Electronic Transactions on Numerical Analysis.
Volume 2, pp. 1-21, March 1994.
Copyright © 1994, Kent State University.
ISSN 1068-9613.

AN IMPLICITLY RESTARTED LANCZOS METHOD FOR LARGE SYMMETRIC EIGENVALUE PROBLEMS [*]

D. CALVETTI [†], L. REICHEL [‡], AND D.C. SORENSEN [§]

Dedicated to Wilhelm Niethammer on the occasion of his 60th birthday.

Abstract. The Lanczos process is a well known technique for computing a few, say k, eigenvalues and associated eigenvectors of a large symmetric $n \times n$ matrix. However, loss of orthogonality of the computed Krylov subspace basis can reduce the accuracy of the computed approximate eigenvalues. In the implicitly restarted Lanczos method studied in the present paper, this problem is addressed by fixing the number of steps in the Lanczos process at a prescribed value, $k+p$, where p typically is not much larger, and may be smaller, than k. Orthogonality of the $k+p$ basis vectors of the Krylov subspace is secured by reorthogonalizing these vectors when necessary. The implicitly restarted Lanczos method exploits that the residual vector obtained by the Lanczos process is a function of the initial Lanczos vector. The method updates the initial Lanczos vector through an iterative scheme. The purpose of the iterative scheme is to determine an initial vector such that the associated residual vector is tiny. If the residual vector vanishes, then an invariant subspace has been found. This paper studies several iterative schemes, among them schemes based on Leja points. The resulting algorithms are capable of computing a few of the largest or smallest eigenvalues and associated eigenvectors. This is accomplished using only $(k+p)n + O((k+p)^2)$ storage locations in addition to the storage required for the matrix, where the second term is independent of n.

Key words. Lanczos method, eigenvalue, polynomial acceleration.

AMS subject classification. 65F15.

1. Introduction.

The Lanczos process is an effective method for computing a few eigenvalues and associated eigenvectors of a large symmetric matrix $A \in \mathbf{R}^{n\times n}$. This method only requires one to compute the action of the matrix on a vector through a matrix-vector product. Often this may be accomplished without explicit storage of the matrix. This property, along with a number of theoretical and computational features, contributes to the appeal of the Lanczos process. The basic Lanczos method can suffer from large storage requirements and from numerical difficulties caused by loss of orthogonality of the basis vectors generated. Efforts to remedy these shortcomings have been the focus of considerable research over the last two decades, see, e.g., [2, 3, 13, 19, 20, 21, 22, 29]. The Implicitly Restarted Lanczos (IRL) method studied in the present paper addresses some of these difficulties. This method is obtained by specializing the Implicitly Restarted Arnoldi (IRA) scheme, designed for the solution of nonsymmetric eigenvalue problems and presented in [30], to symmetric eigenvalue problems. The IRL method may be viewed as a truncation of the standard implicitly shifted QR-algorithm for dense symmetric eigenvalue problems. Numerical difficulties and storage problems normally associated with the Lanczos process are avoided because it is possible to maintain and update a numerically orthogonal set of basis vectors of pre-determined size. The algorithm is capable of computing a few,

[*] Received October 25, 1993. Accepted for publication February 21, 1994. Communicated by M. Eiermann.

[†] Department of Pure and Applied Mathematics, Stevens Institute of Technology, Hoboken, NJ 07030. E-mail: calvetti@mathrs1.stevens-tech.edu.

[‡] Department of Mathematics and Computer Science, Kent State University, Kent, OH 44242. This research was supported in part by NSF grant DMS-9205531. E-mail: reichel@mcs.kent.edu.

[§] Department of Mathematical Sciences, Rice University, Houston, TX 77251. This research was supported in part by NSF cooperative agreement CCR-8809615 and DOE contract DE-FG0f-91ER25103. E-mail: sorensen@rice.edu.

say k, of the largest or smallest eigenvalues and associated eigenvectors using only $(k+p)n + O((k+p)^2)$ storage locations, where the integer p typically is not much larger, and may in fact be smaller, than k, and where the second term is independent of n. This count of storage locations does not include the storage necessary for the matrix A.

