.3. *Assume* μ *of (4.7) and a nonnegative integer* k*. Then* $|\mu| = 1$ *and* $-2 \leq \mu^k + \mu^{-k} \leq 2$ *if* λ *is real, whereas* $|\mu^k + \mu^{-k}| \to \infty$ *as* $k \to \infty$ *otherwise.*

Corollary 4.2. *Assume 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 $\pm\sqrt{-1}$. By extending (4.4) and (4.7), write*

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

$$T_k = P^k + P^{-k} = \sum_{g=0}^{k} (-1)^g \binom{2k}{2g} M^{k-2g}(M^2+1)^{-k}, \tag{4.8}$$

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

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

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 a real value.

5 The computation of the dominant eigenspaces by approximating the matrix sign function

5.1 The matrix sign function: definition and basic properties

Definition 5.1. *For two real numbers $x \neq 0$ and y, the function* $\mathrm{sign}(x + y\sqrt{-1})$ *is equal to 1 if $x > 0$ and is equal to -1 if $x < 0$.*

Definition 5.2. *(See [23].) Let $A = ZJZ^{-1}$ be a Jordan canonical decomposition of an $n \times n$ matrix A where $J = \mathrm{diag}(J_-, J_+)$, 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* $\mathrm{sign}(A) = Z\,\mathrm{diag}(-I_p, I_q)Z^{-1}$. *Equivalently* $\mathrm{sign}(A) = A(A^2)^{-1/2}$ *or* $\mathrm{sign}(A) = \frac{2}{\pi} A \int_0^\infty (t^2 I_n + A^2)^{-1} dt$.

Definition 5.3. *Assume the matrices $A = ZJZ^{-1}$, J_- and J_+ above, except that $n = p + q + r$ and $J = \mathrm{diag}(J_-, J_0, J_+)$ 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 D_r, e.g., $D_r = O_{r,r}$, and define a* generalized matrix sign function $\mathrm{sign}(A)$ *by writing* $\mathrm{sign}(A) = Z\,\mathrm{diag}(-I_p, D_r\sqrt{-1}, I_q)Z^{-1}$.

We have the following simple but basic results.

Theorem 5.1. *Assume the generalized matrix sign function* $\mathrm{sign}(A)$ *defined for an $n \times n$ matrix $A = ZJZ^{-1}$. Then for some real $r \times r$ diagonal matrix D_r we have*

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

Corollary 5.1. *Under the assumptions of Theorem 5.1 the matrix $I_n - \mathrm{sign}(A)^2$ has dominant eigenspace of dimension r associated with the eigenvalues of the matrix A that lie on the imaginary axis $\mathcal{IA} = \{\lambda : \Re(\lambda) = 0\}$, whereas the matrices $I_n - \mathrm{sign}(A)$ (resp. $I_n + \mathrm{sign}(A)$) have dominant eigenspaces associated with the eigenvalues of A that either lie on the left (resp. right) of the axis $\mathcal{IA}$ or lie on this axis and have nonzero images in $I_n - \mathrm{sign}(A)$ (resp. $I_n + \mathrm{sign}(A)$).*

5.2 Eigen-solving by applying matrix sign approximation and Quad Tree construction

Having the matrices A and $\phi(A) = I_n - \text{sign}(A)$ (resp. $\phi(A) = I_n + \text{sign}(A)$) available, we can apply Algorithm 3.1 to approximate all eigenvalues of the matrix A that lie either on the axis $\mathcal{IA}$ or on the left (resp. right) from it. The computed square matrices L have dimensions p_+ and q_+, respectively, where $p \le p_+ \le p+r$ and $q \le q_+ \le q+r$. For $M = C_p$ this means splitting out a degree factor of the polynomial $p(x)$ having degree p_+ or q_+. If this degree is large, we are likely to see dramatic growth of the coefficients, e.g., 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 problem does not arise, however, as long as we work with matrices and approximate the eigenspaces. The subdivision techniques (cf. [41]) enable us to deal with matrices whose sizes are decreased recursively, and we can stop when their eigenvalues are the roots of the small degree factors of the polynomial $p(x)$, and so the coefficients of these factors are of the same order of magnitude as their roots. The approach relies on the following simple fact.

