COMPUTING FUNDAMENTAL MATRIX DECOMPOSITIONS ACCURATELY VIA THE MATRIX SIGN FUNCTION IN TWO ITERATIONS: THE POWER OF ZOLOTAREV'S FUNCTIONS

YUJI NAKATSUKASA* AND ROLAND W. FREUND†

Abstract. The symmetric eigenvalue decomposition and the singular value decomposition (SVD) are fundamental matrix decompositions with many applications. Conventional algorithms for computing these decompositions are suboptimal in view of recent trends in computer architectures, which require minimizing communication together with arithmetic costs. Spectral divide-and-conquer algorithms, which recursively decouple the problem into two smaller subproblems, can achieve both requirements. Such algorithms can be constructed with the polar decomposition playing two key roles: it forms a bridge between the symmetric eigendecomposition and the SVD, and its connection to the matrix sign function naturally leads to spectral-decoupling. For computing the polar decomposition, the scaled Newton and QDWH iterations are two of the most popular algorithms, as they are backward stable and converge in at most nine and six iterations, respectively. Following this framework, we develop a higher-order variant of the QDWH iteration for the polar decomposition. The key idea of this algorithm comes from approximation theory: we use the best rational approximant for the scalar sign function due to Zolotarev in 1877. The algorithm exploits the extraordinary property enjoyed by the sign function that a high-degree Zolotarev function (best rational approximant) can be obtained by appropriately composing low-degree Zolotarev functions. This lets the algorithm converge in just *two* iterations in double-precision arithmetic, with the whopping rate of convergence *seventeen*. The resulting algorithms for the symmetric eigendecompositions and the SVD have higher arithmetic costs than the QDWH-based algorithms, but are better-suited for parallel computing and exhibit excellent numerical backward stability.

Key words. Zolotarev function, rational approximation of the sign function, symmetric eigenvalue problem, singular value decomposition, polar decomposition, minimizing communication

AMS subject classifications. 15A23, 65F15, 65F30, 65G50

1. Introduction.

Every matrix $A \in \mathbb{C}^{m\times n}$ with $m \geq n$ can be written in the form

$$A = U\Sigma V^*, \tag{1.1}$$

where $U \in \mathbb{C}^{m\times n}$ and $V \in \mathbb{C}^{n\times n}$ are matrices with orthonormal columns, i.e., $U^*U = I_n$ and $V^*V = I_n$, and $\Sigma \in \mathbb{R}^{n\times n}$ is a diagonal matrix with nonnegative diagonal entries. Here and throughout the paper, the superscript "$*$" denotes the complex conjugate. The factorization (1.1) is called the *singular value decomposition* (SVD) and the diagonal entries of Σ the *singular values* of A; see, e.g., [22, § 2.4]. The SVD is the most versatile decomposition of a general matrix. We note that the factorization (1.1) is sometimes called the *thin* SVD to distinguish it from the factorization of the same form (1.1), but with $U \in \mathbb{C}^{m\times m}$ and $\Sigma \in \mathbb{R}^{m\times n}$.

One of the many applications of the SVD is the polar decomposition. Every matrix $A \in \mathbb{C}^{m\times n}$ with $m \geq n$ can be written in the form

$$A = U_p H, \tag{1.2}$$

*Department of Mathematical Informatics, University of Tokyo, Tokyo 113-8656, Japan (e-mail: nakatsukasa@mist.i.u-tokyo.ac.jp). Supported by JSPS Grants-in-Aid for Scientific Research No. 26870149. Part of this work was done while this author was a postdoc at the University of Manchester and supported by EPSRC grant EP/I005293/1.

†Department of Mathematics, University of California at Davis, One Shields Avenue, Davis, CA 95616 (e-mail: freund@math.ucdavis.edu)

where $U_p \in \mathbb{C}^{m\times n}$ has orthonormal columns and $H \in \mathbb{C}^{n\times n}$ is Hermitian positive semidefinite. The factorization (1.2) is called the *polar decomposition*, and U_p and H the unitary (or orthogonal if A is real) and Hermitian *polar factors* of A, respectively. The existence of the polar decomposition (1.2) easily follows by writing the SVD of A as $A = U\Sigma V^* = (UV^*)(V\Sigma V^*) = U_pH$ where $U_p = UV^*$ and $H = V\Sigma V^*$. We note that the Hermitian polar factor H is unique for any A. Furthermore, if $A \in \mathbb{C}^{m\times n}$ has full column rank n, then the unitary polar factor U_p is also unique and H is positive definite.

The polar decomposition (1.2) is used in various situations, such as projection onto the Stiefel manifold [38], the orthogonal Procrustes problem [16, 24], and reorthogonalization of inexact solutions [44]; see [31, § 2.6] for more applications. In this paper, we are particularly motivated by the use of the polar decomposition in spectral divide-and-conquer algorithms for computing the SVD $A = U\Sigma V^*$ of general matrices and the eigenvalue decomposition $A = V\Lambda V^*$ of Hermitian (and real symmetric) matrices $A = A^*$. Here, V is a unitary matrix and Λ is a diagonal matrix with the eigenvalues of A as its diagonal entries, which are all real.

The standard algorithm for computing the symmetric eigendecomposition of dense matrices [22, Ch. 8],[50, Ch. 7,8] proceeds by first reducing the matrix into tridiagonal form, then computing a tridiagonal eigenvalue decomposition. At the end of this introduction we give pictorial descriptions of this algorithm, along with the standard SVD algorithms and algorithms based on spectral divide-and-conquer.

There has been much recent progress on designing algorithms that reduce communication in addition to the arithmetic cost [8], so that they are well-suited for parallel computing. Unfortunately, the standard eigendecomposition algorithm for does not minimize communication in a naive implementation, with the initial reduction being a bottleneck [7]. Recent attempts [9] modify the reduction stage for the symmetric eigendecomposition to reduce communication, but its implementation details are unavailable, and involve a significant amount of tuning that depends on the architecture.

A different approach based on spectral divide-and-conquer using the polar decomposition is pursued in [45, § 3,4], [48]. An advantage of this approach is that they can be implemented in a communication-minimizing manner using only widely available and highly optimized matrix operations, such as matrix multiplication, QR factorization, and Cholesky decomposition.

Much of the discussion carries over to the SVD. Standard SVD algorithms [22, § 8.6] (again for dense matrices; for approximating the SVD of large sparse matrices, see e.g. [5, 29]) first reduces the matrix $A \in \mathbb{C}^{m\times n}$ to bidiagonal form.

In a spectral divide-and-conquer approach, we first compute (again) the polar decomposition $A = U_pH$, and then obtain the symmetric (positive semidefinite) eigendecomposition $H = V\Sigma V^*$ by spectral divide-and-conquer. Then $A = (U_pV)\Sigma V^* = U\Sigma V^*$ is the SVD. Here the polar decomposition is used to reduce computing the SVD to computing the symmetric eigendecomposition.

We stress the fundamental role played by the computation of the polar decomposition $A = U_pH$ in a spectral divide-and-conquer method, both for the symmetric eigendecomposition and the SVD. As shown in [45], the arithmetic cost for the symmetric eigendecomposition is about 4/3 times that of computing $A = U_pH$, and about 7/3 for the SVD. Furthermore, the backward stability of the computed symmetric eigendecomposition and the SVD rests crucially on that of the polar decompositions [48].

Many iterations are known for computing the polar decomposition. The classical

scaled Newton iteration [31, § 8.6], $X_{k+1} = \frac{1}{2}(\mu_k X_k + \mu_k^{-1} X_k^{-1})$ with $X_0 = A$, is still a practical method for square nonsingular matrices, with several effective scaling strategies for choosing μ_k proposed [14, 33]. Other methods exist such as Newton-Schulz and the Padé family of methods; see [31, Ch. 8],[33] for details.

The approach in [45, 48] for computing the polar decomposition is to use the QDWH (QR-based dynamically weighted Halley) iteration developed in [46]. QDWH has the advantage over scaled Newton that QDWH is inverse-free and typically more accurate for large matrices, and requires a smaller number of iterations; in Section 3.7, we give a unified view of the two iterations. The analysis in [47] establishes backward stability of QDWH. This in turn implies the resulting algorithms for the eigendecomposition (QDWH-eig) and the SVD (QDWH-SVD) are numerically backward stable, and in fact they typically give better backward errors than standard algorithms [48]. Since QDWH uses only matrix multiplication, QR factorization, and Cholesky decomposition as building blocks, the resulting algorithms can be implemented in a communication-minimizing manner in the asymptotic sense [8]. The dominant cost of one QDWH iteration is in performing the QR factorization of matrices of the form $\left[\begin{smallmatrix} X \\ c_k I \end{smallmatrix}\right]$ for a scalar c_k. QDWH requires at most six iterations to converge for any practical matrix satisfying $\kappa_2(X) < u^{-1}$, where $u \approx 1.1 \times 10^{-16}$ is the unit roundoff in IEEE double-precision arithmetic [46]. The arithmetic operation costs for the resulting spectral divide-and-conquer algorithms QDWH-eig and QDWH-SVD are within a factor 3 of those for the standard algorithms [48].

However, the experiments in [48] indicate that the QDWH-based algorithms are still considerably slower on a sequential machine (by a factor between 2 and 4) than the fastest standard algorithm. Although the relative performance is expected to be better on parallel systems, and indeed a recent report [55] suggests QDWH-SVD can outperform standard methods on multicore-GPU systems, further improvements to the algorithm itself are therefore much desired. Since the spectral divide-and-conquer algorithms use the polar decomposition as fundamental building blocks, the design of an improved algorithm for computing $A = U_p H$ directly leads to improved spectral divide-and-conquer algorithms for the symmetric eigendecomposition and the SVD.

The main contribution of this paper is the design of Zolo-pd, an algorithm for the polar decomposition that requires just *two* iterations[1] for convergence in double-precision for any matrix A with $\kappa_2(A) \leq 10^{16}$. The crucial observation is that the mapping function for the singular values used by the QDWH iteration is exactly the type-$(3,2)$ best rational approximation of the scalar sign function. Then a natural idea is to consider its higher-order variant, i.e., type-$(2r+1, 2r)$ best rational approximations for general $r \geq 1$. The explicit solution of this rational approximation problem was given by Zolotarev in 1877. We show that the type-$((2r+1)^k, (2r+1)^k - 1)$ best approximation can be expressed as a composition of the form $R_k(\cdots(R_1(x))\cdots)$, where each R_i is a rational function of type $(2r+1, 2r)$. This observation yields an algorithm that requires only k matrix iterations to apply the type-$((2r+1)^k, (2r+1)^k - 1)$ best approximation to the singular values. The high-order rational approximations to the sign function $\mathrm{sign}(x)$ are so powerful that a type-$(m+1, m)$ best approximant on the interval $[10^{-15}, 1]$ with $m \geq 280$ has accuracy $\mathcal{O}(u)$. Since taking $r = 8$ yields a type $(289, 288)$ approximant with $k = 2$, this means just *two* iterations is enough to obtain

[1] The two-step convergence of Zolo-pd suggests that perhaps it should be regarded as a direct rather than an iterative algorithm: even though it is known [20, 21] that Abel's impossibility theorem implies that the exact polar decomposition cannot be computed in a finite number of arithmetic operations, an approximation correct to $\mathcal{O}(10^{-16})$ can be obtained.

$\mathcal{O}(u)$ accuracy on $[10^{-15}, 1]$. This is the rational function used by Zolo-pd, which therefore converges in two matrix iterations. The algorithm is still iterative (it has to be [20]), and the rate of convergence is as high as *seventeen*, which means one iteration reduces the error from ε to $\mathcal{O}(\varepsilon^{17})$. This is a significantly higher rate of convergence than for any other practical numerical algorithm that the authors are aware of. For example, the widely used Newton's method typically converges quadratically, i.e., its convergence rate is two, and cubically convergent methods with rate three are often regarded as very fast; examples include the QR algorithm for symmetric tridiagonal matrices [50] and QDWH.

It is often believed (and sometimes true) that higher-order iterations can cause numerical instability. However, with the help of a QR-based implementation, Zolo-pd exhibits excellent numerical stability comparable to that of QDWH, leading to stable algorithms also for computing the symmetric eigendecomposition (Zolo-eig) and the SVD (Zolo-SVD). Their experimental backward stability and orthogonality measure are comparable to QDWH-eig and QDWH-SVD, and are notably better than those of standard algorithms.

Using the partial fraction form of the type $(2r+1, 2r)$ rational function, each iteration of Zolo-pd involves computing the QR factorization $\left[\begin{smallmatrix} X \\ c_j I \end{smallmatrix}\right] = QR$ for distinct values of c_j, $j = 1, \ldots, r(\leq 8)$. Just like the QDWH-based algorithms, the Zolo-based algorithms require only matrix multiplications and QR and Cholesky factorizations, and thus minimize communication. The arithmetic cost of the Zolo-based algorithms is higher than the QDWH-based ones (by a factor < 3), so on a serial implementation Zolo-pd is slower than QDWH; this is reflected in our MATLAB experiments. Fortunately, the r QR factorizations can be performed independently, in which case the runtime per iteration is expected to be roughly the same as QDWH. Then, due to the faster convergence, Zolo-pd can be faster than QDWH by a factor up to 3. Indeed, in a parallel implementation, the arithmetic cost of Zolo-SVD along the critical path is comparable to that of the standard algorithms based on reduction to bidiagonal form. Zolo-based algorithms therefore have ideal properties for parallel computing, and our preliminary MATLAB experiments suggest that if properly parallelized, our algorithms can outperform standard algorithms in both speed and accuracy. This paper focuses on the mathematical foundations of the algorithm, and optimizing the performance on massively parallel systems is left as future work.

The structure of this paper is as follows. In the remainder of this section we review the matrix sign function and give pictorial descriptions of the algorithms. In Section 2, we revisit the QDWH iteration from the viewpoint of approximation theory. In Section 3 we develop its higher-order variant, in which we show that an appropriately composed Zolotarev function is another Zolotarev function. Section 4 discusses detailed algorithmic implementation issues such as a stable and parallelizable evaluation of the iteration. We then summarize the Zolo-based algorithms and compare them with other standard algorithms in Section 5. Numerical experiments with a sequential implementation are presented in Section 6.

Notation: $\sigma_1(A) \geq \cdots \geq \sigma_n(A)$ are the singular values of a rectangular matrix $A \in \mathbb{C}^{m\times n}$ with $m \geq n$, and $\sigma_{\max}(A) = \sigma_1(A)$ and $\sigma_{\min}(A) = \sigma_n(A)$, which is assumed positive unless otherwise stated. $\|A\|_2 = \sigma_{\max}(A)$ denotes the spectral norm and $\|A\|_F = (\sum_{i,j} |A_{ij}|^2)^{\frac{1}{2}}$ is the Frobenius norm. $\lambda_i(A)$ denotes the ith largest eigenvalue of a symmetric matrix A. $\kappa_2(A) = \sigma_{\max}(A)/\sigma_{\min}(A)$ is the condition number of A. To avoid confusion between the unitary polar factor and the singular value decomposition (SVD) of A, we always use subscripts U_p to denote the unitary

polar factor, while U denotes the matrix of left singular vectors. Hence for example $A = U_p H = U\Sigma V^*$. We denote by $\mathcal{P}_r$ the set of all polynomials P with real coefficients of degree at most r and by $\mathcal{R}_{r,s}$ the set of all rational functions $R = \frac{P}{Q}$ where $P \in \mathcal{P}_r$, $Q(\neq 0) \in \mathcal{P}_s$. We say that a rational function R is of type (r, s) if $R \in \mathcal{R}_{r,s}$.

We develop algorithms for complex matrices $A \in \mathbb{C}^{m\times n}$, but for $A \in \mathbb{R}^{m\times n}$ all the operations can be carried out using real arithmetic only, and for simplicity we call a matrix $A = A^*$ symmetric (instead of Hermitian). We assume the use of IEEE double-precision arithmetic, in which the unit roundoff $u = 2^{-53} \approx 1.1 \times 10^{-16}$, but minor modifications extends the discussion to higher (or lower) precision arithmetic. The polar decomposition is unique if A has full column rank, which we assume in most of the paper (see Section 5.3 for the rank-deficient case).

1.1. The matrix sign function. The scalar sign function $\mathrm{sign}(z)$ is defined for all complex numbers $z \in \mathbb{C}$ off the imaginary axis $\mathrm{Re}(z) = 0$ as follows: $\mathrm{sign}(z) = 1$ if $\mathrm{Re}(z) > 0$ and $\mathrm{sign}(z) = -1$ if $\mathrm{Re}(z) < 0$. The *matrix sign function* $\mathrm{sign}(A)$ is an extension of $\mathrm{sign}(z)$ to square matrices $A \in \mathbb{C}^{n\times n}$ that have no eigenvalues on the imaginary axis [42, 53]. Formally, $\mathrm{sign}(A)$ can be defined via the Jordan decomposition $A = ZJZ^{-1}$, where Z is nonsingular and J is a Jordan canonical form of A. The eigenvalues of A appear on the diagonal of J, and we can order the rows and columns of J such that the eigenvalues λ with $\mathrm{Re}(\lambda) > 0$ appear first. This means that J is of the form $J = \left[\begin{smallmatrix} J_+ & 0 \\ 0 & J_- \end{smallmatrix}\right]$ with $J_+ \in \mathbb{C}^{n_+\times n_+}$ and $J_- \in \mathbb{C}^{(n-n_+)\times(n-n_+)}$ corresponding to the eigenvalues of A with $\mathrm{Re}(\lambda) > 0$ and $\mathrm{Re}(\lambda) < 0$, respectively. The matrix sign function of A is then given by

$$\mathrm{sign}(A) = Z \begin{bmatrix} I_{n_+} & 0 \\ 0 & -I_{n-n_+} \end{bmatrix} Z^{-1}. \tag{1.3}$$

The matrix sign function is related to the polar decomposition. In particular, for symmetric matrices $A = A^*$, the polar factor U_p of A is equal to $\mathrm{sign}(A)$; see [34].