The IRA and IRL methods are polynomial acceleration schemes and the rate at which eigenvalues and invariant subspaces are determined depends on the choice of accelerating polynomials. Following [30], we refer to these polynomials as polynomial filters. Their purpose is to force the initial vector into the appropriate invariant subspace. The polynomial filters are constructed and applied implicitly to the initial vector by carrying out a truncated version of the implicitly shifted QR-algorithm. The polynomial filters may be specified either by their zeros or by their expansion coefficients in a basis of Lanczos polynomials. In agreement with the terminology used for the QR-algorithm, we refer to these zeros also as "shifts". A natural choice of zeros are eigenvalues in the undesired part of the spectrum of a sequence of $(k+p)\times(k+p)$ symmetric tridiagonal matrices generated by the Lanczos process in the IRL method. We refer to these zeros as "exact shifts". It is shown in [30] that polynomial filters based on exact shifts yield convergence when a few of the smallest or a few of the largest eigenvalues of a symmetric matrix are desired. The method of proof in [30], however, does not display the rate of convergence, and computed examples in Section 5 of the present paper show that convergence can be very slow when exact shifts are applied.

The possible difficulties with exact shifts motivated us to study other shift selection strategies. In the present paper, we investigate the application of Leja points to the selection of shifts, i.e., the shifts are chosen to be Leja points for certain intervals on the real axis. We refer to these shifts as Leja shifts. We compare them with exact shifts, and demonstrate that Leja shifts can yield faster convergence of the IRL method than exact shifts.

We remark that polynomial acceleration for eigenvalue computation was first used by Flanders and Shortley [11], who applied Chebyshev polynomials. Subsequently polynomial acceleration has been employed frequently to enhance the performance of computational methods for finding a few selected eigenvalues and associated eigenvectors of large sparse matrices. Such schemes for the symmetric and nonsymmetric eigenvalue problems are discussed in, e.g., [15, 28, 29, 31]. The present paper considers the computation of a few extreme eigenvalues and associated eigenvectors of a large sparse symmetric matrix. Our scheme is of interest, e.g., for the computation of the smallest eigenvalues of a symmetric positive definite matrix when the matrix cannot be factored due to computer storage limitations. Applications of such eigenvalue problems are considered in [18, 23, 32].

This paper is organized as follows. In Section 2 we review the Lanczos factorization, the functional dependence of the residual vector on the initial vector, and necessary and sufficient conditions for an initial vector to produce a zero residual. We also describe the IRL method and show how implicit application of a polynomial filter to the initial vector is accomplished through iterations by a truncated version of the implicitly shifted QR-algorithm. Section 3 defines Leja shifts and discusses the rate of convergence of the IRL method based on Leja shifts. An IRL algorithm based on exact shifts and two IRL algorithms based on Leja shifts are described in Section 4, and Section 5 presents a few computed examples. Our experience with the IRL algorithms is summarized in Section 6.

2. The implicitly restarted Lanczos method.

When the implicitly restarted Arnoldi method, described in [30], is applied to a symmetric matrix, certain simplifications of the computational scheme are possible. This section describes the simplified scheme so obtained. We use the notation of [30] in order to make a comparison of the present paper with [30] easy.

The Lanczos factorization may be viewed as a truncated reduction of an $n \times n$ symmetric matrix A to tridiagonal form. After k steps of the factorization one has

$$AV = VH + fe_k^T, \tag{2.1}$$

where $V \in \mathbf{R}^{n\times k}$, $V^TV = I_k$, $H \in \mathbf{R}^{k\times k}$ is symmetric and tridiagonal and $f \in \mathbf{R}^n$ with $0 = V^Tf$. Throughout this paper e_k denotes the kth axis vector of appropriate dimension and I_k denotes the $k\times k$ identity matrix. Equation (2.1) can also be written as

$$AV = (V, v)\begin{pmatrix} H \\ \beta e_k^T \end{pmatrix}, \qquad \beta = \|f\|, \qquad v = \frac{1}{\beta}f, \tag{2.2}$$