Fact 5.1. *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 the matrix A associated with the eigenvalues lying in a selected region on the complex plane bounded by straight lines, e.g., in any rectangle. In particular this supports the search policy widely known as *Quad Tree Construction*, proposed by H. Weyl in 1924 for polynomial root-finding. Strengthened by some modern techniques of numerical computing, Weyl's algorithm is practically promising and supports the record Boolean complexity estimates for approximating a single root of a univariate polynomial [41]. By including matrix inversions into these computations, we define the eigenvalue regions bounded by straight lines, their segments, circles and their arcs.

5.3 Iterative algorithms for computing the matrix sign function and their convergence

[23, equations (6.17)–(6.20)] define effective iterative algorithms for approximating the square root function $B^{1/2}$. One can readily extend them to approximating the matrix sign function $\text{sign}(A) = A(A^2)^{-1/2}$. [23, Chapter 5] presents a number of effective iterative algorithms devised directly for the matrix sign function. Among them we recall Newton's iteration

$$N_0 = A,\ N_{i+1} = 0.5(N_i + \alpha_i\ N_i^{-1}),\ i = 0, 1, \dots, \tag{5.1}$$

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

$$N_0 = A,\ N_{i+1} = (15I_n - 10N_i^2 + 3N_i^4)N_i/8,\ i = 0, 1, \dots \tag{5.2}$$

Theorem 2.3 implies the following simple corollary.

Corollary 5.2. *Assume iterations (5.1) and (5.2) where neither of the matrices N_i is singular. Let $\lambda = \lambda^{(0)}$ denote an eigenvalue of the matrix N_0 and define*

$$\lambda^{(i+1)} = (\lambda^{(i)} + (\lambda^{(i)})^{-1})/2 \text{ for } i = 0, 1, \dots, \tag{5.3}$$

$$\lambda^{(i+1)} = \lambda^{(i)}(15 - 10(\lambda^{(i)})^2 + 3(\lambda^{(i)})^4)/8,\ i = 0, 1, \dots \tag{5.4}$$

Then $\lambda^{(i)} \in \Lambda(N_i)$ for $i = 1, 2, \dots$ provided the pairs $\{N_i, \lambda^{(i)}\}$ are defined by the pairs of equations (5.1), (5.3) or (5.2), (5.4), respectively.

Corollary 5.3. *In iterations (5.3) and (5.4) the images $\lambda^{(i)}$ of an eigenvalue λ of the matrix N_0 for all i lie on the imaginary axis $\mathcal{IA}$ if so does λ.*

By virtue of the following theorems, the sequences $\{\lambda^{(0)}, \lambda^{(1)}, \dots\}$ defined by equations (5.3) and (5.4) converge to ± 1 exponentially fast right from the start. The convergence is quadratic for sequence (5.3) where $\Re(\lambda) \neq 0$ and cubic for sequence (5.4) where $|\lambda - \text{sign}(\lambda)| \leq 1/2$.

Theorem 5.2. *(See [23], [12, page 500].) Write $\lambda = \lambda^{(0)}$, $\delta = \text{sign}(\lambda)$ and $\gamma = |\frac{\lambda-\delta}{\lambda+\delta}|$. Assume (5.3) and $\Re(\lambda) \neq 0$. Then $|\lambda^{(i)} - \delta| \leq \frac{2\gamma^{2^i}}{1-\gamma^{2^i}}$ for $i = 0, 1, \dots$.*

Theorem 5.3. *Write $\delta_i = \text{sign}(\lambda^{(i)})$ and $\gamma_i = |\lambda^{(i)} - \delta_i|$ for $i = 0, 1, \dots$. Assume (5.4) and $\gamma_0 \leq 1/2$. Then $\gamma_i \leq \frac{32}{113}(\frac{113}{128})^{3^i}$ for $i = 1, 2, \dots$*

Proof. Complete the proof of [12, Proposition 4.1] by using the bound $\gamma_0 \leq 0.5$. First verify that $\gamma_{i+1} = \gamma_i^3 |3(\lambda^{(i)})^2 + 9\lambda^{(i)} + 8|/8$ and therefore $\gamma_{i+1} \leq \frac{113}{32}\gamma_i^3$ for $i = 0, 1, \dots$. Now the claimed bounds follow by induction on i because $\gamma_0 \leq 1/2$. □