Of particular relevance to this paper is the role of $\mathrm{sign}(A)$ in solving eigenvalue problems by a spectral divide-and-conquer process, see e.g. [4, 6, 10, 36]. Invariant subspaces can be computed via $\mathrm{sign}(A)$ because $\frac{1}{2}(\mathrm{sign}(A) + I)$ and $\frac{1}{2}(\mathrm{sign}(A) - I)$ are projections onto the invariant subspaces corresponding to the eigenvalues with $\mathrm{Re}(\lambda) > 0$ and $\mathrm{Re}(\lambda) < 0$, respectively; we illustrate this below for $A = A^*$.

1.2. Description of the standard and spectral divide-and-conquer algorithms. Here we illustrate the four algorithms: (i) standard and (ii) spectral divide-and-conquer algorithms for (a) the symmetric eigendecomposition, and (b) the SVD.

Standard algorithm for symmetric eigendecomposition. One first reduces the symmetric matrix A to tridiagonal form by $n - 2$ Householder transformations, whose pictorial description for $n = 5$ is

$$A = \begin{bmatrix} * & * & * & * & * \\ * & * & * & * & * \\ * & * & * & * & * \\ * & * & * & * & * \\ * & * & * & * & * \end{bmatrix} \overset{H_1}{\rightarrow} \begin{bmatrix} * & * & & & \\ * & * & * & * & * \\ & * & * & * & * \\ & * & * & * & * \\ & * & * & * & * \end{bmatrix} \overset{H_2}{\rightarrow} \begin{bmatrix} * & * & & & \\ * & * & * & & \\ & * & * & * & * \\ & & * & * & * \\ & & * & * & * \end{bmatrix} \overset{H_3}{\rightarrow} \begin{bmatrix} * & * & & & \\ * & * & * & & \\ & * & * & * & \\ & & * & * & * \\ & & & * & * \end{bmatrix} \equiv T.$$

Here, H_i indicates an application of a Householder transformation $A \leftarrow H_i A H_i$, where $H_i = I - 2v_i v_i^*$, $\|v_i\|_2 = 1$; see, e.g., [22, Ch. 5]. Thus we have $A = HTH^*$, where $H = H_{n-2}H_{n-3}\cdots H_1$. Then the eigendecomposition of the tridiagonal matrix $T = Q\Lambda Q^*$ is computed, for which many efficient algorithms are available, such as QR [50, Ch. 8], divide-and-conquer [26], and MR^3 [17]. Together we obtain the eigendecomposition $A = V\Lambda V^*$ where $V = HQ$.

Spectral divide-and-conquer for symmetric eigendecomposition. Recall that for symmetric A, we have $U_p = \mathrm{sign}(A)$. Moreover, the matrix $Z = \begin{bmatrix} Z_+ & Z_- \end{bmatrix}$ in (1.3) can be chosen to be unitary. From (1.3), it then follows that $\frac{1}{2}(U_p + I) = Z_+ Z_+^*$ is an orthogonal projection onto the eigenspace corresponding to the positive eigenvalues of A. Thus by extracting a matrix V_+ with orthonormal columns such that $V_+ V_+^* = \frac{1}{2}(U_p + I)$ and finding its orthogonal complement V_-, we obtain a unitary matrix $V_1 := \begin{bmatrix} V_+ & V_- \end{bmatrix}$ such that $V_1 A V_1^*$ is block diagonalized. The whole matrix is diagonalized by recursively working with the smaller diagonal blocks; pictorially,

$$A = \begin{bmatrix} * & * & * & * & * \\ * & * & * & * & * \\ * & * & * & * & * \\ * & * & * & * & * \\ * & * & * & * & * \end{bmatrix} \overset{V_1}{\rightarrow} \begin{bmatrix} * & * & * & & \\ * & * & * & & \\ * & * & * & & \\ & & & * & * \\ & & & * & * \end{bmatrix} \overset{V_2}{\rightarrow} \begin{bmatrix} * & & & & \\ & * & * & & \\ & * & * & & \\ & & & * & \\ & & & & * \end{bmatrix} \overset{V_3}{\rightarrow} \begin{bmatrix} * & & & & \\ & * & & & \\ & & * & & \\ & & & * & \\ & & & & * \end{bmatrix} = \Lambda. \tag{1.4}$$

Here, V_i represents the unitary transformation $A \leftarrow V_i^* A V_i$, where V_i is block diagonal with up to 2^{i-1} diagonal blocks; for the example shown above, $V_2 = \mathrm{diag}(V_{21}, V_{22})$ and $V_3 = \mathrm{diag}(1, V_{32}, I_2)$ where $V_{21} \in \mathbb{C}^{3\times 3}, V_{22} \in \mathbb{C}^{2\times 2}, V_{32} \in \mathbb{C}^{2\times 2}$. Then the eigendecomposition is $A = V \Lambda V^*$, where $V = V_1 V_2 \cdots V_k$. Here, k is the recursion depth, usually $\approx \log_2 n$. Such recursive decoupling into smaller submatrices is called spectral divide-and-conquer.

Standard SVD algorithm. The first step is to reduce $A \in \mathbb{C}^{m\times n}$ to bidiagonal form. For example with $m = 5, n = 4$,

$$A \overset{H_{L,1}}{\rightarrow} \begin{bmatrix} \star & \star & \star & \star \\ & \star & \star & \star \\ & \star & \star & \star \\ & \star & \star & \star \\ & \star & \star & \star \end{bmatrix} \overset{H_{R,1}}{\rightarrow} \begin{bmatrix} \star & \star & & \\ & \star & \star & \star \\ & \star & \star & \star \\ & \star & \star & \star \\ & \star & \star & \star \end{bmatrix} \overset{H_{L,2}}{\rightarrow} \begin{bmatrix} \star & \star & & \\ & \star & \star & \star \\ & & \star & \star \\ & & \star & \star \\ & & \star & \star \end{bmatrix} \overset{H_{R,2}}{\rightarrow} \begin{bmatrix} \star & \star & & \\ & \star & \star & \\ & & \star & \star \\ & & \star & \star \\ & & \star & \star \end{bmatrix} \overset{H_{L,3}}{\rightarrow} \begin{bmatrix} \star & \star & & \\ & \star & \star & \\ & & \star & \star \\ & & & \star \\ & & & \star \end{bmatrix} \overset{H_{L,4}}{\rightarrow} B,$$

where $H_{L,i}$ indicates a left-multiplication by a Householder reflection, and $H_{R,i}$ a right-multiplication. Above, $H_{L,4}$ eliminates the $(5,4)$ element to complete the bidiagonalization. Here we use $\star$ instead of $*$ to indicate that the matrix is non-symmetric. Thus $H_{L,m-1} \cdots H_{L,1} A H_{R,1} \cdots H_{R,n-2} (=: H_L A H_R) = B$, and one then invokes a bidiagonal SVD for the bidiagonal $B = U_B \Sigma V_B^*$, and obtain the whole SVD as $A = U \Sigma V^*$, where $U = H_L^* U_B$ and $V = H_R V_B$.

Spectral divide-and-conquer for SVD. We first compute the polar decomposition $A = U_p H$, then obtain the symmetric (positive semidefinite) eigendecomposition $H = V \Sigma V^*$ by spectral divide-and-conquer. as in (1.4):

$$A = \begin{bmatrix} \star & \star & \star & \star \\ \star & \star & \star & \star \\ \star & \star & \star & \star \\ \star & \star & \star & \star \\ \star & \star & \star & \star \end{bmatrix} = U_p \begin{bmatrix} * & * & * & * \\ * & * & * & * \\ * & * & * & * \\ * & * & * & * \end{bmatrix} = U_p H, \qquad H \overset{V_1}{\rightarrow} \begin{bmatrix} * & * & & \\ * & * & & \\ & & * & * \\ & & * & * \end{bmatrix} \overset{V_2}{\rightarrow} \begin{bmatrix} * & & & \\ & * & & \\ & & * & \\ & & & * \end{bmatrix} = \Sigma.$$

Then $A = (U_p V) \Sigma V^* = U \Sigma V^*$ is the SVD.

2. The QDWH algorithm as a rational approximation to $\mathrm{sign}(x)$**.** The QDWH algorithm [46] computes the unitary polar factor U_p of a full-rank matrix A as the limit of the sequence X_k defined by

$$X_{k+1} = X_k (a_k I + b_k X_k^* X_k)(I + c_k X_k^* X_k)^{-1}, \qquad X_0 = A/\alpha. \tag{2.1}$$

Here, $\alpha > 0$ is an estimate of $\|A\|_2$ such that $\alpha \gtrsim \|A\|_2$. Setting $a_k = 3$, $b_k = 1$, $c_k = 3$ gives the Halley iteration, which is the cubically convergent member of the family of

principal Padé iterations [31, § 8.5]. The iterates (2.1) preserve the singular vectors while mapping the singular values by a rational function $R_k(\cdots R_2(R_1(x))\cdots)$, i.e., $X_k = U\Sigma_k V^*$, where $\Sigma_k = R_k(\cdots R_2(R_1(\Sigma_0))\cdots)$ is the diagonal matrix with ith diagonal $R_k(\cdots R_2(R_1(\sigma_i/\alpha))\cdots)$; equivalently $R(\Sigma_0)$ denotes the matrix function in the classical sense [31]. The choice of the rational functions $R_k(x)$ is of crucial importance in this paper, and in QDWH $R_k(x) = x\frac{a_k+b_kx^2}{1+c_kx^2}$, in which the parameters a_k, b_k, c_k are chosen dynamically to optimize the convergence. They are computed by $a_k = h(\ell_k)$, $b_k = (a_k-1)^2/4$, $c_k = a_k + b_k - 1$, where $h(\ell) = \sqrt{1+\gamma} + \frac{1}{2}\big(8 - 4\gamma + 8(2-\ell^2)/(\ell^2\sqrt{1+\gamma})\big)^{1/2}$, $\gamma = \big(4(1-\ell^2)/\ell^4\big)^{1/3}$. Here, ℓ_k is a lower bound for the smallest singular value of X_k, which is computed from the recurrence $\ell_k = \ell_{k-1}(a_{k-1} + b_{k-1}\ell_{k-1}^2)/(1 + c_{k-1}\ell_{k-1}^2)$ for $k \geq 1$. Note that all the parameters are available for free (without any matrix computations) for all $k \geq 0$ once we have estimates $\alpha \gtrsim \|A\|_2$ and $\ell_0 \lesssim \sigma_{\min}(X_0)$, obtained for example via a condition number estimator.

With such parameters, the iteration (2.1) is cubically convergent and needs at most six iterations for convergence to U_p with the tolerance u for any matrix A with $\kappa_2(A) \leq u^{-1}$, i.e., $\|X_6 - U_p\|_2 = \mathcal{O}(u)$. We note that iteration (2.1) has a mathematically equivalent QR-based implementation [46, § 4], which is numerically more stable (this is the actual QDWH iteration):

$$\left[\begin{array}{c}\sqrt{c_k}X_k\\ I\end{array}\right] = \left[\begin{array}{c}Q_1\\ Q_2\end{array}\right]R, \quad X_{k+1} = \frac{b_k}{c_k}X_k + \frac{1}{\sqrt{c_k}}\left(a_k - \frac{b_k}{c_k}\right)Q_1Q_2^*, \quad k \geq 0. \tag{2.2}$$

Once the computed polar factor $\widehat{U}_p$ is obtained, we compute the symmetric polar factor $\widehat{H}$ by $\widehat{H} = \frac{1}{2}(\widehat{U}_p^*A + (\widehat{U}_p^*A)^*)$ [31, § 8.8].

2.1. QDWH mapping function seen as the best type-$(3,2)$ approximation. The QDWH parameters a, b, c in (2.1) are chosen so that the interval $[\ell, 1]$ (in which all the singular values of X lie) is mapped as close as possible to the point 1 by the type $(3,2)$ rational function $R(x) = x\frac{a+bx^2}{1+cx^2} \in \mathcal{R}_{3,2}$. In [46] they are obtained as the solution for the rational max-min optimization problem

$$\max_{a,b,c}\ \min_{\ell\leq x\leq 1}\ x\frac{a+bx^2}{1+cx^2}, \tag{2.3}$$

subject to the constraint $x\frac{a+bx^2}{1+cx^2} \leq 1$ on $[0,1]$. The solution a, b, c results in the function that takes exactly one local maximum $R(x_M)$ and minimum $R(x_m)$ on $[\ell, 1]$ such that $\ell \leq x_M \leq x_m \leq 1$, and $R(x_M) = R(1) = 1$, $R(x_m) = R(\ell)$. The function $R(x)$ maps the interval $[\ell, 1]$ to $[R(\ell), 1]$. Figure 2.1 is a plot of $R(x)$ for the case $\ell = 0.01$.

Note that defining $\tau = (R(1)+R(\ell))/2$, the function $R(x)-\tau$ exhibits an *equioscillation* behavior on $[\ell, 1]$ with four extreme points, which is easily seen to be the largest possible by counting the number of critical points that an odd function $R(x) \in \mathcal{R}_{3,2}$ can have. This is known as the equioscillation property of best rational approximants [57], and implies optimality of $\frac{R(x)}{\tau}$ as an approximation to the sign function on $[-1, -\ell] \cup [\ell, 1]$; we give more details in Section 3.3. This means the parameters in the QDWH iteration can also be obtained by finding the best type-$(3,2)$ rational approximation to the sign function in the infinity norm (called a Zolotarev function), then scaling so that $\max_{0\leq x\leq 1} R(x) = 1$.

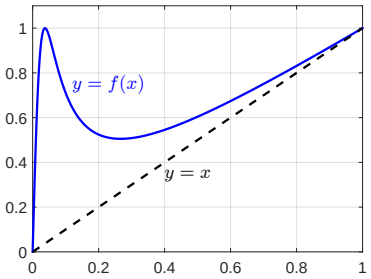


FIG. 2.1. *Plot of $R(x)$ for the QDWH iteration when $\ell = 0.01$.*

QDWH runs this process iteratively taking $\ell_k = \min_{\ell_{k-1} \le x \le 1} x\frac{a_{k-1}+b_{k-1}x^2}{1+c_{k-1}x^2}$, and the resulting function R_k in the kth iteration maps the interval $[\ell_{k-1}, 1]$ to $[\ell_k, 1] = R_k([\ell_{k-1}, 1])$ with $|1-\ell_k| \ll |1-\ell_{k-1}|$. Then the next QDWH iteration forms $R_{k+1}(x)$ defined by the parameters $(a_{k+1}, b_{k+1}, c_{k+1}) \neq (a_k, b_k, c_k)$, so that

$$[\ell_{k+1}, 1] = R_{k+1}([\ell_k, 1]) = R_{k+1}(R_k([\ell_{k-1}, 1])) = R_{k+1}(R_k \circ \cdots R_1([\ell_0, 1]))$$

with ℓ_{k+1} as close as possible to the point 1. The QDWH iteration therefore attempts to map the interval $[\ell_0, 1] = [\sigma_{\min}(X_0), 1]$ as close as possible to 1 by the composed rational function $G_k(x) = R_k(R_{k-1}(\cdots R_2(R_1(x))\cdots)$. As we shall see below, $G_k(x)$ is in fact the *best* rational approximation to $\mathrm{sign}(x)$ (i.e., a Zolotarev function) of type-$(3^k, 3^k-1)$ on $[-1, -\ell] \cup [\ell, 1]$, which are so powerful as approximants that with $k = 6$, $G_6([\ell, 1])) \subseteq [1-u, 1]$ for any ℓ such that $u < \ell \le 1$. Since $X_6 = UG_6(\Sigma_0)V^*$ where $X_0 = U\Sigma_0 V^*$ is the SVD (recall $A = X_0/\alpha$), this means $\|X_6 - U_p\|_2 \le \max_{x\in[\ell,1]} |G_6(x) - 1| \le u$, explaining why QDWH converges in six steps.