and this representation shows that (2.1) is just a truncation of the complete reduction of the matrix A to tridiagonal form. Approximate eigenvalues and eigenvectors are readily available through this factorization. Let $\{\theta, y\}$ be an eigenvalue-eigenvector pair of the matrix H. Then the vector $x = Vy$ satisfies

$$\|Ax - x\theta\| = \|(AV - VH)y\| = |\beta e_k^T y|. \tag{2.3}$$

The vector x is referred to as a Ritz vector and θ as a Ritz value of A. The residual error (2.3) associated with the Ritz pair $\{\theta, x\}$ can be determined by evaluating $|\beta e_k^T y|$, without explicitly determining the Ritz vector x.

The factorization (2.2) may be advanced one step through the following recursion formulas:

(2.4.1) $\beta = \|f\|; \quad v = \frac{1}{\beta}f;$

(2.4.2) $V^+ = (V, v);$

(2.4.3) $w = Av; \quad \alpha = w^Tv;$

(2.4.4) $H^+ = \begin{pmatrix} H & e_k\beta \\ \beta e_k^T & \alpha \end{pmatrix};$

(2.4.5) $f^+ = w - V^+\begin{pmatrix} e_k\beta \\ \alpha \end{pmatrix} = (I - V^+(V^+)^T)w.$

It is easily seen that

$$AV^+ = V^+H^+ + f^+e_{k+1}^T, \quad (V^+)^TV^+ = I_{k+1}, \quad (V^+)^Tf^+ = 0.$$

Numerical difficulties can arise at Step (2.4.5). The vector f^+ should be numerically orthogonal to all of the columns of V^+, but this property is generally not obtained without further computation. Neglecting to enforce orthogonality virtually assures the appearance of spurious eigenvalues. Selective reorthogonalization against Ritz vectors corresponding to converged Ritz values has been proposed to address this problem; see [20, 22]. The most direct way to avoid the appearance of spurious eigenvalues is to enforce numerical orthogonality of the columns of V. There are three options: one can compute and store V in Householder form (see [14, 33]), reorthogonalize V using a QR-factorization, or use iterative refinement to orthogonalize f against V as it is computed; see [4, 19, 20, 26]. The last option is preferred because the computations required can be expressed in terms of Level 2 BLAS, see, e.g., [5], and because the columns of V can be updated with the orthogonal transformations required to perform the implicitly shifted QR-steps as outlined below. The computational cost of iterative refinement remains acceptable when the matrix V has a modest number of columns.

Restarting provides a means for restricting the number of columns of V. The goal of restarting is to replace the initial vector $v_1 = Ve_1$ by a vector that is as near as possible to being a linear combination of the eigenvectors associated with the k eigenvalues of interest. This condition is motivated by the fact that f vanishes if and only if v_1 is such a linear combination. We state this formally as a theorem.

THEOREM 2.1. *Let A be a symmetric matrix of order n and let $AV_k - V_kH_k = f_ke_k^T$ be a k-step Lanczos factorization of A, where H_k is an unreduced symmetric tridiagonal matrix (i.e., $f_j \neq 0$, $1 \leq j < k$). Then $f_k = 0$ if and only if $v_1 = U_ky$, where $AU_k = U_k\Lambda_k$ with $U_k^TU_k = I_k$ and Λ_k a diagonal matrix of order k.*

Various ways to make the initial vector v_1 be close to a linear combination of eigenvectors associated with desired eigenvalues have been proposed; see [2, 16]. All of them involve an update of the initial vector followed by an explicit restart of the Lanczos factorization. The technique proposed in [30], in the context of an implicitly restarted Arnoldi method, iteratively applies a polynomial filter to the initial vector. We now describe the modification of this scheme obtained when the matrix A is symmetric.