5.4 Real versions of Newton's and Padé's iterations

Having the matrix $F(A) = I_n - \text{sign}(A)^2$ available, we can apply Algorithm 3.1 to approximate the eigenvalues of the matrix A that lie on the axis $\mathcal{IA}$, and we can devise real eigen-solvers for a real $n \times n$ matrix M, based on applying these techniques to the matrix $A = M\sqrt{-1}$. Next we modify this approach a little, to avoid involving nonreal values. We substitute $N_0 = M$ in lieu of $N_0 = A$ into matrix sign iterations (5.1) and (5.2) and equivalently rewrite them as follows,

$$N_0 = M,\ N_{i+1} = 0.5(N_i - N_i^{-1}) \text{ for } i = 0, 1, \dots, \tag{5.5}$$

$$N_0 = M,\ N_{i+1} = -(3N_i^5 + 10N_i^3 + 15N_i)/8 \text{ for } i = 0, 1, \dots. \tag{5.6}$$

The matrices N_i and the images $\lambda^{(i)}$ of every real eigenvalue λ of M are real for all i, whereas the results of Theorems 5.2 and 5.3 are immediately extended. The images of every nonreal point λ converge to the complex point $\text{sign}(\Im(\lambda))\sqrt{-1}$ with quadratic rate under (5.5) if $\Re(\lambda) \neq 0$ and with cubic rate under (5.6) if $\lambda \in \mathcal{D}_{1/2}(\text{sign}(\Im(\lambda))\sqrt{-1})$. Under the maps $M \to I_n + N_i^2$ for the matrices N_i of the above iterations, the images $1 + (\lambda^{(i)})^2$ of nonreal eigenvalues λ of the matrix M converge to 0 as long as the iteration is initiated in its basin of convergence, whereas the images of a real point λ are real and are at least 1 for all i. Thus for sufficiently large integers i we yield strong domination of the eigenspace of the matrix N_i associated with the images of the real eigenvalues of the matrix M.

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

Iteration (5.5) fails where for some integer i the matrix N_i is singular or nearly singular, that is has an eigenvalue equal to 0 or lying near 0, but then we can approximate this eigenvalue by applying the Rayleigh Quotient Iteration [20, Section 8.2.3], [9] or the Inverse Orthogonal Iteration [20, page 339].

If we seek other real eigenvalues as well, we can deflate the matrix M and apply Algorithm 3.1 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 $-r \leq \rho_i \leq r$ for a positive r. We choose the value r reasonably small and then can expect to avoid degeneracy and, by virtue of Theorems 5.2 and 5.3, to 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 the real eigenvalues.

5.6 Controlling the norms in the [2/0] Padé iterations

We have no singularity problem with iteration (5.6), but have numerical problems where the norms $||N_i||$ grow large. If the nonreal eigenvalues of the matrix N_0 lie in the union of the two discs $\mathcal{D}_{1/2}(\pm\sqrt{-1})$, then their images also stay there by virtue of a simple extension of Theorem 5.3, 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 the matrix M have been mapped into the union of the two discs $\mathcal{D}_{y_i}(\pm\sqrt{-1})$ for $0 < y_i < 0.1$. (One or two steps (5.6) move every $\mu \in \mathcal{D}_{1/2}(\pm\sqrt{-1})$ into the discs $\mathcal{D}_{y_i}(\pm\sqrt{-1})$, cf. Theorem 5.3.) Then the transformation $N_i \rightarrow N_i(N_i^2 + 2I_n)^{-1}$ 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 $w_i \leq \frac{1+y_i}{1-2y_i-y_i^2}$. 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 (5.6) would more than compensate for such a minor dilation of the discs $\mathcal{D}_{y_i}(\pm\sqrt{-1})$ (see Theorem 5.3).

5.7 Moving real eigenvalues into Padé's basin of convergence

Padé's iteration (5.6) is attractive because it avoids matrix inversions and has cubic rate of convergence, but it has a quite narrow basin of convergence, given by the union of the discs $\mathcal{D}_{1/2}(\pm\sqrt{-1})$. We can readily extend the maps $M \rightarrow P^k$ for the matrix P of (4.4), however, to move all real eigenvalues of an input matrix M into this basin. Indeed for sufficiently large integers k this map moves all nonreal eigenvalues of the matrix M towards the points 0 and ∞, while sending the real eigenvalues into the unit circle $\{z : \ |z| = 1\}$. The maps $P^k \rightarrow 0.1\ T_k \pm \sqrt{-1}\ I$ for $T_k = P^k + P^{-k}$ moves this unit circle into the discs $D_{0.2} \pm \sqrt{-1}$, both lying in the basin of convergence of Padé's iteration (5.6), whereas this map moves the images of the nonreal eigenvalues of the input matrix M towards ∞, that is keeps them outside this basin for reasonably large integers k.