3. Zolotarev's rational approximations of the sign function. In light of the above observation, a natural idea is to consider approximations $R \in \mathcal{R}_{2r+1,2r}$ of the sign function $\mathrm{sign}(x)$ for general $r \ge 1$, with the goal to map all the singular values to 1. Since $\mathrm{sign}(x)$ is an odd function, the optimal approximant in $\mathcal{R}_{2r+1,2r}$ has the form $R(x) \equiv x\frac{P(x^2)}{Q(x^2)}$, where $P, Q \in \mathcal{P}_r$. As in the QDWH case (2.3), one way to obtain P and Q is to solve the max-min problem

$$\max_{P,Q\in\mathcal{P}_r} \min_{\ell\le x\le 1} x\frac{P(x^2)}{Q(x^2)} \tag{3.1}$$

subject to the constraint $x\frac{P(x^2)}{Q(x^2)} \le 1$ on $[0, 1]$. However, solving (3.1) via extending the approach in [46] for $r = 1$ to $r \ge 2$ seems highly nontrivial.

3.1. Zolotarev functions. Fortunately, the max-min problem (3.1) is equivalent to one of the classical rational approximation problems that Zolotarev [63] solved explicitly in 1877 in terms of Jacobi elliptic functions. Zolotarev showed [3, Ch. 9], [51, Ch. 4] that up to a scalar scaling, the solution $R(x) \equiv x\frac{P(x^2)}{Q(x^2)}$ of (3.1) is identical to the solution of the min-max problem

$$\min_{R\in\mathcal{R}_{2r+1,2r}} \max_{x\in[-1,-\ell]\cup[\ell,1]} |\mathrm{sign}(x) - R(x)|\,. \tag{3.2}$$

For any $0 < \ell < 1$ and integer $r \ge 0$, problem (3.2) has a unique solution, which we denote by $Z_{2r+1}(x;\ell) \in \mathcal{R}_{2r+1,2r}$. We call $Z_{2r+1}(x;\ell)$ the *type* $(2r+1, 2r)$ *Zolotarev*

function[2] corresponding to ℓ. As shown by Zolotarev [63] (see, e.g., [2, Ch. 9] or [3, Add. E]), the solution of (3.2) is given by

$$Z_{2r+1}(x;\ell) := Mx \prod_{j=1}^{r} \frac{x^2 + c_{2j}}{x^2 + c_{2j-1}}. \tag{3.3}$$

Here, the constant $M > 0$ is uniquely determined by the condition

$$1 - Z_{2r+1}(1;\ell) = -\big(1 - Z_{2r+1}(\ell;\ell)\big),$$

and the coefficients $c_1, c_2, \ldots, c_{2r}$ are given by

$$c_i = \ell^2 \frac{\operatorname{sn}^2\left(\frac{iK'}{2r+1};\ell'\right)}{\operatorname{cn}^2\left(\frac{iK'}{2r+1};\ell'\right)}, \quad i = 1, 2, \ldots, 2r, \tag{3.4}$$

where $\operatorname{sn}(u;\ell')$ and $\operatorname{cn}(u;\ell')$ are the Jacobi elliptic functions (see, e.g., [1, Ch. 17] or [2, Ch. 5]). Here $\ell' = \sqrt{1-\ell^2}$ and $K' = \int_0^{\pi/2} \frac{d\theta}{\sqrt{1-(\ell')^2 \sin^2\theta}}$. ℓ is called the modulus, ℓ' the complementary modulus, and K' the complete elliptic integral of the first kind for the complementary modulus. For later use, we note that as shown in [3, Add. E], the maximum

$$\|\operatorname{sign}(\,\cdot\,) - Z_{2r+1}(\,\cdot\,;\ell)\|_\infty := \max_{x\in[-1,-\ell]\cup[\ell,1]} |\operatorname{sign}(x) - Z_{2r+1}(x;\ell)|$$

is attained at exactly $2r+2$ points $x_1 := \ell < x_2 < \cdots < x_{2r+1} < x_{2r+2} := 1$ in the interval $[\ell, 1]$ and also at exactly $2r+2$ points $x_{-j} := -x_j$, $j = 1, 2, \ldots, 2r+2$, in the interval $[-1, -\ell]$. Furthermore, the function $\operatorname{sign}(x) - Z_{2r+1}(x;\ell)$ equioscillates between the x_j's, in particular,

$$1 - Z_{2r+1}(x_j;\ell) = (-1)^{j+1} \|\operatorname{sign}(\,\cdot\,) - Z_{2r+1}(\,\cdot\,;\ell)\|_\infty, \quad j = 1, 2, \ldots, 2r+2. \tag{3.5}$$

3.1.1. Quality of $Z_{2r+1}(x;\ell)$ as approximants to $\operatorname{sign}(x)$. To illustrate the power of Zolotarev functions as approximants to $\operatorname{sign}(x)$, for $r = 1, 2, \ldots, 6$ and $\ell = 10^{-3}$, in Figure 3.1 we plot $Z_{2r+1}(x;\ell)$ for $0 \le x \le 1$; this suffices for an illustration as $Z_{2r+1}(x;\ell)$ is an odd function. Observe that as r increases, the Zolotarev function quickly becomes a very good approximation to the sign function.

The quality of Zolotarev functions as approximants to $\operatorname{sign}(x)$ is known to improve exponentially with the degree $n = 2r+1$ [51, § 4.3]:

$$\|\operatorname{sign}(\,\cdot\,) - Z_n(\,\cdot\,;\ell)\|_\infty \approx C \exp(-cn) \tag{3.6}$$

for some $C, c > 0$ that depend on ℓ. This can also be understood via the exponential convergence of the trapezoidal rule, combined with a change of variables by a conformal map; see [28, § 5],[58]. More quantitative estimates for the error (3.6) are available in [23, eq. (32)]:

$$\frac{2}{\rho^n + 1} \le \|\operatorname{sign}(\,\cdot\,) - Z_n(\,\cdot\,;\ell)\|_\infty \le \frac{2}{\rho^n - 1}, \quad \rho = \rho(\ell) > 1, \tag{3.7}$$

[2]The index "$2r+1$", rather than "r", for the Zolotarev function Z_{2r+1} is used to facilitate the statement of the composition property of Zolotarev functions derived below.

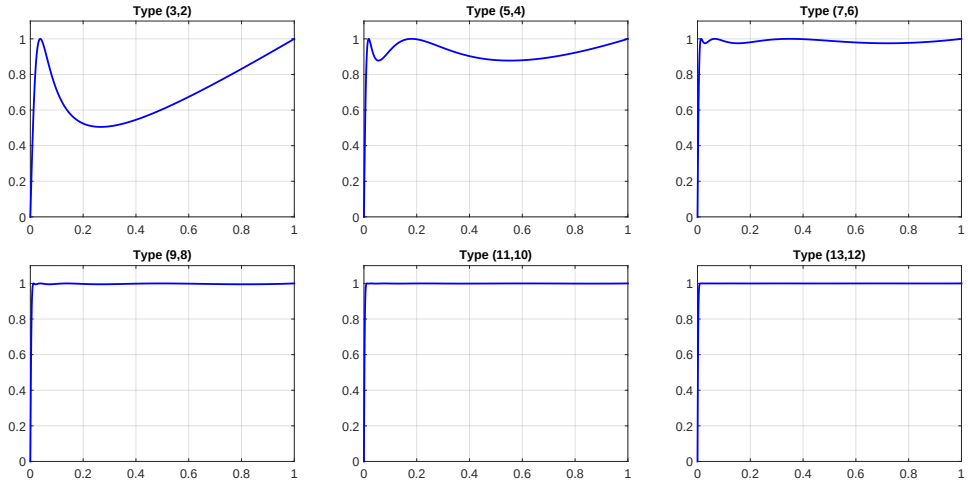


FIG. 3.1. *Zolotarev functions* $Z_{2r+1}(x;\ell)$ *of type* $(2r+1, 2r)$ *for* $r = 1, 2, \ldots, 6$ *and* $\ell = 10^{-3}$.

where $\rho(\ell) = \exp\left(\frac{\pi K(\mu')}{4K(\mu)}\right)$, $\mu = \frac{1-\sqrt{\ell}}{1+\sqrt{\ell}}$, $\mu' = \sqrt{1-\mu^2}$, and $K(\mu) = \int_0^{\pi/2} \frac{d\theta}{1-\mu^2 \sin^2\theta}$ denotes the complete elliptic integral of the first kind for modulus μ. Taking $n = 2r+1$, we conclude that the Zolotarev function Z_{2r+1} has error

$$\|\mathrm{sign}(\,\cdot\,) - Z_{2r+1}(\,\cdot\,;\ell)\|_\infty = C_{2r+1}\rho^{-(2r+1)} \tag{3.8}$$

with $1 \le \frac{2}{1+\rho^{-(2r+1)}} \le C_{2r+1} \le \frac{2}{1-\rho^{-(2r+1)}} \le \frac{2}{1-\rho^{-1}}$.

3.2. Rational function for Zolo-pd. For the following, it will be convenient to use the scaled Zolotarev function

$$\hat{Z}_{2r+1}(x;\ell) := \frac{Z_{2r+1}(x;\ell)}{Z_{2r+1}(1;\ell)} = \hat{M}x \prod_{j=1}^{r} \frac{x^2 + c_{2j}}{x^2 + c_{2j-1}}, \tag{3.9}$$

where $\hat{M} = \prod_{j=1}^{r}(1+c_{2j-1})/(1+c_{2j})$. We remark that $\hat{Z}_{2r+1}(1;\ell) = 1$, and $\hat{Z}_{2r+1}(x;\ell)$ maps the set $[-1,-\ell] \cup [\ell,1]$ onto $[-1,-\tilde{\ell}] \cup [\tilde{\ell},1]$, where $\tilde{\ell} := \min_{x\in[\ell,1]} \hat{Z}_{2r+1}(x;\ell) = \hat{Z}_{2r+1}(\ell;\ell)$. Note that we clearly have $0 < \ell < \tilde{\ell} < 1$.

Combining the iterative process used in QDWH and the scaled Zolotarev functions just described, we arrive at an algorithm that approximates the sign function by composing Zolotarev functions: starting with an interval $[\ell, 1]$ with $0 < \ell < 1$ that contains all the singular values of $X_0 = A/\alpha$, we set $\ell_0 := \ell$ and form $R_1 := \hat{Z}_{2r+1}(x;\ell_0)$, the scaled r-th Zolotarev function corresponding to ℓ_0, which maps $[-1,-\ell_0] \cup [\ell_0,1]$ onto $[-1,-\ell_1] \cup [\ell_1,1]$. Here, $\ell_1 := R_1(\ell_0)$ with $\ell_1 > \ell_0$, typically $|1-\ell_1| \ll |1-\ell_0|$. Then we form the r-th scaled Zolotarev function $R_2 := \hat{Z}_{2r+1}(x;\ell_1)$ corresponding to ℓ_1, which maps $[-1,-\ell_1] \cup [\ell_1,1]$ onto $[-1,-\ell_2] \cup [\ell_2,1]$, where $\ell_2 := R_2(\ell_1)$ and $\ell_2 > \ell_1$. The composed function $R_2(R_1(x))$ maps the original set $[-1,-\ell] \cup [\ell,1]$ onto $[-1,-\ell_2] \cup [\ell_2,1]$. Repeating this process k times we obtain $[\ell_k,1] = R_k([\ell_{k-1},1])$ with $|1-\ell_k| \ll |1-\ell_{k-1}|$; our algorithm Zolo-pd actually needs only $k = 2$ iterations.

3.3. Composed Zolotarev functions are again Zolotarev functions. The algorithm just described employs a composed rational function

$$R(x) := \hat{Z}_{2r+1}\big(\hat{Z}_{2r+1}(x;\ell);\ell_1\big) \tag{3.10}$$

that approximates the sign function on the interval $[-1,-\ell]\cup[\ell,1]$, in which both components are scaled Zolotarev functions, but corresponding to the different values ℓ and ℓ_1. The composed function R is clearly a rational function in $\mathcal{R}_{(2r+1)^2,(2r+1)^2-1}$, and thus we have 'squared' the degree of $\hat{Z}_{2r+1}(x;\ell)$.

In this subsection, we show that $R(x)$ as in (3.10) satisfies

$$R(x) = \hat{Z}_{(2r+1)^2}(x;\ell), \tag{3.11}$$

that is, R is in fact the scaled Zolotarev function of squared type $((2r+1)^2,(2r+1)^2-1)$ corresponding to the original value of ℓ. Similarly, we can compose k Zolotarev functions to obtain $R_k(\cdots R_2(R_1(x))\cdots) = \hat{Z}_{(2r+1)^k}(x;\ell)$, in which each $R_i \in \mathcal{R}_{2r+1,2r}$ is the scaled Zolotarev function of type $(2r+1,2r)$ corresponding to ℓ_{i-1}, where

$$\ell_i := \min_{x\in[\ell_{i-1},1]} \hat{Z}_{2r+1}(x;\ell_{i-1}) = \hat{Z}_{2r+1}(\ell_{i-1};\ell_{i-1}).$$

We emphasize that the fact that a high-degree best rational approximant is obtained as the composition of low-degree rational approximants is a remarkable special property enjoyed by the sign function, since k odd rational functions $R_1, R_2, \ldots, R_k \in \mathcal{R}_{2r+1,2r}$ altogether have only $k(2r+1)$ parameters, whereas general rational functions in $\mathcal{R}_{(2r+1)^k,(2r+1)^k-1}$ have $2(2r+1)^k$ degrees of freedom.

3.3.1. Best rational approximation on union of two intervals. To prove the statement (3.11) for any $\ell \in (0,1)$ and integer r, we need a classical result on the characterization of best rational approximants called *equioscillation* [51, 52, 57]. Consider the problem of approximating a given continuous function $F : [a,b] \mapsto \mathbb{R}$ on a real interval $[a,b]$ by a rational function $R \in \mathcal{R}_{m,n}$ (in this subsection only, m,n denote the degrees, not the matrix size):

$$\min_{R\in\mathcal{R}_{m,n}} \|F-R\|_\infty, \quad \text{where} \quad \|F-R\|_\infty := \max_{x\in[a,b]} |F(x)-R(x)|.$$

We say that the error $F-R$ *equioscillates* between k *extreme points* if there exist $a \le x_1 < x_2 < \cdots < x_k \le b$ such that

$$F(x_j) - R(x_j) = \rho(-1)^j \|F-R\|_\infty \quad j = 1,2,\ldots,k, \tag{3.12}$$

where ρ is either 1 or -1. The *defect* of a rational function $R = \frac{P}{Q} \in \mathcal{R}_{m,n}$ in $\mathcal{R}_{m,n}$ where P,Q are relatively prime polynomials is defined as $d(R) := \min\{m - \deg P, n - \deg Q\}$ if $R \not\equiv 0$ and $d(R) = 0$ if $R \equiv 0$. The classical characterization of best rational approximant [52, Thm. 7.2], [57, Thm. 10.1] can be stated as follows (originally due to Poncelet (1835) and Chebyshev; see [57, Ch. 10] for more on its history).

LEMMA 3.1. *Let $F : [a,b] \mapsto \mathbb{R}$ be a continuous function. Then, $R \in \mathcal{R}_{m,n}$ is the unique best rational approximant of F in $\mathcal{R}_{m,n}$ if, and only if, $F-R$ equioscillates between at least $m+n+2-d(R)$ extreme points, where d is the defect of R in $\mathcal{R}_{m,n}$.*

In what follows, all the rational functions we consider have maximum possible numerator and denominator degree in $\mathcal{R}_{m,n}$ so that the defect $d(R) = 0$. Regardless of $d(R)$, Lemma 3.1 shows that equioscillation of $F-R$ at $m+n+2$ extreme points is a sufficient condition for optimality. For our specific problem of approximating the sign function, we work with a union of two intervals $[-1,-\ell]\cup[\ell,1]$, and in particular we use the following sufficient condition for optimality (see also [51, Thm 4.4]). For a union $[a,c]\cup[d,b]$, where $a < c < d < b$, we still say $F-R$ equioscillates between k extreme points if (3.12) holds for k distinct increasing points $x_j \in [a,c]\cup[d,b]$.

LEMMA 3.2. *Let $F : [a, c]\cup[d, b] \mapsto \mathbb{R}$ be a continuous function and $a < c < d < b$. Then, $R \in \mathcal{R}_{m,n}$ is the unique best rational approximant of F in $\mathcal{R}_{m,n}$ if $F - R$ equioscillates between at least $m + n + 3$ extreme points.*

Note that the number of extreme points $m+n+3$ is one larger than in Lemma 3.1 for the case of a single interval $[a, b]$, and the lemma gives a sufficient condition for optimality, which may not be a necessary condition.