Throughout the following discussion, the number k of desired eigenvalues should be thought of as a fixed pre-specified integer of modest size. Let p be another positive integer, and consider the result of $k+p$ steps of the Lanczos process applied to the matrix A. This yields a $(k+p) \times (k+p)$ symmetric tridiagonal matrix H_{k+p} and a matrix $V_{k+p} \in \mathbf{R}^{n\times(k+p)}$ with orthonormal columns, such that

$$\begin{aligned} AV_{k+p} &= V_{k+p}H_{k+p} + r_{k+p}e_{k+p}^T \\ &= (V_{k+p}, v_{k+p+1}) \begin{pmatrix} H_{k+p} \\ \beta_{k+p}e_{k+p}^T \end{pmatrix} . \end{aligned} \tag{2.5}$$

An analogue of the explicitly shifted QR-algorithm may be applied to this truncated factorization of A. It consists of the following four steps. Let μ be a shift and determine the QR-factorization, $H_{k+p} - \mu I = QR$, where $Q, R \in \mathbf{R}^{(k+p)\times(k+p)}$, $Q^TQ = I_{k+p}$ and R is upper triangular. Putting $V = V_{k+p}$ and $H = H_{k+p}$, we obtain

$$(A - \mu I)V - V(H - \mu I) = f_{k+p}e_{k+p}^T, \tag{2.6.1}$$

$$(A - \mu I)V - VQR = f_{k+p}e_{k+p}^T, \tag{2.6.2}$$

$$(A - \mu I)(VQ) - (VQ)(RQ) = f_{k+p}e_{k+p}^TQ, \tag{2.6.3}$$

$$A(VQ) - (VQ)(RQ + \mu I) = f_{k+p}e_{k+p}^TQ. \tag{2.6.4}$$

Let $V_+ = VQ$ and $H_+ = RQ + \mu I$. Then H_+ is symmetric and tridiagonal. When applying the left-hand side and right-hand side of equation (2.6.2) to the vector e_1, we obtain

$$(A - \mu I)v_1 = v_1^+\rho_{11}, \tag{2.7}$$

where $\rho_{11} = e_1^T R e_1$ and $v_1^+ = V_+ e_1$. Equation (2.7) exposes the relationship between the initial vectors v_1 and v_1^+. An application of p shifts $\mu_1, \mu_2, \ldots, \mu_p$ results in

$$AV_{k+p}^+ = (V_{k+p}^+ \,, v_{k+p+1}) \begin{pmatrix} H_{k+p}^+ \\ \beta_{k+p} e_{k+p}^T \hat{Q} \end{pmatrix}, \tag{2.8}$$

where $V_{k+p}^+ = V_{k+p}\hat{Q}$, $H_{k+p}^+ = \hat{Q}^T H_{k+p} \hat{Q}$, and $\hat{Q} = Q_1 Q_2 \cdots Q_p$. Here Q_j denotes the orthogonal matrix associated with the shift μ_j. Partition the matrices

$$V_{k+p}^+ = (V_k^+, \hat{V}_p) \,, \quad H_{k+p}^+ = \begin{pmatrix} H_k^+ & \hat{\beta}_k e_k e_1^T \\ \hat{\beta}_k e_1 e_k^T & \hat{H}_p \end{pmatrix}, \tag{2.9}$$

and note that

$$\beta_{k+p} e_{k+p}^T \hat{Q} = (\underbrace{0, 0, \ldots, \tilde{\beta}_{k+p}}_{k}, \underbrace{b^{\;T}}_{p}) \,. \tag{2.10}$$

Substitution of (2.9)-(2.10) into (2.8) yields

$$A(V_k^+, \hat{V}_p) = (V_k^+, \hat{V}_p, v_{k+p+1}) \begin{pmatrix} H_k^+ & \hat{\beta}_k e_k e_1^T \\ \hat{\beta}_k e_1 e_k^T & \hat{H}_p \\ \tilde{\beta}_{k+p} e_k^T & b^T \end{pmatrix}. \tag{2.11}$$