We can estimate the integer $k = 2^{h_+}$ supporting the transforms into that basin if we estimate the absolute values of all eigenvalues of the matrix P. Towards this goal we can employ Gerschgörin discs [20, page 320], [54, page 39] (see also the end of the Appendix).

6 Numerical tests

We performed a series of numerical tests in the Graduate Center of the City University of New York using a Dell server with a dual core 1.86 GHz Xeon processor and 2G memory running Windows Server 2003 R2. The test Fortran code was compiled with the GNU gfortran compiler within the Cygwin environment. We generated random numbers with the random_number intrinsic Fortran function assuming the uniform probability distribution over the range $\{x : \ 0 \leq x < 1\}$. To shift to the range $\{y : \ b \leq y \leq a + b\}$ for fixed real a and b, we applied the linear transform $x \rightarrow y = ax + b$.

We tested our algorithms for the approximation of the eigenvalues of $n \times n$ companion matrix C_p and of the shifted matrix $C_p - sI_n$ defined by polynomials $p(x)$ with random real coefficients for $n = 64, 128, 256$ and by random real s. For each class of matrices, each input size and each iterative algorithm we generated 100 input instances and run 100 tests. Our tables show the minimum, maximum, and average (mean) numbers of iteration loops in these runs (until convergence) as well as the standard deviations in the columns marked by "**min**", "**max**", "**mean**", and "**std**", respectively. We applied repeated squaring of Section 4 to the matrix $C_p - sI$, where we used shifts s because polynomials $p(x)$ with random real coefficients tend to have all roots near the circle $\mathcal{C}_1(0)$ and consequently repeated squaring of C_p advances towards eigen-solving very slowly. We applied real Newton's iteration (5.5) to approximate the matrix sign function for the matrix C_p using no shifts. Then we applied Algorithm 3.1 to approximate real eigenvalues.

In both groups of the tests we output roots with at least four correct decimals. In our next group of tests we output roots with at least three correct decimals. In these tests we applied real Padé iteration (5.6) without stabilization to the matrices produced by five Newton's steps (5.5). Table 6.1 displays the results of our tests of repeated squaring of Section 4. The first three lines show the dimension of the output subspace and the matrix L. The next three lines show the number of squarings performed until convergence. Table 6.2 displays the number of Newton's steps (5.5) performed until convergence.

Table 6.4 covers the tests where we first performed five Newton's steps (5.5) followed by sufficiently many Padé steps (5.6) required for convergence. The first three lines of the table show the number of the Padé steps. The next three lines display the percent of the real roots of the polynomials $p(x)$ that the algorithm computed with at least three correct decimals (compared to

the overall number of the real eigenvalues of L). The next three lines show the increased percent of computed roots when we refined the crude approximations by means of Rayleigh Quotient iteration. The iteration rapidly converged from all these initial approximations but in many cases to the same roots from distinct initial points.

Table 6.1: Repeated Squaring

n	**dimension/squarings**	**min**	**max**	**mean**	**std**
64	dimension	1	10	5.31	2.79
128	dimension	1	10	3.69	2.51
256	dimension	1	10	4.25	2.67
64	squarings	6	10	7.33	0.83
128	squarings	5	10	7.37	1.16
256	squarings	5	11	7.13	1.17

Table 6.2: Newton's iteration (5.5).

n	**min**	**max**	**mean**	**std**
64	7	11	8.25	0.89
128	8	11	9.30	0.98
256	9	13	10.22	0.88

Table 6.3: 5 N-steps (5.5) + P-steps (5.6)

n	**P-steps or %**	**min**	**max**	**mean**	**std**
64	P-steps	1	4	2.17	0.67
128	P-steps	1	4	2.05	0.63
256	P-steps	1	3	1.99	0.58
64	% w/o RQ steps	0	100	64	28
128	% w/o RQ steps	0	100	39	24
256	% w/o RQ steps	0	100	35	20
64	% w/RQ steps	0	100	89	19
128	% w/RQ steps	0	100	74	26
256	% w/RQ steps	0	100	75	24