Proof. Suppose $F - R$ equioscillates between $m + n + 3$ extreme points satisfying (3.12) for $\{x_j\}_{j=1}^{m+n+3}$, and suppose there exists a better approximation $\tilde{R} \in \mathcal{R}_{m,n}$ such that $\|F - \tilde{R}\|_\infty < \|F - R\|_\infty$. Then $R - \tilde{R}$ must take alternating signs at the points x_j, which means it must be 0 in at least $m + n + 3 - 2 = m + n + 1$ distinct points[3], each in $[x_j, x_{j+1}]$, possibly except for the one that contains $\frac{c+d}{2}$. However, since $R - \tilde{R} \in \mathcal{R}_{m+n,2n}$, it cannot have more than $m + n$ zeros unless $R - \tilde{R} = 0$. ∎

Applying Lemma 3.2 to the case $F = \mathrm{sign}(x)$ on $[-1, -\ell] \cup [\ell, 1]$ shows that given a rational function $R \in \mathcal{R}_{2r+1,2r}$, we can test its optimality by counting the number of equioscillation points and by checking if this number is at least $2r+1+2r+3 = 4r+4$. For example, in view of (3.5), Zolotarev functions $Z_{2r+1}(x; \ell) \in \mathcal{R}_{2r+1,2r}$ satisfy this property. Observe that each Zolotarev function in Figure 3.1 has exactly $2r + 2$ equioscillation points in $[\ell, 1]$ (the fact that F is the sign function makes counting the equioscillation points straightforward), and by the symmetry about the origin, in total there are $4r + 4$ equioscillation points, verifying its optimality as the best rational approximant.

3.3.2. Equioscillation of composed Zolotarev functions. We now show that the composed scaled Zolotarev function $\hat{Z}_{2r+1}\big(\hat{Z}_{2r+1}(x; \ell); \ell_1\big)$ is indeed a scaled Zolotarev function of higher type. We actually prove a slightly more general statement in which the two Zolotarev functions are allowed to be of different types.

THEOREM 3.3. *Let $\hat{Z}_{2r_1+1}(x; \ell) \in \mathcal{R}_{2r_1+1,2r_1}$ be the scaled Zolotarev function corresponding to $\ell \in (0, 1)$, and let $\hat{Z}_{2r_2+1}(x; \ell_1) \in \mathcal{R}_{2r_2+1,2r_2}$ be the scaled Zolotarev function corresponding to $\ell_1 := \hat{Z}_{2r_1+1}(\ell; \ell)$. Then*

$$\hat{Z}_{2r_2+1}\big(\hat{Z}_{2r_1+1}(x; \ell); \ell_1\big) = \hat{Z}_{(2r_1+1)(2r_2+1)}(x; \ell). \tag{3.13}$$

Proof. The function $\hat{Z}_{2r_2+1}\big(\hat{Z}_{2r_1+1}(x; \ell); \ell_1\big)$ is clearly a rational function of type $((2r_1+1)(2r_2+1), (2r_1+1)(2r_2+1)-1)$. Thus, by Lemma 3.2, it suffices to prove that $\tilde{M}_2 \,\mathrm{sign}(x) - \hat{Z}_{2r_2+1}(\hat{Z}_{2r_1+1}(x; \ell); \ell_1)$ equioscillates between at least $(2r_1 + 1)(2r_2 + 1)+(2r_1+1)(2r_2+1)-1+3 = 2(2r_1+1)(2r_2+1)+2$ extreme points in $[-1, -\ell]\cup[\ell, 1]$ for some $\tilde{M}_2 > 0$; indeed $\tilde{M}_2 = (\ell_2 + 1)/2$ with $\ell_2 = \hat{Z}_{2r_2+1}(\ell_1; \ell_1)$. By symmetry it suffices to show that $\tilde{M}_2 - \hat{Z}_{2r_2+1}(\hat{Z}_{2r_1+1}(x; \ell); \ell_1)$ equioscillates between at least $(2r_1 + 1)(2r_2 + 1) + 1$ extreme points in the positive interval $[\ell, 1]$.

In view of (3.5), the function $\tilde{M}_1 - \hat{Z}_{2r_1+1}(x; \ell)$ where $\tilde{M}_1 = (\ell_1 + 1)/2$ equioscillates between the $2r+2$ points $\ell = x_1 < x_2 < \cdots < x_{2r_1+2} = 1$. Consider the intervals $[x_j, x_{j+1}]$ for $j = 1, 2, \ldots, 2r_1 + 1$. The function $\hat{Z}_{2r_1+1}(x; \ell)$ is continuous and maps each $[x_j, x_{j+1}]$ onto the interval $[\ell_1, 1]$, which is then mapped by $\hat{Z}_{2r_2+1}(x; \ell_1)$, in such a way that $\tilde{M}_2 - \hat{Z}_{2r_2+1}\big(\hat{Z}_{2r_1+1}(x; \ell); \ell_1\big)$ has $2r_2 + 2$ equioscillation points on $[x_j, x_{j+1}]$, including those precisely at x_j, x_{j+1}. Hence summing up the equioscillation points on $[\ell, 1]$ gives $(2r_1 + 1)(2r_2 + 2) - 2r_1$ points, in which the subtraction

[3]This number would be $m + n + 2$ if we have only a single interval instead of a union of two intervals.

of $2r_1$ accounts for the double-counted points x_j for $j = 2, 3, \ldots, 2r_1 + 1$. Since $(2r_1 + 1)(2r_2 + 2) - 2r_1 = (2r_1 + 1)(2r_2 + 2) + 1$, we thus have the required number of equioscillation points. ∎

Figure 3.2 gives an illustration for $r := r_1 = r_2 = 1$, $k = 2$ and $\ell = \frac{1}{500}$. The red points are the equioscillation points x_j of $\tilde{M}_1 - \hat{Z}_{2r+1}(x;\ell)$, with each $[x_j, x_{j+1}]$ being mapped to $[\ell_1, 1]$ with $\ell_1 \approx 0.31$. The black points are equioscillation points of $\tilde{M}_2 - \hat{Z}_{2r+1}(x;\ell_1)$, which generate equioscillation points at the x-values of the blue points. Shown in solid green is the composed Zolotarev function $\hat{Z}_{2r+1}(\hat{Z}_{2r+1}(x;\ell);\ell_1) = \hat{Z}_{(2r+1)^2}(x;\ell)$.

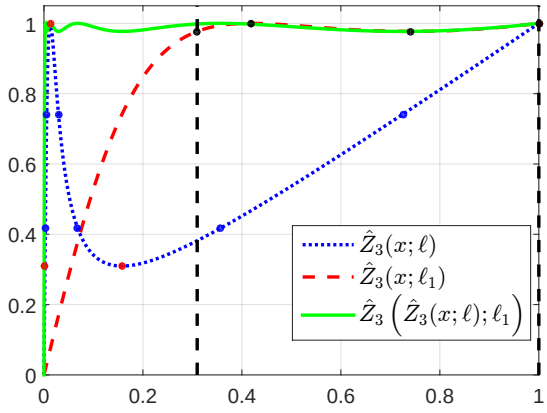


FIG. 3.2. *Zolotarev functions $\hat{Z}_3(x;\ell)$, $\hat{Z}_3(x;\ell_1)$, their composition $\hat{Z}_3\left(\hat{Z}_3(x;\ell);\ell_1\right) = \hat{Z}_9(x;\ell)$, and their equioscillation points for $\ell = \frac{1}{500}$.*

By applying Theorem 3.3 k times, one readily obtains the following result.

COROLLARY 3.4. *Let $r_1, r_2, \ldots, r_k \geq 0$ be integers. Under the assumptions of Theorem 3.3, define ℓ_i iteratively by $\ell_0 = \ell$ and $\ell_{i+1} := \hat{Z}_{2r_{i+1}+1}(\ell_i;\ell_i)$. Then,*

$$\hat{Z}_{2r_k+1}(\cdots \hat{Z}_{2r_2+1}(\hat{Z}_{2r_1+1}(x;\ell);\ell_1);\cdots;\ell_{k-1}) = \hat{Z}_{\prod_{i=1}^{k}(2r_i+1)}(x;\ell) \tag{3.14}$$

is the scaled Zolotarev function of type $(\prod_{i=1}^{k}(2r_i+1), \prod_{i=1}^{k}(2r_i+1)^k - 1)$.

In what follows we shall take r_i to be the same which we simply write r, so we employ Zolotarev functions of type $((2r+1)^k, (2r+1)^k - 1)$.

3.4. Benefits of computing $\hat{Z}_{(2r+1)^k}(x,\ell)$ by composition. There are two ways to compute $\hat{Z}_{(2r+1)^k}(x;\ell)$, eventually at a matrix argument $x = X_0 = A/\alpha$.

1. Directly by (3.9) as $MX_0 \prod_{j=1}^{\frac{(2r+1)^k-1}{2}} (X_0^* X_0 + c_{2j} I)(X_0^* X_0 + c_{2j-1} I)^{-1}$.
2. Compute k Zolotarev functions $\hat{Z}_{2r+1}(x;\ell), \hat{Z}_{2r+1}(x;\ell_1), \ldots, \hat{Z}_{2r+1}(x;\ell_{k-1})$ and then form $\hat{Z}_{2r+1}(\cdots \hat{Z}_{2r+1}(\hat{Z}_{2r+1}(X_0;\ell);\ell_1)\cdots);\ell_{k-1})$.

We argue that the second approach by composition is significantly better, both in speed and numerical stability.

Regarding the speed, let us compare the evaluations $Z = \hat{Z}_{(2r+1)^k}(X_0;\ell)$ and $\hat{Z}_{2r+1}(\cdots(\hat{Z}_{2r+1}(X_0;\ell);\cdots);\ell_{k-1})$ at a matrix X_0. Using the partial fraction representation (discussed in Section 4.1) the direct evaluation of $\hat{Z}_{(2r+1)^k}(X_0;\ell)$ requires $\frac{(2r+1)^k-1}{2}$ matrix operations (QR factorizations and matrix multiplications). By con-

trast, consider computing $\hat{Z}_{2r+1}(\cdots(\hat{Z}_{2r+1}(X_0;\ell);\cdots);\ell_{k-1})$, which we do sequentially as follows: $Y_1 = \hat{Z}_{2r+1}(X_0;\ell)$, $Y_{i+1} = \hat{Z}_{2r+1}(Y_i;\ell_i)$ for $i = 1, 2, \ldots, k-1$, and $Z = Y_{k-1}$. Since forming Y_i needs just one matrix operation for each i, this evaluation uses rk matrix operations. Clearly $\frac{(2r+1)^k-1}{2} \gg rk$ for $k > 1$ with difference growing exponentially with k, so evaluation by composition is much more efficient. Essentially the same argument holds when the input X_0 is a scalar.

Concerning stability, Figure 3.3 shows the error $\left\|\mathrm{sign}(\,\cdot\,) - Z_{(2r+1)^2}(\,\cdot\,;\ell)\right\|_\infty$ of the computed Zolotarev functions. We take various values of $\ell := 1/\kappa$, in which κ represents the matrix condition number in Zolo-pd, since $[\ell, 1]$ contains the singular values of $X_0 = A/\alpha$: recall that $\ell = 1/\kappa_2(A)$ when exact estimates $\alpha = \|A\|_2$ and $\ell = \sigma_{\min}(X_0)$ are used. The blue plots are from direct computation. Composition-based computation takes isolated values: the red circles show $r = 1$, so they take degrees $(3^k, 3^k - 1)$, and the black triangles are $r = 7$. The red dashed lines illustrate the theoretical asymptotic convergence.

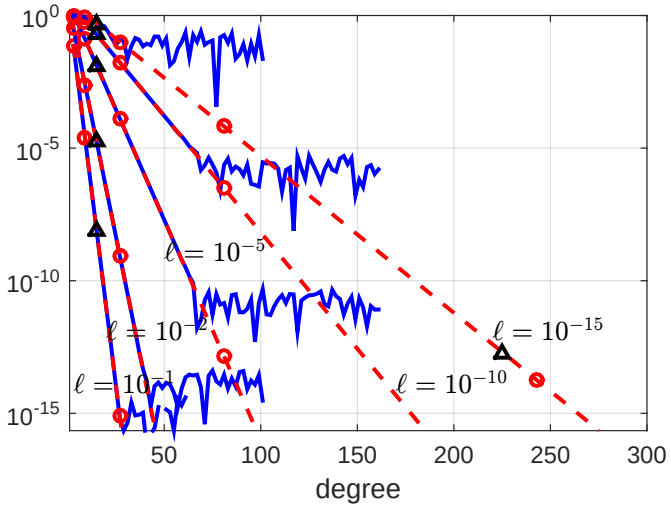


FIG. 3.3. *Zolotarev functions computed directly (blue) and via composition (red circle for $r = 1$, blue triangles for $r = 7$). The horizontal axis is the degree of the numerator polynomial of the Zolotarev function. Note that these plots are for scalars (not matrices).*

Observe that the blue plots stagnate at some threshold, showing that computing high-degree Zolotarev functions directly is unstable for high degree (this is with the "robust" way of computing the elliptic functions as discussed in Section 4.3). Indeed, some of the coefficients c_{2j-1}, c_{2j} in (3.3), (3.9) become very small as r grows (when $\ell = 10^{-16}$, $\min_i \sqrt{c_i} \approx 10^{-14}$ with $r = 8$), and relative precision is lost when dealing with terms like $x^2 + c_{2j-1}$, even if c_{2j-1} was computed very accurately. By contrast, computation by composition is accurate even when the degree $(2r+1)^k$ is high, as long as $r \le 8$. This is because each low-degree Zolotarev function $\hat{Z}_{2r+1}(x;\ell_i)$ is computed accurately, and hence so is its composition.

3.5. Choice of the integer r. Choosing a larger value of r increases the degree of the Zolotarev functions in $\mathcal{R}_{2r+1,2r}$, leading to an improved approximation to $\mathrm{sign}(x)$ as predicted by (3.8) and illustrated in Figure 3.1. This means the iteration generally converges in fewer steps by taking r larger. However, the arithmetic cost per iteration grows with r, and taking r too large can lead to numerical instability as

we just saw.

The key to the optimal choice of r is to have an accurate estimate of the number of iterations required for convergence. Specifically, recalling Section 2.1 and the fact that k iterations of Zolo-pd employs $\hat{Z}_{(2r+1)^k}(x;\ell)$, the number of iterations required is essentially the smallest value of k for which $\hat{Z}_{(2r+1)^k}([\ell,1];\ell) \subseteq [1-\mathcal{O}(u),1]$. This argument is much in the same vein as those in the literature for the scaled Newton iteration [31, p. 206] and the QDWH iteration [46].

Table 3.1 shows the smallest k for which $\hat{Z}_{(2r+1)^k}([\frac{1}{\kappa_2(A)},1];\ell) \subseteq [1-10^{-15},1]$ for varying values of r and $\ell = \frac{1}{\kappa_2(A)}$. Observe that the Zolotarev approximants with moderately large r are so powerful that *two* iterations is enough for convergence for a wide range of $\kappa_2(A)$.

TABLE 3.1
Required number of iterations for varying $\kappa_2(A)$ and r, obtained as the smallest k for which $\hat{Z}_{(2r+1)^k}([\ell,1])) \subseteq [1-\mathcal{O}(u),1]$.

$\kappa_2(A)$	1.001	1.01	1.1	1.2	1.5	2	10	10^2	10^3	10^5	10^7	10^{16}
$r=1$ (QDWH)	2	2	2	3	3	3	4	4	4	5	5	6
$r=2$	1	2	2	2	2	2	3	3	3	3	4	4
$r=3$	1	1	2	2	2	2	2	2	3	3	3	3
$r=4$	1	1	1	2	2	2	2	2	2	3	3	3
$r=5$	1	1	1	1	2	2	2	2	2	2	3	3
$r=6$	1	1	1	1	1	2	2	2	2	2	2	3
$r=7$	1	1	1	1	1	1	2	2	2	2	2	3
$r=8$	1	1	1	1	1	1	2	2	2	2	2	2

The numbers in Table 3.1 reflect the fact that for matrices of practical interest $\kappa_2(A) \leq u^{-1}$, we need a Zolotarev function of type about $(281,280)$ or higher for numerical convergence, as can be verified using (3.8). Since two iterations with $r=8$ gives a type $(289,288)$ Zolotarev function, it follows that we can take $r \leq 8$ in double-precision arithmetic to obtain an algorithm that converges in two steps.

In view of Corollary 3.4, it is possible to use different values of r_i for the first and second iterations, and by tuning these, one can slightly reduce the overall arithmetic cost. However, we do not pursue this here as the effect on the runtime will be marginal in a parallel implementation.

When $\kappa_2(A)$ happens to be well conditioned so that $\kappa_2(A) \leq 2$, one iteration is enough by taking r to be the value for which the entry is 1 in Table 3.1 (with a Cholesky-based efficient implementation; see Section 4.2). In the context of the symmetric eigendecomposition, our primary interest is in $\kappa_2(A) \simeq n$ because this is the average-case scenario of the condition number of a shifted matrix $A := A - \sigma I$ where σ is a scalar parameter that we can choose. For practical values $\kappa_2(A) \leq 5\times 10^5$, $r \leq 5$ is enough to obtain convergence in two iterations.

3.6. Convergence rate. Although our algorithm converges in two steps, it is still an iterative method; it has to be, since after all, the polar decomposition is not finitely computable [20]. Therefore a valid question arises: what is the convergence rate? The answer is $2r+1$, which can be as high as 17, as we take r to be up to 8.

Verifying this is straightforward. We denote the error after k iterations of composing Zolotarev functions of type $(2r+1,2r)$ by

$$\text{error}_k := \max_{x\in[-1,-\ell]\cup[\ell,1]} |\operatorname{sign}(x) - Z_{(2r+1)^k}(x;\ell)|, \tag{3.15}$$

which is the error of the approximant employed by k iterations of Zolo-pd.