Equating the first k columns on the right-hand side and left-hand side of (2.11) gives

$$AV_k^+ = V_k^+ H_k^+ + f_k^+ e_k^T,$$

which we write in the form

$$AV_k^+ = (V_k^+, v_{k+1}^+) \begin{pmatrix} H_k^+ \\ \beta_k^+ e_k^T \end{pmatrix}, \tag{2.12}$$

where $v_{k+1}^+ = \frac{1}{\beta_k^+} f_k^+$, $f_k^+ = (\hat{V}_p e_1 \hat{\beta}_k + v_{k+p+1} \tilde{\beta}_{k+p})$ and $\beta_k^+ = \|f_k^+\|$. Note that $(V_k^+)^T \hat{V}_p e_1 = 0$ and $(V_k^+)^T v_{k+p+1} = 0$ so $(V_k^+)^T v_{k+1}^+ = 0$. Thus, (2.12) is a Lanczos factorization of the matrix A. The initial vector of this factorization can be written as $v_1^+ = \gamma_p \psi_p(A) v_1$, where ψ_p is a monic polynomial of degree p having the shifts $\mu_1, \mu_2, \ldots, \mu_p$ as zeros, and γ_p is a scaling factor such that the vector v_1^+ is of unit length.

The matrices in (2.12) are of the same sizes as in (2.2). We now can apply the recursion formulas (2.4.1)-(2.4.5) of the Lanczos process p times to the matrices (2.12) in order to obtain a factorization with matrices of the same sizes as in (2.5) and with initial vector v_1 in (2.5) replaced by v_1^+. We remark that the application of p shifts and p steps of the Lanczos process only requires p matrix-vector multiplications with the matrix A. The scheme proceeds in this manner to alternatively apply p shifts and p steps of the Lanczos process until convergence has been achieved.

Our scheme is to be compared with the explicit restarting methods described in [2, 16], where the entire Lanczos sequence is restarted. From the standpoint of numerical stability our updating scheme has several advantages:

(1) Orthogonality can be maintained since the value of k is modest.
(2) There is no occurrence of spurious eigenvalues.
(3) The storage requirement is fixed; it does not grow with the number of iterations.
(4) Deflation techniques similar to those applied in the QR-algorithm for dealing with numerically small subdiagonal elements of the matrices H_{k+j}, $1 \le j \le p$, may be applied.

3. Shift selection. The selection of the shifts determines the convergence properties of the algorithm. This section discusses properties of exact shifts and of Leja shifts. The latter shifts are Leja points for an interval on the real axis containing unwanted eigenvalues. Exact shifts, discussed in [30], are obtained by computing the eigenvalues of H_{k+p}, and selecting p of them in the unwanted portion of the spectrum as shifts. The following lemma shows some properties of exact shifts. The lemma uses notation introduced in Section 2. We denote the spectrum of a matrix B by $\lambda(B)$.

LEMMA 3.1. ([30, Lemma 3.10]) *Let $\lambda(H_{k+p}) = \{\theta_1, \ldots, \theta_k\} \cup \{\mu_1, \ldots, \mu_p\}$ be a disjoint partition of the spectrum of H_{k+p}, and let*

$$H_{k+p}^{+} = \hat{Q}^T H_{k+p} \hat{Q},$$

where $\hat{Q} = Q_1 Q_2 \cdots Q_p$, and Q_j is implicitly determined by the shift μ_j. Let β_j denote the $(j+1)$st subdiagonal element of H_{k+p}^{+}. If $\beta_j \neq 0$ for $1 \leq j < k$, then $\beta_k = 0$ and the matrices Q_j can be chosen so that

$$H_{k+p}^{+} = \begin{pmatrix} H_k^{+} & 0 \\ 0 & D_p \end{pmatrix},$$

where $\lambda(H_k^{+}) = \{\theta_1, \ldots, \theta_k\}$ and D_p is a diagonal matrix with diagonal entries $\mu_1, \mu_2, \ldots, \mu_p$. Moreover,

$$v_1^{+} = V_{k+p} \hat{Q} e_1 = \sum_{j=1}^{k} \eta_j x_j,$$

where $\eta_j \in \mathbf{R}$ and the x_j are Ritz vectors of A associated with the Ritz values θ_j, i.e., $x_j = V_{k+p} y_j$ with $H_{k+p} y_j = y_j \theta_j$ for $1 \leq j \leq k$.