7 Conclusions

While presenting a number of promising approaches we have only partly developed them to demonstrate their power and to motivate further research efforts. In some cases we skipped even some natural modifications. For example, recall Newton's iteration (5.1) for computing matrix sign function. If the norms of its two terms have different orders of magnitude, then the iteration degenerates due to rounding errors, and its convergence slows down. To avoid this problem we can apply scaling, that is, modify the iteration as follows,

$$N_0 = A,\ N_{i+1} = 0.5(N_i + \alpha_i\ N_i^{-1}),\ \alpha_i = ||N_i||/||N_i^{-1}||,\ i = 0, 1, \dots, \tag{7.1}$$

and similarly we can modify the variant (5.5) of the iteration for real eigen-solving,

Table 6.4: 5 N-steps (5.5) + P-steps (5.6)

n	**P-steps or %**	**min**	**max**	**mean**	**std**
64	P-steps	2	8	4.26	1.19
128	P-steps	2	10	4.20	1.23
256	P-steps	2	6	4.24	1.22
64	% w/o RQ steps	0	100	67	26
128	% w/o RQ steps	0	100	43	24
256	% w/o RQ steps	0	100	33	23
64	% w/RQ steps	0	100	87	21.3
128	% w/RQ steps	0	100	87	20.5
256	% w/RQ steps	0	100	88	21.5

$$N_0 = M,\ N_{i+1} = 0.5(N_i - \alpha_i N_i^{-1}) \text{ for } \alpha_i = ||N_i||/||N_i^{-1}|| \text{ and } i = 0, 1, \dots. \tag{7.2}$$

Empirically this scaling technique substantially improves convergence, which is an example of great many potential refinements of our algorithms. One can expect to see new advances of our approaches, e.g., based on more intricate maps of the complex plane. Another potential resource of further progress is the combination with other matrix eigen-solvers and polynomial root-finders, for example, a variant of the Lanczos algorithm for real eigen-solving, the Rayleigh Quotient iteration, and the subdivision and continued fraction methods of polynomial root-finding (see [17], [18], [27], [33], [55], [60], and the bibliography therein). Various symbolic techniques can supply auxiliary information for our computations (e.g., the number of real roots and their bounds) and can handle the inputs that are hard for our numerical treatment.

Appendix

A Variations that involve the characteristic polynomial

In the case where $M = C_p$ is the companion matrix of a polynomial $p(x)$, the monic characteristic polynomial $c_P(x)$ for the matrix P of (4.4) equals $\gamma(x-1)^n p(\frac{x+1}{x-1}\frac{\sqrt{-1}}{a}) = \gamma(x-1)^n p(1-\frac{2}{x-1}\frac{\sqrt{-1}}{a})$ for a scalar γ. We can obtain its coefficients by performing two shifts of the variable (see [43, Chapter 2] on this operation) and the single reversion of the polynomial coefficients. When this is done we can replace k repeated squarings of the matrix P with k steps of the Dandelin's root-squaring iteration, also attributed to some later works by Lobachevsky and Gräffe (see [21]),

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

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 (A.1) essentially amounts to polynomial multiplication and can be performed in $O(n \log n)$ flops. One can improve numerical stability by applying modifications in [35], which use order of n^2 flops per iteration. 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 of Section 4.4 to the matrix C_{p_k}, the companion matrix of this polynomial, yields its roots lying on the circle $\mathcal{C}_1$ (they are the eigenvalues of the matrix C_{p_k}). From these roots we can recover the roots μ of the polynomial $c_P(x) = p_0(x)$ by means of the descending techniques of [37] (applied also in [38], [39], [44], and [49, Stage 8 of Algorithm 9.1]), and then can recover the real roots λ of the polynomial $p(x)$ from the values μ by applying the expression in part (a) of Fact 4.1. In this approach we can readily approximate the eigenvalues of the matrix P from the origin as the root radii of the characteristic polynomial $c_P(x) = \det(xI_n - P)$. Indeed as long as we are given the coefficients we can approximate

all the root radii with relative errors of at most 1% by using $O(n \log n)$ flops (see [4], [8], [41], [44], [52]).

Remark A.1. *Having isolated the roots of $p_k(x)$ on the circle $\mathcal{C}_1$ from its other roots, we can apply the algorithms of [29], [37], [38], [44], [52] to split out the factor $f(x)$ sharing with the polynomial precisely all the roots that lie on the circle $\mathcal{C}_1$. Then these roots can be moved into the real line and then readily approximated based on the Laguerre or modified Laguerre algorithms [36], [25], [15], [16], and [61]. Numerical problems can be caused by potentially dramatic growth of the coefficients of the polynomial $p_k(x)$ in the transition to the factor $f(x)$ unless its degree is small.*

Acknowledgements: Our research has been supported by NSF Grant CCF–1116736 and PSC CUNY Awards 64512–0042 and 65792–0043.