THEOREM 3.5. *Let r be an integer and $\ell \in (0,1)$. Then the error after k iterations as in* (3.15) *converges to* 0 *with convergence rate* $2r+1$*, i.e.,*

$$\text{error}_{k+1} = C_{k,r}\big(\text{error}_k\big)^{2r+1}, \tag{3.16}$$

where, using C_n as in (3.8)*, $C_{k,r} = \frac{C_{(2r+1)^{k+1}}}{C_{(2r+1)^k}^{2r+1}}$. We have $C_{k,r} \approx 1$, or more specifically*

$$(1-\rho^{-1})^{2r+1} \le (1-\rho^{-(2r+1)^k})^{2r+1} \le C_{k,r} \le \frac{1}{1-\rho^{-(2r+1)^k}} \le \frac{1}{1-\rho^{-(2r+1)}}. \tag{3.17}$$

Proof. Recall that the error of the Zolotarev functions decays exponentially with the degree as in (3.8), which gives $\text{error}_k = C_{2r+1}\rho^{-2r+1}$. Hence the error after $k+1$ iterations is

$$\begin{aligned}
\text{error}_{k+1} &= \max_{x\in[-1,-\ell]\cup[\ell,1]} |Z_{(2r+1)^{k+1}}(x;\ell) - \text{sign}(x)| \\
&= C_{(2r+1)^{k+1}}\rho^{-(2r+1)^{k+1}} = C_{(2r+1)^{k+1}}(\rho^{-(2r+1)^k})^{2r+1} \\
&= \frac{C_{(2r+1)^{k+1}}}{C_{(2r+1)^k}^{2r+1}}(C_{(2r+1)^k}\rho^{-(2r+1)^k})^{2r+1} \\
&= C_{k,r}\big(\text{error}_k\big)^{2r+1}.
\end{aligned}$$

To obtain (3.17) we use the bounds $\frac{2}{1+\rho^{-(2r+1)}} \le C_{2r+1} \le \frac{2}{1-\rho^{-(2r+1)}}$ as in (3.8). ∎

The order $2r+1 = 17$ convergence in Zolo-pd $\text{error}_{k+1} = \mathcal{O}(\text{error}_{k+1})^{17}$ is deceptively fast: once we have $\text{error}_k \lesssim 0.1$, one more iteration is enough to give convergence to machine precision. Indeed this is a reasonably accurate illustration of what the algorithm does, as the first iteration maps the interval $[-1,-\ell]\cup[\ell,1]$ to $[-1,-\ell_1]\cup[\ell_1,1]$ with $1-\ell_1 = \mathcal{O}(0.1)$, and the second iteration maps $[-1,-\ell_1]\cup[\ell_1,1]$ to $[-1,-\ell_2]\cup[\ell_2,1]$ with $|\ell_2 - 1| \approx |\ell_1 - 1|^{17} \lesssim u$.

Of course, there is nothing special about the specific number 17; it just happens to be the smallest value of $2r+1$ for which the error after two iterations is $\mathcal{O}(u)$ for $\ell = u$ in double-precision arithmetic $u \approx 10^{-16}$. For example, in quadruple precision for which $u \approx 10^{-32}$, we will need $r = 16$ with convergence rate 33 for a two-step convergent algorithm, and in single precision $u \approx 10^{-8}$ it suffices to take $r = 4$.

3.7. Related studies on Zolotarev functions. Zolotarev's functions have a long and rich history. Regarding the result "composed Zolotarev is high-degree Zolotarev", a related observation has been made for the square root function originally by Rutishauser [54] (see also [13, Ch. V.5.D]), and also mentioned by Ninomiya [49] and Braess [12], in the context of accelerating Heron's iteration. They observe that by appropriately scaling Heron's iteration, which composes type $(2,1)$ approximants in each iteration, one can obtain the best rational approximant to the square root function, in terms of minimizing the relative error $\|\frac{\frac{P(\cdot)}{Q(\cdot)}-\sqrt{\cdot}}{\sqrt{\cdot}}\|_\infty$. Recently Beckermann [11] revisited this observation in the context of the matrix square root.

As Zolotarev proved [2, Ch. 9] [3, Add. E], the best rational approximants for the square root on $[\ell,1]$ and that for the sign function Z_{2r+1} are highly related: for the best rational approximant of type $(2r+1,2r)$ for the sign function $Z_{2r+1}(x;\ell) = x\frac{P_r(x^2)}{Q_r(x^2)}$, the function $\frac{Q_r(x)}{P_r(x)}$ is the minimizer over $P,Q\in\mathcal{P}_r$ of $\max_{x\in[\ell^2,1]}|(\sqrt{x}-\frac{P(x)}{Q(x)})/\sqrt{x}|$, the

relative error as an approximant for $\sqrt{x}$. Hence by using the results of Rutishauser, one can obtain Theorem 3.3 for composing Zolotarev functions of type $(2,1)$. This work extends the observation to composing rational approximants of arbitrary degrees, and revisits the connection between the best rational approximants for the sign and square root functions.

Van den Eshof and his coauthors [59, § 4.5.2],[60] used Zolotarev's results to approximate $\mathrm{sign}(A)x$ for a given vector x and symmetric A, for which we recall $U_p = \mathrm{sign}(A)$. To compute $\mathrm{sign}(A)x$ efficiently, they find the lowest degrees of P and Q such that the computed result has acceptable error (which they set to $\approx 10^{-2}$). A recent work of Güttel, Polizzi, Tang and Viaud [27] computes partial eigenpairs of large-sparse matrices using Zolotarev functions: they apply $Z_{2r+1}(A;\ell)$ to a vector (or a set of vectors) to obtain a "filter function" that gives a subspace rich in the eigenspace corresponding to the eigenvalues of A in a specified interval, and repeatedly apply the filter function to improve the accuracy. Druskin, Güttel, and Knizhnerman [18] use Zolotarev's function for the inverse square root function.

The crucial difference from these studies is that we employ a matrix iteration and compute the entire unitary polar factor U_p, instead of its action on a vector. This permits us to iterate on the computed result, and thus to form composed rational functions. The difference is exponential: applying the filter function repeatedly k times gives error $C\exp(-crk)$, whereas composing k times gives $C\exp(-cr^k)$. As a consequence, the accuracy to working precision in double-precision arithmetic is achievable without involving a very high-degree rational function (we need $r = 8$ or smaller, as opposed to $r \geq 140$ to get a single-step convergence to the tolerance u).

While some studies use the type $(2r-1,2r)$ Zolotarev functions, we focus on type $(2r+1,2r)$ functions because this results in a slightly better approximation since $\mathcal{R}_{2r-1,2r} \in \mathcal{R}_{2r+1,2r}$, and they require the same computational cost to evaluate at a matrix argument.

In the conclusion of [37] Iannazzo compares the iteration via Padé approximant for computing matrix functions (the paper focuses on the geometric mean) with the best rational approximant and notes that while Padé gives an order of convergence larger than 1, it requires degree much higher than the best rational approximant, so it is unclear which is more efficient. This work gives a clear answer for the sign function, which gets the best of both worlds: we obtain the highest possible order of convergence while simultaneously using the best rational approximant. Other cases in which Zolotarev's rational approximation is used in the context of matrix iterations include [28, 40, 42].

One may have wondered why we have restricted ourselves to odd rational functions, when the goal is to map the singular values, which are nonnegative numbers, to 1: perhaps a non-odd function that focuses on $[\ell,1]$ would be more effective. However, a matrix expressed as $Uf(\Sigma)V^*$ is generally difficult to compute unless $f(\Sigma)$ is an odd function; for example computing $U\Sigma^2V^*$ without the knowledge of the polar decomposition or SVD is already nontrivial. If f is odd so that $f(x) = x\frac{P(x^2)}{Q(x^2)}$ then we can compute $Uf(\Sigma)V^* = AP(A^*A)Q(A^*A)^{-1}$; we will use its partial fraction form for its evaluation (see Section 4.1). Moreover, iterations of the form $X_{k+1} = Uf(\Sigma)V^* = X_kP(X_k^*X_k)Q(X_k^*X_k)^{-1}$ can be interpreted as an iteration for the matrix sign function of $\left[\begin{smallmatrix} 0 & A \\ A^* & 0 \end{smallmatrix}\right]$ [31, Thm. 8.13]. Another explanation is that for the symmetric eigendecomposition, the goal of computing the polar decomposition of a shifted symmetric matrix $A-\sigma I$ is to split the spectrum into two distinct groups, so we need to map both (positive and negative) sides of the spectrum to distinct values,

a natural choice of which is ± 1.

Finally, Theorem 3.3 can be extended, by essentially the same proof, to best rational approximants to $\mathrm{sign}(x)$ of type $(2r, 2r-1)$. This shows that the scaled Newton iteration $X_{k+1} = \frac{1}{2}(\mu_k X_k + \mu_k^{-1} X_k^{-1})$ with the scaling $\mu_{k+1} = \sqrt{2/(\mu_k + \mu_k^{-1})}$ as in [14] composes k rational functions of type $(2,1)$ to obtain the type $(2^k, 2^k - 1)$ best rational approximant; this also explains why $k = 9$ iterations is enough, as it is the smallest integer for which $2^k \geq 280$. We thus have a unifying view of scaled Newton, QDWH, and Zolo-pd that they all employ high-degree Zolotarev functions obtained by composition.

4. Implementation issues. We now discuss the implementation of Zolo-pd for computing the polar decomposition of a matrix $A \in \mathbb{C}^{m\times n}$.

4.1. Evaluating $\hat{Z}_{2r+1}(x;\ell)$ at matrix arguments $\hat{Z}_{2r+1}(A;\ell)$. To apply the Zolotarev functions to the singular values of A, we need an efficient and stable way of computing $R(X_k) = U\hat{Z}_{2r+1}(\Sigma_k;\ell)V^*$, where $X_k = U\Sigma_k V^*$ is the SVD. For stability and communication efficiency, we look for an inverse-free implementation, i.e., one that does not explicitly invert matrices or require solutions to linear systems; this was the original motivation for the QDWH iteration [46].

Crucial to this task is the following result, which was given in [62], [31, p. 219], and was also used in the QDWH iteration (which is Zolo-pd for the special case $r = 1$).

LEMMA 4.1. *Let* $\begin{bmatrix} \eta X \\ I \end{bmatrix} = \begin{bmatrix} Q_1 \\ Q_2 \end{bmatrix} R$ *be the QR decomposition of* $\begin{bmatrix} \eta X \\ I \end{bmatrix}$, *where* $X, Q_1 \in \mathbb{C}^{m\times n}$ *and* $Q_2, R \in \mathbb{C}^{n\times n}$. *Then*

$$Q_1 Q_2^* = \eta X (I + \eta^2 X^* X)^{-1}. \tag{4.1}$$

We now show that by using the partial fraction representation of $Z_{2r+1}(x;\ell)$ we can use this lemma for a stable computation of $R(X)$ for any $r \geq 1$.

In general, a partial fraction representation expresses a given rational function in terms of a sum of fractions involving polynomials of low degree. For the Zolotarev function $\hat{Z}_{2r+1}(x;\ell)$ given by (3.9), we obtain the following partial fraction decomposition representation:

PROPOSITION 4.2. *The function* $\hat{Z}_{2r+1}(x;\ell)$ *as in* (3.9) *can be expressed as*

$$\hat{Z}_{2r+1}(x;\ell) = \hat{M}x \prod_{j=1}^{r} \frac{x^2 + c_{2j}}{x^2 + c_{2j-1}} = \hat{M}x \left(1 + \sum_{j=1}^{r} \frac{a_j}{x^2 + c_{2j-1}}\right), \tag{4.2}$$

where

$$a_j = -\left(\prod_{k=1}^{r} (c_{2j-1} - c_{2k})\right) \cdot \left(\prod_{k=1, k\neq j}^{r} (c_{2j-1} - c_{2k-1})\right). \tag{4.3}$$

Proof. Any rational function has a partial fraction form [30, Ch. 7], and since the c_{2j-1} are distinct an expression as in (4.2) must exist. It remains to establish (4.3); an easy way to verify it is to multiply (4.2) by $x^2 + c_{2j-1}$ and take $x = ic_{2j-1}$. ∎

Equation (4.3) provides a simple and stable way to compute the coefficients a_j in (4.2), as it involves only multiplications except for the subtractions in the c_i's, which are distinct numbers. We have $a_j > 0$ for all j and $\frac{a_j}{c_{2j-1}} = O(1)$, so a_j can be as large as $O(\ell^{-1})$.

To apply the Zolotarev functions to $X_0 = A/\alpha$, we need to compute $\hat{Z}_{2r+1}(x;\ell)$ at a matrix argument. The next result shows how this is accomplished.

PROPOSITION 4.3. *For the function $\hat{Z}_{2r+1}(x;\ell)$ as in (3.9), and a matrix X with SVD $X = U\mathrm{diag}(\sigma_i)V^*$, the matrix $\hat{Z}_{2r+1}(X;\ell) := U\mathrm{diag}(\hat{Z}_{2r+1}(\sigma_i;\ell))V^*$ is equal to*

$$\hat{Z}_{2r+1}(X;\ell) = \hat{M}X\prod_{j=1}^{r}(X^*X + c_{2j}I)\prod_{j=1}^{r}(X^*X + c_{2j-1}I)^{-1}, \tag{4.4}$$

which in turn can be rewritten in partial fraction form as

$$\hat{Z}_{2r+1}(X;\ell) = \hat{M}(X + \sum_{j=1}^{r} a_j X(X^*X + c_{2j-1}I)^{-1}). \tag{4.5}$$

Moreover, $\hat{Z}_{2r+1}(X;\ell)$ can be computed in an inverse-free manner as

$$\begin{cases} \begin{bmatrix} X \\ \sqrt{c_{2j-1}}I \end{bmatrix} &= \begin{bmatrix} Q_{j1} \\ Q_{j2} \end{bmatrix} R_j, \\ R(X) &= \hat{M}(X + \sum_{j=1}^{r} \frac{a_j}{\sqrt{c_{2j-1}}} Q_{j1}Q_{j2}^*). \end{cases} \tag{4.6}$$

Proof. Plugging $X = U\mathrm{diag}(\sigma_i)V^*$ into the right-hand side of (4.4) establishes the first statement. The same process together with Proposition 4.2 yields (4.5). Using Lemma 4.1 for each term $X(X^*X + c_{2j-1}I)^{-1}$ in (4.5) we obtain (4.6). ∎

Note that the r QR factorizations and matrix multiplications $Q_{j1}Q_{j2}^*$ in (4.6) are completely independent of each other, therefore we can easily compute $Q_{j1}Q_{j2}^*$ in a parallel fashion for $j = 1, 2, \ldots, r$ and compute $R(X)$ simply by adding up the matrices. Furthermore, numerically the evaluation (4.6) based on partial fractions is much more accurate than a direct evaluation.

We note that the use of partial fraction for Padé-type matrix iterations was employed by Kenney and Laub in [41] for the matrix sign function, and by Higham and Papadimitriou [32] for the polar decomposition. For the action of a matrix function on a vector, it was used for example in [18, 60].

Single-iteration algorithm?. Due the the "embarrassing" parallelizability, increasing r further does not pose a serious computational bottleneck in a parallel implementation. Hence a natural idea is to choose r large enough so that a single iteration is enough to obtain the unitary polar factor, i.e., $\hat{Z}_{2r+1}([1/\kappa_2(A), 1];\ell) \subseteq [1 - \mathcal{O}(u), 1]$. However, the value r needed to achieve this grows rapidly with $\kappa_2(A) = 1/\ell$. The exponential decay of the error (3.6) with the degree means this number grows like $\mathcal{O}(\log \kappa_2(A))$, and it is about $\frac{280}{2}$ when $\kappa_2(A) \approx 10^{16}$. Besides the obvious increase in the arithmetic cost, taking r this large entails the serious numerical instability observed in Figure 3.3.

In Zolo-pd, by allowing for the second iteration we reduce r from $\mathcal{O}(\log \kappa_2(A))$ to $\mathcal{O}(\sqrt{\log \kappa_2(A)})$. This improves the stability dramatically as we saw in Figure 3.3. Moreover, the overhead in speed of allowing the second iteration is marginal, and in fact the runtime is much less than doubled. This is because the second iteration can be executed more efficiently than the first using the Cholesky factorization, as we discuss next.

4.2. Using the Cholesky factorization in the second iteration. Recall that the iterate (4.6) is mathematically equivalent to (4.5), which can be computed (more directly) via (here Chol denotes the Cholesky factor)

$$\text{(4.7)} \qquad \begin{cases} Y_{2j-1} &= X^*X + c_{2j-1}I, \quad W_{2j-1} = \mathrm{Chol}(Y_{2j-1}), \\ R(X) &= \hat{M}(X + \sum_{j=1}^{r} a_j (XW_{2j-1}^{-1})W_{2j-1}^{-*}). \end{cases}$$

When X is well-conditioned, it is preferable to execute (4.7) instead of (4.6) for efficiency as discussed in [48, §5.6] in the context of QDWH. The arithmetic cost of (4.7) is $3mn^2 + n^3/3$ and that of (4.6) is $5mn^2$ for each j.