This lemma shows the effect of the polynomial filter when exact shifts are used. Eliminating the unwanted set of eigenvalues of H_{k+p} by using exact shifts is mathematically equivalent to restarting the Lanczos factorization with the initial vector

$$v_1^{+} = \sum_{j=1}^{k} \eta_j x_j,$$

where the right-hand side is a linear combination of Ritz vectors of A associated with the desired eigenvalues. Thus, the initial vector v_1 has been implicitly replaced by a sum of k approximate eigenvectors.

In many cases this shift selection strategy works well. However, slow convergence can be observed for certain matrices, e.g., when computing the smallest eigenvalues of a positive definite matrix that has well separated large eigenvalues. In this case the large eigenvalues reappear as shifts in almost every iteration, and components of eigenvectors in v_1 associated with other undesired eigenvalues do not get damped rapidly. This is illustrated in Example 5.2 of Section 5.

Leja shifts can overcome this difficulty. In order to define these shifts and discuss their properties we need to introduce some notation. In this paper we only require Leja shifts that lie in a real interval, however, it is suggestive to define Leja shifts that lie in more general compact sets in the complex plane; our development indicates that several extensions of the scheme of the present paper may be possible. These generalizations are commented on in the end of this section.

Let $\mathbf{C}$ denote the complex plane, and identify $\mathbf{C}$ with the real plane $\mathbf{R}^2$. Throughout this section $s, t \in \mathbf{R}$ and $z \in \mathbf{C}$. If $z = s+it$, $i = \sqrt{-1}$, then z and (s,t) denote the same point. Let $\mathbf{K} \subset \mathbf{C}$ be a compact set, whose complement $\mathbf{\Omega} = (\mathbf{C} \cup \{\infty\})\backslash\mathbf{K}$ is connected and possesses a Green function $G(s,t)$ with a logarithmic singularity at infinity. This Green function is uniquely determined by the requirements i) $\Delta G(s,t) = 0$ in $\mathbf{\Omega}\backslash\{\infty\}$, ii) $G(s,t) = 0$ on $\partial\mathbf{\Omega}$, where $\partial\mathbf{\Omega}$ denotes the boundary of $\mathbf{\Omega}$, and iii)

$$\frac{1}{2\pi}\int_{\partial\mathbf{\Omega}} \frac{\partial}{\partial n} G(s,t) d\sigma = 1,$$

where $\partial/\partial n$ denotes the normal derivative directed into $\mathbf{\Omega}$ and $d\sigma$ stands for the element of arc length; see, e.g., [34, Chapter 4.1] for details. The nonnegative constant c defined by

$$c = \lim_{|z|\to\infty} |z| \exp(-G(s,t)), \qquad z = s+it,$$

is called the *capacity* of $\mathbf{K}$. The capacity depends on the size of $\mathbf{K}$. If $\mathbf{K}$ has capacity c and α is a positive constant, then $\alpha\mathbf{K} = \{\alpha z : z \in \mathbf{K}\}$ has capacity αc.

Example 3.1. Let $\mathbf{K} = \{z : |z| \le r\}$ for some constant $r > 0$. Then the Green function is given by

$$G(s,t) = \ln|z/r|, \qquad z = s+it,$$

and therefore $\mathbf{K}$ has capacity r. $\Box$

For the scheme of the present paper, sets $\mathbf{K}$ that are intervals are of interest. Such sets are discussed in the following example.

Example 3.2. Let $\mathbf{K} = [-a, a]$, $a > 0$. We obtain the Green function

$$G(s,t) = \ln|(z/a) + ((z/a)^2 - 1)^{1/2}|, \qquad z = s+it, \tag{3.1}$$

where the branch of the square root is chosen so that $|(z/a) + ((z/a)^2 - 1)^{1/2}| > 1$ for $z \in \mathbf{\Omega}$. The capacity of $\mathbf{K}$ is therefore $a/2$. For future reference, we also note that if instead $\mathbf{K} = [a, b]$, then the associated Green function is given by