References

[1] J. L. Aurentz, R. Vandebril, and D. S. Watkins, Fast computation of the zeros of a polynomial via factorization of the companion matrix, *SIAM J. Sci. Comput.*, **35, 1**, A255–A269, 2013.

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

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

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

[5] D. A. Bini, P. Boito, Y. Eidelman, L. Gemignani, I. Gohberg, A Fast Implicit QR Algorithm for Companion Matrices, *Linear Algebra and Its Applications*, **432**, 2006–2031, 2010.

[6] Z. Bai, J. Demmel, J. Dongarra, A. Ruhe, H. van der Vorst, editors, *Templates for the Solution of Algebraic Eigenvalue Problems: A Practical Guide*, SIAM, Philadelphia, 2000.

[7] D. A. Bini, F. Daddi, and L. Gemignani, On the Shifted QR Iteration Applied to Companion Matrices, *Electronic Transactions on Numerical Analysis (ETNA)*, **18**, 137–152, 2004.

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

[9] 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.

[10] D. A. Bini, L. Gemignani, V. Y. Pan, 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).

[11] D. A. Bini, L. Gemignani, V. Y. Pan, Improved Initialization of the Accelerated and Robust QR-like Polynomial Root-finding, *Electronic Transactions on Numerical Analysis* **17**, 195–205, 2004. Proc. version in CASC'2004.

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

[13] 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.

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

[15] Q. Du, M. Jin, T. Y. Li, Z. Zeng, Quasi-Laguerre Iteration in Solving Symmetric Tridiagonal Eigenvalue Problems, *SIAM J. Sci. Computing*, **17**, **6**, 1347–1368, 1996.

[16] Q. Du, M. Jin, T. Y. Li, Z. Zeng, The Quasi-Laguerre Iteration, *Math. of Computation*, **66**, **217**, 345–361, 1997.

[17] A. Eigenwillig, V. Sharma, C. K. Yap, Almost Tight Recursion Tree Bounds for the Descartes Method, *Proc. Int. Symp. on Symbolic and Algebraic Computation (ISSAC 2006)*, 71–78, ACM Press, New York, 2006.

[18] I. Z. Emiris, B. Mourrain, E. Tsigaridas, Real Algebraic Numbers: Complexity Analysis and Experimentation, in *RELIABLE IMPLEMENTATIONS OF REAL NUMBER ALGORITHMS: THEORY AND PRACTICE, LNCS*, **5045**, 57–82, Springer, 2008 (also available in www.inria.fr/rrrt/rr-5897.html).

[19] M. Gu, S. C. Eisenstat, Efficient Algorithms for Computing a Strong Rank-Revealing QR Factorization, *SIAM Journal on Scientific Computing*, **17**, 848–869, 1996.

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

[21] A. S. Householder, Dandelin, Lobachevskii, or Graeffe, *American Mathematical Monthly* **66**, 464–466, 1959.

[22] A. S. Householder, Generalization of an Algorithm by Sebastiao e Silva, *Numerische Math.,*, **16**, 375–382, 1971.

[23] N. J. Higham, *Functions of Matrices: Theory and Computations*, SIAM, Philadelphia, 2008.

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

[25] E. Hansen, M. Patrick, J. Rusnack, Some Modification of Laguerre's Method, *BIT*, **17**, 409–417, 1977.

[26] J. Hubbard, D. Schleicher, S. Sutherland, How to Find All Roots of Complex Polynomials by Newton's Method, *Invent. Math.*, **146**, 1–33, 2001.

[27] M. Hemmer, E.P. Tsigaridas, Z. Zafeirakopoulos, I. Z. Emiris, M. I. Karavelas, B. Mourrain, Experimental Evaluation and Cross-Benchmarking of Univariate Real Solvers, in *Proc. International Symposium on Symbolic-Numerical Computations*, (Kyoto, Japan, August 2009), (edited by Hiroshi Kai and Hiroshi Sekigawa), pp.105–113, ACM Press, New York, 2009.