Since the forward error bound for computing the Cholesky factor $\mathrm{Chol}(Y_{2j-1})$ of Y_{2j-1} is proportional to the condition number $\kappa_2(Y_{2j-1}) = \kappa_2(X^*X + c_{2j-1}I)$, in [48] the iteration switches to the Cholesky-based implementation when $\kappa_2(Y_{2j-1}) \le 100$ is guaranteed, for which $\kappa_2(X) \le 10$ is a sufficient condition.

Fortunately, in practice (for $\kappa_2(A) < u^{-1}$ in double-precision arithmetic) we always have $\kappa_2(Y_{2j-1}) \le 3$ for the second iteration in Zolo-pd. This is because the type $(2r+1, 2r)$ rational approximation to $\mathrm{sign}(x)$ on $[-1, -\ell] \cup [\ell, 1]$ with $\ell = 10^{-15}$ and $r = 8$ maps the singular values to $\hat{Z}_{17}([10^{-15}, 1]; 10^{-15}) \subseteq [.39, 1]$. This can be seen from (3.8) and also from Table 3.1, which shows that we need $\kappa_2(A) \lesssim 2$ to get convergence in one step with $r \le 8$; more precisely we need $\kappa_2(A) \lesssim 2.6$. Since the interval $[10^{-15}, 1]$ is mapped to $[1 - 10^{-15}, 1]$ in two iterations, one iteration must result in an interval contained in $[\frac{1}{2.6}, 1] \supseteq [.39, 1]$.

Hence the second iteration can be safely computed via (4.7) using the Cholesky factorization. In fact, experiments suggest that using (4.7) for the second iteration improves not only the speed but also slightly the stability. To summarize, the first iteration of Zolo-pd needs to be executed using QR as in (4.6), and it is recommended that the second iteration be computed by Cholesky as in (4.7).

We remark that Zolo-pd can be made more efficient for well-conditioned A. Specifically, if $\kappa_2(A)$ (or its estimate) is smaller than 2, then we find from Table 3.1 the smallest r for which one iteration gives convergence and run (4.7) to obtain $X_1 = U_p$. Similarly, if $2 < \kappa_2(A) \le 10$ (or a modest number $\ll u^{-1}$) then we choose r as usual from Table 3.1 but run two steps of (4.7), both of which give stable results.

4.3. Computing the Zolotarev coefficients in MATLAB**.** Recall that the Zolotarev coefficients $c_1, c_2, \ldots, c_{2r}$ are defined by (3.4). In order to obtain the $c_i's$, we need to first compute the complete elliptic integral K' and then compute function values of the Jacobi elliptic functions sn and cn, all for the complementary modulus $\ell' = \sqrt{1-\ell^2}$. The standard approach for these tasks is based on the arithmetic-geometric mean (AGM) method; see, e.g., [1, Ch. 16 and 17]. MATLAB provides the functions `ellipke` (for the computation of complete ellptic integrals) and `ellipj` (for the computation of function values of the Jacobi elliptic functions), which are based on the AGM method. Unfortunately, the functions `ellipke` and `ellipj` use the complementary parameter $m' = (\ell')^2$ as input. Moreover, `ellipke` gives inaccurate values for K' for values of m' close to 1.

For our application, we have $\ell = \frac{1}{\kappa_2(A)}$ and thus

$$m' = 1 - \ell^2 = 1 - \frac{1}{\big(\kappa_2(A)\big)^2}.$$

This means that, as $\kappa_2(A)$ increases, m' is indeed close to 1. Even worse, in double-precision floating-point arithmetic, m' gets rounded to 1 for $\kappa_2(A) \gtrsim 10^8$. Due to

these issues, we cannot use the functions `ellipke` and `ellipj` in the form provided in MATLAB.

Fortunately, it turns out that running the AGM method requires only the value of $\sqrt{1-m'}$, but not the value of m'. Note that

$$\sqrt{1-m'} = \sqrt{1-(\ell')^2} = \ell.$$

We modified both `ellipke` and `ellipj` so that they use ℓ as their input, rather than m'. Furthermore, employing an asymptotic expansion of K' for $m' \to 1$ (or, equivalently, $\ell \to 0$) given in [15], we fixed the function `ellipke` so that it computes K' accurately even when ℓ is close to 0. For our actual computations of the Zolotarev coefficients $c_1, c_2, \ldots, c_{2r}$, we employ these modified versions of `ellipke` and `ellipj`.

We remark that previous attempts exist to compute the Jacobi elliptic functions reliably; see [28, 39, 43].

Finally, even with our improved implementation of computing the Zolotarev coefficients, when $1/\ell$ and r are large, rounding errors prevent the coefficients from being computed accurately, resulting in a computed Zolotarev function that does not approximate the sign function to the desired accuracy. As discussed before, we resolve this issue by allowing for two steps, which reduce r sufficiently so that the elliptic functions are computed accurately enough for our purpose. Indeed, if we use the MATLAB functions the blue plots in Figure 3.3 stagnate at a much higher value.

4.4. Stopping criterion. Although in exact arithmetic our algorithm converges in two steps, it is nontrivial to confirm that the iterate X_2 has indeed converged. One approach is to observe that for an iterate x_k whose rate of convergence is r, once we have $\|x_k - x_{k-1}\|/\|x_k\| \lesssim \epsilon^{1/r}$, we can expect $\|x_{k+1} - x_k\|/\|x_{k+1}\| \lesssim \epsilon$, i.e., convergence to tolerance ϵ is achieved at x_k. More detailed arguments are available in [31, § 4.9.2] for a quadratically convergent iterate and in [46] for a cubically convergent iterate.

Since the iterate we propose converges with rate $2r+1$, this suggests that we accept X_2 as converged if

$$\frac{\|X_2 - X_1\|_F}{\|X_2\|_F} \le u^{1/(2r+1)}. \tag{4.8}$$

This can be understood as an order-$(2r+1)$ variant of the stopping criterion

$$\frac{\|X_{k+1} - X_k\|_F}{\|X_k\|_F} \le cu^{1/2}$$

(for a constant c) for the quadratically convergent Netwon iteration [31, eqn. 8.31].

In all our experiments (4.8) was always satisfied. Another, more expensive but perhaps more reliable, way to test convergence is to check that $\frac{\|X_2^* X_2 - I\|_F}{\sqrt{n}}$ is $\mathcal{O}(u)$.

Although (4.8) was satisfied in all our experiments, it may fail to hold in unlucky cases, for example when poor estimates of α, ℓ_0 are used so that $\alpha \ll \|A\|_2$ or $\ell_0 \gg \sigma_{\min}(X_0)$ (as discussed in [48, § 5.8], ℓ_0 can be estimated using a condition number estimator as `1/condest(A)`; or by using the estimator of [19, 35]. When this happens we suggest running Zolo-pd again on X_2 (not A). Although X_2 may not be a numerically orthogonal matrix its singular values are still mapped to values close to 1, so $\kappa_2(X_2) \ll \kappa_2(A)$. In particular if $\kappa_2(X_2) \le 2$ then Zolo-pd gives single-step convergence via the Cholesky-based implementation as discussed in Section 4.2.

5. Algorithms for the polar, symmetric eigenvalue and singular value decompositions. Thus far we have discussed the fundamentals of our polar decomposition algorithm Zolo-pd. Moreover, as described in the introduction, a polar decomposition algorithm can be used as building blocks of algorithms for the symmetric eigendecomposition (Zolo-eig) and SVD (Zolo-SVD) via spectral divide-and conquer. We now formulate the algorithms as pseudocodes, and discuss their properties.

5.1. Polar decomposition. Algorithm 5.1 is the pseudocode for Zolo-pd. Note that it is shown explicitly as a two-step algorithm without iterations. As noted in Section 4.2, for well-conditioned A ($\kappa_2(A) \le 2$), we can skip the first iteration of (5.1) and just run (5.2) by choosing a value of r that gives convergence in a single step. Furthermore, since $\kappa_2(A) = \mathcal{O}(1)$, we can safely invoke the Cholesky-based implementation in Section 4.2, so the algorithm becomes more than twice as fast, compared to the case $\kappa_2(A) > 100$.

Algorithm 5.1 Zolo-pd: compute the polar decomposition $A = U_pH$

Input: $A \in \mathbb{C}^{m\times n}$ with $m \ge n$.

Output: U_p, H such that $A = U_pH$, where $U_p^*U_p = I_n$ and H is positive semidefinite.

1: Estimate $\alpha \gtrsim \sigma_{\max}(A), \beta \lesssim \sigma_{\min}(A)$, $X_0 = A/\alpha$, $\ell = \beta/\alpha$.

2: Choose r based on $\kappa = \ell^{-1}$ from Table 3.1. If $\kappa < 2$ then $X_1 = A$ and skip to (iv).

3: Compute X_1 and X_2:

(i). Compute $c_j = \ell^2 \operatorname{sn}^2\left(\frac{iK'}{2r+1}; \ell'\right) / \operatorname{cn}^2\left(\frac{iK'}{2r+1}; \ell'\right)$ as in (3.4), and $a_j = -\left(\prod_{k=1}^{r}(c_{2j-1} - c_{2k})\right) \cdot \left(\prod_{k=1,k\neq j}^{r}(c_{2j-1} - c_{2k-1})\right)$ as in (4.3).

(ii). Compute X_1 by $\hat{M} = \prod_{j=1}^{r}(1 + c_{2j-1})/(1 + c_{2j})$ and

$$\text{(5.1)} \qquad \begin{cases} \begin{bmatrix} X_0 \\ \sqrt{c_{2j-1}}I \end{bmatrix} &= \begin{bmatrix} Q_{j1} \\ Q_{j2} \end{bmatrix} R_j, \quad j = 1, 2, \ldots, r, \\ X_1 &= \hat{M}\big(X_0 + \sum_{j=1}^{r} \frac{a_j}{\sqrt{c_{2j-1}}} Q_{j1}Q_{j2}^*\big). \end{cases}$$

(iii). Update $\ell := \hat{M}\ell \prod_{j=1}^{r}(\ell^2 + c_{2j})/(\ell^2 + c_{2j-1})$ and recompute c_j and a_j as in step (i).

(iv). Compute X_2 by $\hat{M} = \prod_{j=1}^{r}(1 + c_{2j-1})/(1 + c_{2j})$ and

$$\text{(5.2)} \qquad \begin{cases} Z_{2j-1} &= X_1^*X_1 + c_{2j-1}I, \quad W_{2j-1} = \operatorname{Chol}(Z_{2j-1}), \\ X_2 &= \hat{M}\big(X_1 + \sum_{j=1}^{r} a_j (X_2 W_{2j-1}^{-1}) W_{2j-1}^{-*}\big). \end{cases}$$

Verify that $\frac{\|X_2 - X_1\|_F}{\|X_2\|_F} \le u^{1/(2r+1)}$ holds. If not, return to step 1 with $A \leftarrow X_2$.

4: $U_p = X_2$, $H = \frac{1}{2}(U_p^*A + (U_p^*A)^*)$.

When high accuracy is desired, we perform a Newton–Schulz postprocessing $U := \frac{3}{2}\widehat{U} - \frac{1}{2}\widehat{U}(\widehat{U}^*\widehat{U})$ after computing the unitary factors (matrices of singular vectors and eigenvectors), which improves the orthogonality and backward stability of the computed results as discussed in [48]. This comes with a nonnegligible cost of $3n^3$ flops for the eigendecomposition, and $6n^3$ (for both factors U, V; they can be done in parallel) for the SVD.

5.1.1. Backward stability. A backward stable polar decomposition algorithm computes $\widehat{U}_p, \widehat{H}$ such that $\widehat{H}$ is symmetric and $\|\widehat{U}_p^*\widehat{U}_p - I\|_F/\sqrt{n}, \|A - \widehat{U}_p\widehat{H}\|_F/\|A\|_F$ are both $\mathcal{O}(u)$ [47]. Backward stability analysis for general iterations for the polar decomposition is given in [47], which shows that the computed polar decomposition is backward stable if two conditions are satisfied: (i) Each iterate is backward-forward stable, i.e., the computed approximant $\widehat{Y}$ to $Y = f(X)$ satisfies $\widehat{Y} = f(\widetilde{X}) + \epsilon\|\widehat{Y}\|_2$ where $\widetilde{X} = \widehat{X} + \epsilon\|\widehat{X}\|_2$, where $\|\epsilon\| = \mathcal{O}(u)$; this means $\widehat{Y}$ is a slightly wrong output of a slightly wrong input. (ii) The function $f(x)$ lies above $y = x$.

Of the two conditions, the second is more nonintuitive and indeed gives insights into some unstable iterations proposed in the literature. QDWH satisfies the two conditions and hence is backward stable if pivoting (row and column) is used for computing the QR factorizations.

For Zolo-pd, it is easy to verify that the second condition is satisfied, because the mapping function $f(x) = \hat{Z}_{2r+1}(x; \ell_i)$ is the best approximant to the sign function, and $y = x$ can be regarded as a member of $(2r + 1, 2r)$ rational functions. However, regarding the first condition, the presence of r QR factorizations seems to make the discussion nontrivial. Although experiments demonstrate the excellent backward stability of Zolo-pd, its proof therefore remains an open problem.

5.1.2. Comparison with standard algorithms. Table 5.1 compares Zolo-pd with QDWH, the scaled Newton iteration [31, Ch. 8], and the SVD-based method (compute $A = U\Sigma V^*$, then obtain $U_p = UV^*$), the three most practical algorithms for the polar decomposition of $A \in \mathbb{C}^{m\times n}$ (we need $m = n$ for the scaled Newton iteration as it is applicable only to nonsingular matrices). It summarizes the backward stability, the dominant type of operation, the maximum iteration count and arithmetic cost in flops required for $\kappa_2(A) \le 10^{16}$; for well-conditioned matrices the flop count is lower. The arithmetic cost for QDWH is taken from the flop count in [48], and we can also derive that of Zolo-pd similarly (in which we took $r = 8$). The parenthesized entry at the bottom of the table shows the arithmetic cost along the critical path when the r QR and Cholesky factorizations in (5.1), (5.2) are computed in parallel. The arithmetic cost of the scaled Newton iteration assumes that matrix inverses are computed in the standard way based on LU factorization with partial pivoting.

Zolo-pd requires more arithmetic cost than QDWH and scaled Newton, by about a factor 3. However, along the critical path it requires the fewest flops, so in a parallel implementation we expect Zolo-pd to be the fastest.

TABLE 5.1
Comparison of algorithms for the polar decomposition.

	Zolo-pd	QDWH	scaled Newton	SVD-based
Backward stability	$(\surd)$	$\surd$	$\surd^a$	$\surd$
Dominant operation	QR	QR	inversion	bidiagonalization
Max. # iterations	2	6	9	
Arithmetic cost[b]	$64mn^2 + \frac{8}{3}n^3$	$22mn^2 + \frac{4}{3}n^3$	$18n^3$	$8mn^2 + 20n^3$
	$(8mn^2 + \frac{1}{3}n^3)$			

[a] To prove backward stability of scaled Newton [14, 47], matrix inverses need to be computed in a mixed backward-forward stable manner, i.e., $fl(X^{-1}) = (X + \Delta X)^{-1} + \Delta Y$ where $\|\Delta X\|/\|X\|, \|\Delta Y\|/\|fl(X^{-1})\|$ are both $\mathcal{O}(u)$. Using the commonly employed LU with partial pivoting for inversion (when matrix inversion is ever done; scaled Newton is one rare example) this condition is not guaranteed. The parenthesized $(\surd)$ means that stability is observed numerically, but not yet established theoretically; we conjecture that they are stable.

[b] For computing U_p: an additional $2mn^2$ flops is needed to compute $H = (U_p^*A + (U_p^*A)^*)/2$.

5.2. Symmetric eigendecomposition. Algorithm 5.1 is the pseudocode of Zolo-eig for computing the eigendecomposition of a symmetric matrix $A = A^*$. As we described in Section 1.2, here the polar decomposition algorithm Zolo-pd is used to split the positive and negative eigenspaces via the matrix sign function. Mapping the singular values to 1 in Zolo-pd corresponds to mapping the eigenvalues to ± 1.

Algorithm 5.2 Zolo-eig: compute a symmetric eigendecomposition $A = V\Lambda V^*$

Input: A symmetric (or Hermitian) matrix $A = A^*$.
Output: V, Λ such that $A = V\Lambda V^*$, where $V^*V = I_n$ and Λ is diagonal.

1: Initialize $V = I_n$.
2: Choose σ, estimate of the median of the eigenvalues of A.
3: Compute polar factor U_p of $A - \sigma I = U_p H$ by Zolo-pd (Algorithm 5.1).
4: Compute V_1 such that $\frac{1}{2}(U_p + I) = V_1 V_1^*$ via subspace iteration, then form a unitary matrix $[V_1\ V_2]$ by a full QR decomposition of V_1. Update $V := V[V_1\ V_2]$.
5: Compute $A_1 = V_1^* A V_1$ and $A_2 = V_2^* A V_2$.
6: Recursively repeat steps 2–5 with $A \leftarrow A_1, V \leftarrow V_1$ and $A \leftarrow A_2, V \leftarrow V_2$ until A_1, A_2 are 1×1, the eigenvalues Λ.

One run of steps 2-5 in Algorithm 5.2 represents one application of V_i in (1.4). In an actual implementation of Zolo-eig, one may additionally compute $E = V_1^* A V_2$ and ensure that it is negligible, i.e., $\|E\|_F = \mathcal{O}(u\|A\|)$.

For computing the shifts σ in QDWH-eig we adopted the method in [48] of taking the median of the diagonal elements. An unlucky choice of σ results in large condition number, which affects the degree r but not the stability of our Zolotarev-based algorithms. If after the shift $A := A - \sigma I$ the condition number $\kappa_2(A) \gg n$ then we can slightly alter σ until $\kappa_2(A) = \mathcal{O}(n)$. Other implementation issues, including executing subspace iteration and estimating $\sigma_{\max}(A)$ and $\sigma_{\min}(A)$, are the same as in the QDWH-based algorithms, and we refer to [48] for details. Note that two-step convergence is attained as long as $\alpha > \sigma_{\max}(A)$ and $\ell < \sigma_{\min}(A/\alpha)$, so "safe" estimates that satisfy these are preferred.

5.2.1. Backward stability. As mentioned in the introduction, the backward stability of Zolo-eig depends crucially on that of Zolo-pd. Specifically, it was shown in [48] that if the polar decomposition and the orthonormal column space V_1 of the orthogonal projection matrix $V_1 V_1^*$ are obtained in a backward stable manner throughout the spectral divide-and-conquer process, then the computed eigendecomposition is backward stable, that is, $\|A - \widehat{V}\widehat{\Lambda}\widehat{V}^*\|_F / \|A\|_F$ and $\|\widehat{V}^*\widehat{V} - I_n\|_F / \sqrt{n}$ are both $\mathcal{O}(u)$.

5.2.2. Comparison with standard algorithms. Table 5.2 compares the spectra-divide-and-conquer algorithms Zolo-eig, QDWH-eig, IRS (implicit repeated squaring [7]), and ZZY [62], along with the standard algorithm that performs tridiagonalization followed by the symmetric tridiagonal QR algorithm [22, § 8.3],[50, Ch. 8]. The algorithms compute both the eigenvalues and eigenvectors. In addition to the information shown in Table 5.1, Table 5.2 shows whether the algorithm minimizes communication in the asymptotic sense.

Following [48], the arithmetic cost of QDWH-eig and Zolo-eig is obtained assuming that the splitting points σ are chosen such that $\kappa_2(A - \sigma I) \le 10^5$, for which $r = 5$ is sufficient as discussed in Section 3.5; we can take a different σ if this does not hold. Note that the arithmetic cost along the critical path of Zolo-eig in a parallel implementation is comparable to that of the standard algorithm.

TABLE 5.2
Comparison of algorithms for symmetric eigendecomposition.

	Zolo-eig	QDWH-eig	IRS [7]	ZZY [62]	standard
Min. communication?	√	√	√	×	×
Backward stability	(√)	√	conditional	(√)	√
Max. # iterations	2	6	53	53	
Arithmetic cost	$54.8n^3$ ($15.7n^3$)	$27n^3$	$\approx 720n^3$	$\approx 370n^3$	$9n^3$

5.3. SVD. Algorithm 5.3 is the pseudocode of Zolo-SVD.

Algorithm 5.3 Zolo-SVD: compute the SVD of a general matrix A

Input: $A \in \mathbb{C}^{m\times n}$ with $m \geq n$.
Output: U, Σ, V such that $A = U\Sigma V^*$, where $U^*U = V^*V = I_n$ and Σ is diagonal.
1: Compute the polar decomposition $A = U_pH$ via Zolo-pd (Algorithm 5.1).
2: Compute the symmetric eigendecomposition $H = V\Sigma V^*$ via Zolo-eig (Algorithm 5.2), and compute $U = U_pV$.

As is the case with QDWH-SVD, taking an initial QR factorization is recommended for Zolo-SVD when m is much larger than n. Zolo-SVD is also able to compute the SVD for rank-deficient matrices via a partial isometry instead of the unitary polar factor in step 1, as described in [48, § 5.5].

5.3.1. Backward stability. Zolo-SVD can be shown to be backward stable provided that its building components Zolo-pd and Zolo-eig are backward stable, by the exact same argument as in [48, §4.1]. Thus in addition to being fundamental for the algorithm derivation, the polar decomposition is key also in the stability analysis.

5.3.2. Comparison with standard algorithms. Table 5.3 compares four SVD algorithms: Zolo-SVD, QDWH-SVD, IRS, and the standard algorithm that performs bidiagonalization followed by bidiagonal QR [22, § 8.6]. The arithmetic cost shows the flop counts for a square $n \times n$ matrix A, and since for Zolo-SVD and QDWH-SVD it depends on the condition number $\kappa_2(A)$, we show the arithmetic cost in the range $\kappa_2(A) = 1.1$–10^{16}. The arithmetic cost along the critical path of Zolo-SVD is about the same as that of the standard algorithm, or less.

TABLE 5.3
Comparison of algorithms for the SVD.

	Zolo-SVD	QDWH-SVD	IRS	standard
Min. communication?	√	√	√	×
Backward stability	(√)	√	conditional	√
Max. # iterations	2	6	53	
Arithmetic cost ($m = n$)	$61.1n^3$–$124n^3$ ($22n^3$–$27n^3$)	$35n^3$–$52n^3$	$\approx 5700n^3$	$26n^3$

6. Numerical experiments. We present numerical experiments to examine the performance of Zolo-pd, Zolo-eig and Zolo-SVD, and to compare them with the QDWH-based and standard algorithms. The experiments here are all sequential: in particular, the r QR factorizations are not computed independently, and nor is a communication-minimizing implementation used. These are left as future work.

All the experiments were carried out in MATLAB version R2013a on a machine with an Intel Xeon processor with eight cores, sixteen threads, and 64GB RAM, using IEEE double-precision arithmetic.

6.1. Polar decomposition. We first compare Zolo-pd with the two most practical iterations among the existing algorithms: QDWH and the scaled Newton iteration [31, p. 205] with the scaling due to Byers and Xu [14], shown in Table 6.1 as "Newton". We generate n-by-n matrices with $n = 20000$ by forming $A = U\Sigma V^*$, where $U, V \in \mathbb{R}^{n\times n}$ are random orthogonal matrices and Σ is a diagonal matrix of singular values, which form an arithmetic sequence (the distribution of singular values has little effect on the performance). Table 6.1 shows the iteration counts "iter", backward error $\|\widehat{U}_p\widehat{H} - A\|_F/\|A\|_F$ "berr", orthogonality measure $\|\widehat{U}_p^*\widehat{U}_p - I\|_F/\sqrt{n}$ "orth", and the runtime in seconds. The numbers in parentheses for Zolo-pd in the "iter" row show the degree r of the polynomials $P(x), Q(x)$, and those in "time" are the runtime of Zolo-pd along the critical path, i.e., assuming the r independent terms in (5.1) and (5.2) are computed in parallel, it counts only the runtime of the one that took the longest among the r QR (or Cholesky) factorizations in (5.1) and (5.2).

TABLE 6.1
Performance comparison of polar decomposition algorithms, $A \in \mathbb{R}^{20000\times 20000}$.

$\kappa_2(A)$		1.1	1.5	10	10^5	10^{10}	10^{15}
iter	Zolo-pd	1 (4)	1 (6)	2 (3)	2 (5)	2 (7)	2 (8)
	QDWH	3	3	4	5	6	6
	Newton	4	5	6	8	8	9
berr	Zolo-pd	1.6e-15	2.1e-15	1.5e-15	1.6e-15	1.7e-15	2.1e-15
	QDWH	1.1e-15	1.2e-15	1.2e-15	1.5e-15	1.4e-15	1.4e-15
	Newton	2.4e-13	2.5e-13	2.5e-13	2.5e-13	2.4e-13	2.4e-13
orth	Zolo-pd	1.5e-15	2.0e-15	1.1e-15	1.0e-15	1.1e-15	1.7e-15
	QDWH	7.7e-16	1.1e-15	8.9e-16	1.1e-15	7.6e-16	1.1e-15
	Newton	1.7e-13	1.7e-13	1.7e-13	1.7e-13	1.7e-13	1.7e-13
time	Zolo-pd	475 (164)	682 (164)	680 (267)	1604 (369)	2249 (373)	2539 (370)
	QDWH	353	355	448	659	877	878
	Newton	343	418	483	626	629	706

We observe the following.

- As predicted by the theory, Zolo-pd requires just two iterations for convergence. The number of iterations for the other two methods also accurately reflect the theory, see [14], [31, p. 206] for scaled Newton and [46] for QDWH.
- For large $\kappa_2(A)$, the runtime of Zolo-pd becomes much longer than those of the other two methods. This is because r grows, though very gradually like $\log\log(\kappa_2(A))$, and hence more QR factorizations are needed per iteration, which are computed sequentially here. However, along the critical path, its runtime is much shorter, and independent of r for $\kappa_2(A) \geq 10^2$ (in which case the critical path is essentially one QR and Cholesky).
- Zolo-pd and QDWH give excellent backward error and orthogonality measure. Those of scaled Newton are about two orders of magnitude larger.

6.2. Symmetric eigendecomposition.

6.2.1. Spectral divide-and-conquer algorithms. We first compare spectral divide-and-conquer algorithms: Zolo-eig, QDWH-eig, the general algorithm applicable to generalized eigenproblems due to Ballard, Demmel and Dumitriu [7], which we call

IRS (Implicit Repeated Squaring), and the algorithm QUAD in [62], which we call ZZY. These methods can be implemented in a communication-minimizing manner.

We set the matrix size to $n = 100$ and generate symmetric matrices $A = V\Lambda V^T$, where $V \in \mathbb{R}^{n\times n}$ is a random orthogonal matrix and $\Lambda = \mathrm{diag}(1, r, r^2, \ldots, r^{n-1})$ with $r = -\kappa^{-1/(n-1)}$, in which $\kappa = \kappa_2(A)$ is the prescribed condition number, which we set to $10^2, 10^5$ and 10^{15}. The eigenvalue of A closest to 0 is κ^{-1}.

Here we consider computing an invariant subspace V_1 corresponding to the positive eigenvalues of A. To do so we apply one recursion of Zolo-eig (steps 1–4 of Algorithm 5.2) and QDWH-eig on A.

We generated 100 matrices for each case $\kappa = 10^2, 10^8, 10^{15}$, and show in Table 6.2 the maximum and minimum values of the iteration counts, shown as "iter", along with the backward error $\|E\|_F/\|A\|_F$ where $[\widehat{V}_1\ \widehat{V}_2]^*A[\widehat{V}_1\ \widehat{V}_2] = \left[\begin{smallmatrix} A_1 & E^* \\ E & A_2 \end{smallmatrix}\right]$, shown as "berr". The orthogonality measure $\|\widehat{V}_1^*\widehat{V}_1 - I_k\|_F/\sqrt{n}$ was $\mathcal{O}(u)$ for all the methods.

TABLE 6.2
Iteration count and residual of spectral divide-and-conquer algorithms.

$\kappa_2(A)$		10^2		10^8		10^{15}	
		min	max	min	max	min	max
iter	Zolo-eig	2 (3)	2 (3)	2 (6)	2 (6)	2 (8)	2 (8)
	QDWH-eig	4	5	5	5	6	6
	ZZY	12	12	32	32	55	56
	IRS	12	13	32	32	54	55
berr	Zolo-eig	5.6e-16	6.1e-16	5.8e-16	6.5e-16	6.4e-16	7.3e-16
	QDWH-eig	8.5e-16	9.4e-16	8.5e-16	9.7e-16	8.4e-16	9.8e-16
	ZZY	1.6e-15	1.9e-15	2.6e-15	2.9e-15	2.9e-15	4.1e-15
	IRS	2.1e-15	2.9e-14	2.4e-13	3.3e-12	3.8e-10	4.0e-8

Observations:

- Zolo-eig always converges in two iterations. QDWH-eig converges within six iterations, whereas ZZY and IRS need many more iterations, especially in the difficult cases where $\kappa_2(A)$ is large.
- Zolo-eig and QDWH-eig performed in a backward stable manner throughout.

This illustrates Zolo-eig and QDWH-eig are significantly superior to the other spectral divide-and-conquer algorithms as predicted also by Table 5.2.

6.2.2. Comparison with conventional algorithms for the full symmetric eigendecomposition. We now compare algorithms for computing the full eigendecomposition of a symmetric matrix. We compare Zolo-eig, QDWH-eig and MATLAB's built-in function `eig`, which is based on reduction to tridiagonal form followed by the divide-and-conquer algorithm [26]. Here we generated symmetric matrices A as follows: form a random matrix B by the MATLAB function $B =$`randn(n)`, then let $A = \frac{1}{2}(B + B^*)$. Below we report the average of three runs.

Table 6.3 shows the backward error $\|\widehat{V}\widehat{\Lambda}\widehat{V}^T - A\|_F/\|A\|_F$. While all the methods are backward stable, the backward errors of QDWH-eig and Zolo-eig are smaller than those of `eig`, by more than a factor 3.

Table 6.4 shows the orthogonality measure $\|\widehat{V}^T\widehat{V} - I\|_F/\sqrt{n}$. We see that those of Zolo-eig and QDWH-eig are much smaller than `eig`. This improvement is largely due to the Newton–Schulz postprocessing.

The above behavior was not peculiar to this class of matrices, and was observed in

TABLE 6.3
Backward error $\|A - \widehat{V}\widehat{\Lambda}\widehat{V}^T\|_F/\|A\|_F$.

n	4000	8000	12000	16000	20000
QDWH-eig	2.4e-15	2.8e-15	3.2e-15	3.6e-15	3.8e-15
Zolo-eig	2.4e-15	2.9e-15	3.2e-15	3.6e-15	3.8e-15
MATLAB `eig`	7.6e-15	1.0e-14	1.3e-14	1.4e-14	1.6e-14

TABLE 6.4
Orthogonality measure of $\widehat{V}$: $\|\widehat{V}^T\widehat{V} - I\|_F/\sqrt{n}$.

n	4000	8000	12000	16000	20000
Zolo-eig	8.0e-16	8.4e-16	8.6e-16	8.8e-16	9.0e-16
QDWH-eig	8.0e-16	8.4e-16	8.6e-16	8.8e-16	9.0e-16
MATLAB `eig`	6.4e-15	8.7e-15	1.1e-14	1.2e-14	1.4e-14

all our experiments. These results suggest that the stability of Zolo-eig is comparable to QDWH-eig, and is typically better than conventional algorithms. Of course, all three algorithms are stable and the difference in backward error is a small constant that is usually insignificant.

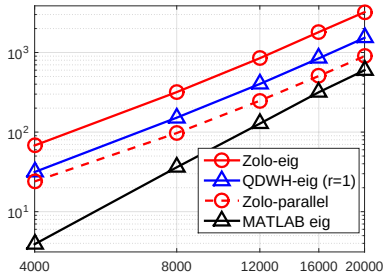


FIG. 6.1. *Eigendecomposition runtime for varying matrix size.*

Figure 6.1 shows the SVD runtime. Zolo-parallel shows the runtime of Zolo-eig along the critical path (as in Zolo-pd, accounting only for the longest of the r QR factorizations). Observe that while Zolo-eig and Zolo-parallel are slower than MATLAB's `eig`, they seem to scale slightly better as n grows.

6.3. SVD algorithms. We now turn to the SVD algorithms Zolo-SVD, QDWH-SVD and MATLAB's `svd` (based on bidiagonal reduction and divide-and-conquer [25]). We generate test matrices by forming $n \times n$ matrices $A = U\Sigma V^*$, where $U, V \in \mathbb{R}^{n\times n}$ are random orthogonal matrices and the singular values are uniformly distributed.

Varying matrix sizes. We set $\kappa_2(A) = 10^5$ and varied the matrix size n. Tables 6.5 and 6.6 show the backward error $\|A - \widehat{U}\widehat{\Sigma}\widehat{V}^T\|_F/\|A\|_F$ and orthogonality measure $\max(\|\widehat{U}^T\widehat{U} - I\|_F/\sqrt{n}, \|\widehat{V}^T\widehat{V} - I\|_F/\sqrt{n})$. The same comments as for the symmetric eigendecomposition apply: while all algorithms give acceptably small backward error and orthogonality measure, those of Zolo-SVD and QDWH-SVD are notably better than the standard algorithm MATLAB `svd`. The stability behavior is largely independent of the condition number $\kappa_2(A)$.

Figure 6.2 shows the SVD runtime for varying matrix size, and $\kappa_2(A) = \{1.5, 10^5\}$.