[28] R. J. Johnston, Gershgorin Theorems for Partitioned Matrices, *Linear Algebra and Its Applications*, **4**, 205–220, 1971.

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

[30] J.M. McNamee, A 2002 Update of the Supplementary Bibliography on Roots of Polynomials, *J. of Computational and Applied Math.* **142**, 433-434, 2002; also at web-site `www.yorku.ca/~mcnamee/`

[31] J.M. McNamee, *Numerical Methods for Roots of Polynomials (Part 1)*, Elsevier, Amsterdam, 2007.

[32] J.M. McNamee and V.Y. Pan, *Numerical Methods for Roots of Polynomials, Part 2*, Elsevier, 2013.

[33] K. Mehlhorn, M. Sagraloff, A Deterministic Algorithm for Isolating Real Roots of a Real Polynomial, *J. of Symbolic Computation* **46, 1**, 70–90, 2011.

[34] H. I. Medley, K. S. Varga, On Smallest Isolated Gerschgorin Disks for Eigenvalues, *Numerische Mathematik*, **11**, 361–369, 1968.

[35] G. Malajovich, J. P. Zubelli, On the Geometry of Graeffe Iteration, *J. of Complexity*, **17, 3**, 541-573, 2001.

[36] B. Parlett, Laguerre's Method Applied to the Matrix Eigenvalue Problem, *Math. of Computation*, **18**, 464–485, 1964.

[37] V. Y. Pan, Optimal (up to Polylog Factors) Sequential and Parallel Algorithms for Approximating Complex Polynomial Zeros, *Proc. 27th Ann. ACM Symp. on Theory of Computing*, 741–750, ACM Press, New York, 1995.

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

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

[40] V. Y. Pan, Solving Polynomials with Computers, *American Scientist*, **86**, January–February 1998. Available via

http://comet.lehman.cuny.edu/vpan/research/publications

[41] 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.

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

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

[44] 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.

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

[46] 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

[47] 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.

[48] V. Y. Pan, E. P. Tsigaridas, On the Boolean Complexity of the Real Root Refinement, preprint 2013, Proc. version in in *Proc. Intern. Symposium on Symbolic and Algebraic Computation (ISSAC 2013)*, (M. Kauers ed.), 299–306, Boston, MA, June 2013, ACM Press, New York, 2013.

[49] 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.

[50] J. Sebastiao e Silva, Sur une méthode d'approximation semblable a celle de Graeffe, *Portugal Math.*, **2**, 271–279, 1941.

[51] G. W. Stewart, On the Convergence of Sebastiao E Silva's Method for Finding a Zero of a Polynomial, *SIAM Review*, **12**, 458–460, 1970.

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

[53] G. W. Stewart, *Matrix Algorithms, Vol I: Basic Decompositions*, SIAM, Philadelphia, 1998.

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

[55] E. P. Tsigaridas, I. Z. Emiris, Univariate Polynomial Real Root Isolation: Continued Fractions Revisited, *ESA'06 Proceedings of the 14th Conference on Annual European Symposium*, Zurich, 2006, *LNCS*, **4168**, 817–828, Springer, London, 2006.

[56] R. S. Varga, Minimal Gerschgorin Sets for Partitioned Matrices, *SIAM J. on Numericaql Analysis*, **7**, 493–507, 1970.

[57] M. Van Barel, R. Vandebril, P. Van Dooren, K. Frederix, Implicit Double Shift QR-algorithm for Companion Matrices, *Numerische Mathematik* **116**, **2**, 177–212, 2010.

[58] D. S. Watkins, *Fundamentals of Matrix Computations*, Wiley, New York, 2002 (second edition).

[59] D. S. Watkins, *The Matrix Eigenvalue Problem: GR and Krylov Subspace Methods*, SIAM, Philadelphia, PA, 2007.

[60] C. Yap, M. Sagraloff, A Simple but Exact and Efficient Algorithm for Complex Root Isolation, *Proc. of International Symp. on Symbolic and Algebraic Computation (ISSAC '11)*, San Jose, California, June 2011 (edited by A. Leykin), 353–360, ACM Press, New York, 2011.

[61] X. Zou, Analysis of the Quasi-Laguerre Method, *Numerische Math.*, **82**, 491–519, 1999.

[62] P. Zhlobich, Differential qd Algorithm with Shifts for Rank-Structured Matrices. *SIAM J. on Matrix Analysis and Applications*, 33(4), 1153–1171, 2012.