TABLE 6.5
Backward error for SVD $\|A - \widehat{U}\widehat{\Sigma}\widehat{V}^T\|_F/\|A\|_F$.

n	4000	8000	12000	16000	20000
Zolo-SVD	2.4e-15	2.9e-15	3.2e-15	3.6e-15	3.8e-15
QDWH-SVD	2.4e-15	2.7e-15	3.0e-15	3.3e-15	3.5e-15
MATLAB `svd`	7.9e-15	1.0e-14	1.3e-14	1.4e-14	1.6e-14

TABLE 6.6
Orthogonality of computed $\widehat{U}, \widehat{V}$: $\max(\|\widehat{U}^T\widehat{U} - I\|_F/\sqrt{n}, \|\widehat{V}^T\widehat{V} - I\|_F/\sqrt{n})$.

n	4000	8000	12000	16000	20000
Zolo-SVD	8.1e-16	8.4e-16	8.6e-16	8.8e-16	9.0e-16
QDWH-SVD	8.1e-16	8.4e-16	8.6e-16	8.8e-16	9.0e-16
MATLAB `svd`	7.4e-15	9.7e-15	1.2e-14	1.4e-14	1.5e-14

As before, Zolo-parallel shows the runtime of Zolo-SVD along the critical path. Zolo-SVD and Zolo-parallel scale slightly better than MATLAB's `svd` as n grows, and for $n \geq 10000$ Zolo-parallel becomes faster when A is well-conditioned, suggesting that Zolo-SVD can outperform a standard SVD algorithm with an optimized implementation.

A big bulk of the runtime of a standard, reduction-based algorithm is consumed in the reduction step [61] (to tridiagonal or bidiagonal form), which becomes a communication bottleneck in a parallel implementation. Spectral divide-and-conquer algorithms overcome this issue by bypassing the reduction and always working with the whole matrix.

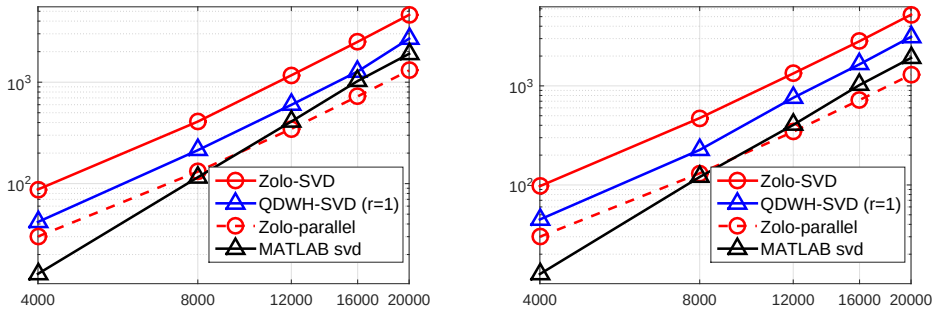
The above experiments are only with square nonsingular matrices, but all the algorithms are applicable to rectangular and rank-deficient matrices. Indeed, the experiments in [48] illustrate that QDWH-eig tends to compute the rank of rank-deficient matrices or small singular values more accurately, and we observed the same behavior with Zolo-SVD.

6.4. Summary of numerical experiments. The results of our experiments can be summarized as follows.

- Zolo-based ($r \geq 1$) algorithms have excellent numerical backward stability, comparable to that of QDWH-based ($r = 1$).
- Zolo-based and QDWH-based algorithms are generally much faster and more stable than other spectral divide-and-conquer algorithms.
- On a sequential implementation, the MATLAB functions `eig` and `svd` employing divide-and-conquer following reduction to tridiagonal or bidiagonal were the fastest, for the eigendecomposition and the SVD, respectively. However, Zolo-eig and Zolo-SVD are predicted to have competitive speed when implemented in parallel.

On massively parallel computing architectures where the communication cost dominates arithmetic cost, it is expected that the high parallelizability and excellent stability of Zolo-based algorithms provide an attractive alternative to the standard ones.

7. Summary and discussion. Zolo-pd is a matrix iteration that draws heavily on rational approximation theory. Rational approximation, in turn, has close connections to numerical contour integration in the complex plane: Hale, Higham and Trefethen [28] propose an algorithm for computing matrix functions via numerical contour integration combined with conformal mapping to improve the region of ana-



$\kappa_2(A) = 1.5.$ $\kappa_2(A) = 10^5.$

FIG. 6.2. *SVD runtime for varying matrix size.*

lyticity, and they show that numerical contour integration can be regarded as a rational approximation, see also [56]. In particular, when the function is $\sqrt{x}$, their contour integration algorithm becomes closely related to Zolotarev's rational approximation to the square root (obtained by $\frac{P_r(x)}{Q_r(x)} \approx \sqrt{x}$, where $Z_{2r+1}(x;\ell) = x\frac{P_r(x^2)}{Q_r(x^2)} \approx \mathrm{sign}(x)$ is the Zolotarev function for the sign function), applicable for computing $A^{1/2}$ for a matrix with positive eigenvalues. We can adjust this approach for the sign function to compute the unitary polar factor U_p of A. This results in a type $(2r+1, 2r)$ approximant

$$U_p \approx A + \sum_{j=1}^{r} a_j A(A^*A + c_{2j-1}I)^{-1}, \tag{7.1}$$

where a_j, c_{2j-1} are as in (4.6). This is essentially the first iteration of Zolo-pd.

Compared with this contour integral-based derivation of (7.1), our algorithm Zolo-pd makes three improvements. First, by using the QR factorization to avoid matrix inversions for computing matrices of the form $X(X^*X+c_{2i-1}I)^{-1}$, we improve the numerical stability significantly: a direct implementation of (7.1) (or one using Cholesky factorization as in (4.7)) gives poor stability when $\kappa_2(A) \gg 1$. Second, by allowing the algorithm to iterate for two steps, we reduce the arithmetic cost dramatically (from $r > 140$ once to $r = 8$ twice). Finally, Zolo-pd resolves the numerical instability when r is large, as observed in Section 3.4.

We note that the idea of allowing for the second step is highly nontrivial from the viewpoint of contour integration (or its relation to rational approximation), whereas our derivation of Zolo-pd as an iterative algorithm for the polar decomposition makes it just natural.

Acknowledgments. We would like to thank Stefan Güttel, Nick Higham, Françoise Tisseur, and Nick Trefethen for their valuable comments. We are grateful to Ilse Ipsen and the referees for their many constructive suggestions that significantly improved the presentation of the paper.

REFERENCES

[1] M. Abramowitz and I. A. Stegun. *Handbook of Mathematical Functions*. Dover, NY, 1965.
[2] N. I. Akhiezer. *Elements of the Theory of Elliptic Functions*. AMS, Providence, RI, 1990.
[3] N. I. Akhiezer. *Theory of Approximation*. Dover, NY, 1992.
[4] Z. Bai and J. Demmel. Using the matrix sign function to compute invariant subspaces. *SIAM J. Matrix Anal. Appl.*, 19:205–225, 1998.
[5] Z. Bai, J. Demmel, J. Dongarra, A. Ruhe, and H. van der Vorst. *Templates for the Solution of Algebraic Eigenvalue Problems: A Practical Guide*. SIAM, Philadelphia, PA, 2000.
[6] Z. Bai, J. Demmel, and M. Gu. An inverse free parallel spectral divide and conquer algorithm for nonsymmetric eigenproblems. *Numer. Math.*, 76(3):279–308, 1997.
[7] G. Ballard, J. Demmel, and I. Dumitriu. Minimizing communication for eigenproblems and the singular value decomposition. Technical Report 237, LAPACK Working Note, 2010.
[8] G. Ballard, J. Demmel, O. Holtz, and O. Schwartz. Minimizing communication in numerical linear algebra. *SIAM J. Matrix Anal. Appl.*, 32(3):866–901, 2011.
[9] G. Ballard, J. Demmel, and N. Knight. Avoiding communication in successive band reduction. Technical Report UCB/EECS-2013-131, University of California, Berkeley, 2013.
[10] A. N. Beavers, Jr. and E. D. Denman. A computational method for eigenvalues and eigenvectors of a matrix with real eigenvalues. *Numer. Math.*, 21:389–396, 1973.
[11] B. Beckermann. Optimally scaled Newton iterations for the matrix square root. Talk at workshop "Advances in Matrix Functions and Matrix Equations", Manchester, UK, 2013.
[12] D. Braess. On rational approximation of the exponential and the square root function. In P. R. et al., editor, *Rational Approximation and Interpolation*, volume 1105 of *Lecture Notes in Mathematics*, pages 89–99, Berlin-Heidelberg-New York, 1984. Springer.
[13] D. Braess. *Nonlinear Approximation Theory*. Springer, Berlin-Heidelberg-New York, 1986.
[14] R. Byers and H. Xu. A new scaling for Newton's iteration for the polar decomposition and its backward stability. *SIAM J. Matrix Anal. Appl.*, 30:822–843, 2008.
[15] B. C. Carlson and J. L. Gustafson. Asymptotic expansion of the first elliptic integral. *SIAM J. Math. Anal.*, 16(5):1072–1092, 1985.
[16] T. F. Cox and M. A. A. Cox. *Multidimensional Scaling*. CRC Press, FL, 2nd edition, 2000.
[17] I. S. Dhillon and B. N. Parlett. Orthogonal eigenvectors and relative gaps. *SIAM J. Matrix Anal. Appl.*, 25:858–899, 2004.
[18] V. Druskin, S. Güttel, and L. Knizhnerman. Near-optimal perfectly matched layers for indefinite Helmholtz problems. MIMS EPrint, The University of Manchester, UK, 2013. To appear in SIAM Review.
[19] S. W. Gaaf and M. E. Hochstenbach. Probabilistic bounds for the matrix condition number with extended lanczos bidiagonalization. *SIAM J. Sci. Comp.* to appear.
[20] A. George and K. Ikramov. Is the polar decomposition finitely computable? *SIAM J. Matrix Anal. Appl.*, 17:348–354, 1996.
[21] A. George and K. Ikramov. Addendum: Is the polar decomposition finitely computable? *SIAM J. Matrix Anal. Appl.*, 18:264, 1997.
[22] G. H. Golub and C. F. Van Loan. *Matrix Computations*. The Johns Hopkins University Press, Baltimore, MD, 4th edition, 2013.
[23] A. A. Gončar. Zolotarev problems connected with rational functions. *Math. USSR Sb.*, 7(4):623–635, 1969.
[24] J. C. Gower and G. B. Dijksterhuis. *Procrustes Problems*. Oxford University Press, Oxford, UK, 2004.
[25] M. Gu and S. C. Eisenstat. A divide-and-conquer algorithm for the bidiagonal SVD. *SIAM J. Matrix Anal. Appl.*, 16(1):79–92, 1995.
[26] M. Gu and S. C. Eisenstat. A divide-and-conquer algorithm for the symmetrical tridiagonal eigenproblem. *SIAM J. Matrix Anal. Appl.*, 16(1):172–191, 1995.
[27] S. Güttel, E. Polizzi, P. Tang, and G. Viaud. Zolotarev quadrature rules and load balancing for the FEAST eigensolver. MIMS EPrint 2014.39, The University of Manchester, UK, 2014.
[28] N. Hale, N. J. Higham, and L. N. Trefethen. Computing A^α, $\log(A)$, and related matrix functions by contour integrals. *SIAM J. Numer. Anal.*, 46(5):2505–2523, 2008.
[29] N. Halko, P.-G. Martinsson, and J. A. Tropp. Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. *SIAM Rev.*, 53(2):217–288, 2011.
[30] P. Henrici. *Applied and Computational Complex Analysis. Vol. 1, Power Series, Integration, Conformal Mapping, Location of Zeros*. Wiley, 1974.
[31] N. J. Higham. *Functions of Matrices: Theory and Computation*. SIAM, PA, 2008.
[32] N. J. Higham and P. Papadimitriou. A parallel algorithm for computing the polar decomposition. *Parallel Comput.*, 20:1161–1173, 1994.
[33] N. J. Higham. Computing the polar decomposition-with applications. *SIAM J. Sci. Stat.*

Comp., 7(4):1160–1174, 1986.
[34] N. J. Higham. The matrix sign decomposition and its relation to the polar decomposition. *Linear Algebra Appl.*, 212/213:3–20, 1994.
[35] M. E. Hochstenbach. Eigenvalue Tools. Available via `http://www.win.tue.nl/~hochsten/eigenvaluetools/`.
[36] S. Huss-Lederman, E. S. Quintana-Ortí, X. Sun, and Y.-J. Y. Wu. Parallel spectral division using the matrix sign function for the generalized eigenproblem. *Int. J. High Speed Com.*, 11(01):1–14, 2000.
[37] B. Iannazzo. The geometric mean of two matrices from a computational viewpoint. *arXiv:1201.0101*, 2011.
[38] T. Kaneko, S. Fiori, and T. Tanaka. Empirical arithmetic averaging over the compact stiefel manifold. *IEEE Trans. Signal Process.*, 61(4):883–894, 2013.
[39] A. D. Kennedy. Fast evaluation of Zolotarev coefficients. In *Proceedings of the Third International Workshop on Numerical Analysis and Lattice QCD*, pages 169–189, 2003.
[40] A. D. Kennedy. Approximation theory for matrices. *Nucl. Phys. B–Proc. Sup.*, 128:107–116, 2004.
[41] C. Kenney and A. Laub. A hyperbolic tangent identity and the geometry of Padé sign function iterations. *Numer. Algorithms*, 7:111–128, 1994. 10.1007/BF02140677.
[42] C. Kenney and A. Laub. The matrix sign function. *IEEE Trans. Automat. Control*, 40(8):1330–1348, 1995.
[43] B. Laszkiewicz and K. Zietak. Numerical experiments with algorithms for the ADI and Zolotarev coefficients. *Appl. Math. Comput.*, 206(1):298–312, 2008.
[44] J. Mao. Optimal orthonormalization of the strapdown matrix by using singular value decomposition. *Computers Math. Applic.*, 12(3):353–362, 1986.
[45] Y. Nakatsukasa. *Algorithms and Perturbation Theory for Matrix Eigenvalue Problems and the Singular Value Decomposition*. PhD thesis, University of California, Davis, 2011.
[46] Y. Nakatsukasa, Z. Bai, and F. Gygi. Optimizing Halley's iteration for computing the matrix polar decomposition. *SIAM J. Matrix Anal. Appl.*, 31(5):2700–2720, 2010.
[47] Y. Nakatsukasa and N. J. Higham. Backward stability of iterations for computing the polar decomposition. *SIAM J. Matrix Anal. Appl.*, 33(2):460–479, 2012.
[48] Y. Nakatsukasa and N. J. Higham. Stable and efficient spectral divide and conquer algorithms for the symmetric eigenvalue decomposition and the SVD. *SIAM J. Sci. Comp*, 35(3):A1325–A1349, 2013.
[49] I. Ninomiya. Best rational starting approximations and improved Newton iteration for the square root. *Math. Comp.*, 24(110):391–404, 1970.
[50] B. N. Parlett. *The Symmetric Eigenvalue Problem*. SIAM, Philadelphia, PA, 1998.
[51] P. P. Petrushev and V. A. Popov. *Rational Approximation of Real Functions*. Cambridge University Press, Cambridge, United Kingdom, 2011.
[52] M. J. D. Powell. *Approximation Theory and Methods*. Cambridge University Press, Cambridge, United Kingdom, 1981.
[53] J. D. Roberts. Linear model reduction and solution of the algebraic Riccati equation by use of the sign function. *Int. J. Control*, 32(4):677–687, 1980.
[54] H. Rutishauser. Betrachtungen zur Quadratwurzeliteration. *Monatsh. Math.*, 67(5):452–464, 1963.
[55] D. E. Sukkari, H. Ltaief, and D. E. Keyes. A high performance QDWH-SVD solver using hardware accelerators. Technical report, KAUST Repository, April 2015.
[56] L. N. Trefethen, J. A. C. Weideman, and T. Schmelzer. Talbot quadratures and rational approximations. *BIT*, 46(3):653–670, 2006.
[57] L. N. Trefethen. *Approximation Theory and Approximation Practice*. SIAM, PA, 2013.
[58] L. N. Trefethen and J. A. C. Weideman. The exponentially convergent trapezoidal rule. *SIAM Rev.*, 56(3):385–458, 2014.
[59] J. van den Eshof. *Nested Iteration Methods for Nonlinear Matrix Problems*. PhD thesis, Proefschrift Universiteit Utrecht, 2003.
[60] J. van den Eshof, T. Lippert, A. Frommer, K. Schilling, and H. van der Vorst. Numerical methods for the QCD overlap operator: I. sign-function and error bounds. *Comput. Phys. Comm.*, 146:203–224, 2002.
[61] F. V. Zee, R. van de Geijn, and G. Quintana-Orti. Restructuring the tridiagonal and bidiagonal QR algorithm for performance. *ACM Trans. Math. Soft.*, 40(3):18:1–18:34, 2014.
[62] Z. Zhang, H. Zha, and W. Ying. Fast parallelizable methods for computing invariant subspaces of Hermitian matrices. *J. Comput. Math.*, 25(5):583–594, 2007.
[63] E. I. Zolotarev. Application of elliptic functions to questions of functions deviating least and most from zero. *Zap. Imp. Akad. Nauk. St. Petersburg,*, 30(5), 1877. Reprinted in his

Collected Works, Vol. II, Izdat. Akad. Nauk SSSR, Moscow, 1932, pp. 1–59. In Russian.