$$G(s,t) = \ln\left|\frac{2}{b-a}\left(z - \frac{b+a}{2} + (z^2 - z(b+a) + ba)^{1/2}\right)\right|, \qquad z = s+it, \tag{3.2}$$

and $\mathbf{K}$ has capacity $\frac{1}{4}(b-a)$. $\Box$

Let $w(z)$ be a continuous weight function on $\mathbf{K}$, such that

$$\alpha \le w(z) \le \beta, \qquad z \in \mathbf{K}, \tag{3.3}$$

for some constants $0 < \alpha \le \beta < \infty$, and introduce a recursively defined sequence $\{z_j\}_{j=0}^{\infty}$ of points in $\mathbf{K}$ as follows. Let z_0 be a point such that

$$w(z_0)|z_0| = \max_{z\in\mathbf{K}} w(z)|z|, \qquad z_0 \in \mathbf{K}, \tag{3.4}$$

and let z_j satisfy

$$w(z_j)\prod_{l=0}^{j-1} |z_j - z_l| = \max_{z\in\mathbf{K}} w(z)\prod_{l=0}^{j-1} |z - z_l|, \qquad z_j \in \mathbf{K}, \qquad j = 1, 2, \dots . \tag{3.5}$$

ETNA
Kent State University
etna@mcs.kent.edu

The points z_j determined by (3.4)-(3.5) might not be unique. We call any sequence of points $\{z_j\}_{j=0}^{\infty}$ that satisfies (3.4)-(3.5) a sequence of weighted Leja points for $\mathbf{K}$, or sometimes briefly Leja points for $\mathbf{K}$. Because we will use these points as shifts in the IRL method, we also will refer to them as Leja shifts.

When $w(z) = 1$, the weighted Leja points agree with the "classical" Leja points studied by Leja [17], and probably first introduced by Edrei [6]. Leja [17] showed that the classical Leja points for $\mathbf{K}$ are uniformly distributed with respect to the density function

$$\frac{1}{2\pi}\frac{\partial}{\partial n}G(s,t), \qquad z = s + it \in \partial\mathbf{\Omega}. \tag{3.6}$$

A simple modification of Leja's proof shows that the weighted Leja points also have this property.

Example 3.3. Let $\mathbf{K} = \{z : |z| \le r\}$ for some constant $r > 0$. Then

$$\frac{\partial}{\partial n}G(s,t) = \frac{1}{r}, \qquad s^2 + t^2 = r^2.$$

Thus, the weighted Leja points for $\mathbf{K}$ are uniformly distributed on $\partial\mathbf{\Omega}$. $\square$

Example 3.4. Let $\mathbf{K} = [a, b]$. Then the weighted Leja points are uniformly distributed on $\mathbf{K}$ with respect to the density function

$$\frac{1}{2\pi}\frac{\partial}{\partial n}G(s,0) = \frac{1}{\pi}(b-s)^{-1/2}(s-a)^{-1/2}, \qquad a < s < b. \tag{3.7}$$

$\square$

Lemma 3.2. ([17, 25]) *Let $\{z_j\}_{j=0}^{\infty}$ be a sequence of weighted Leja points for $\mathbf{K}$ and assume that the weight function $w(z)$ satisfies (3.3). Then*

$$\prod_{l=0}^{j-1} |z_j - z_l| \ge \frac{\alpha}{\beta}c^j, \tag{3.8}$$

$$\lim_{j\to\infty}\prod_{l=0}^{j-1} |z_j - z_l|^{1/j} = c, \tag{3.9}$$

$$\lim_{j\to\infty}\prod_{l=0}^{j-1} |z - z_l|^{1/j} = c\exp(G(s,t)), \qquad z = s + it \in \mathbf{\Omega}, \tag{3.10}$$

where c denotes the capacity of $\mathbf{K}$. The convergence in (3.10) is uniform for z belonging to any compact subset of $\mathbf{\Omega}$.

We remark that Lemma 3.2 holds for any sequence $\{z_j\}_{j=0}^{\infty}$ of nodes uniformly distributed on $\partial\mathbf{\Omega}$ with respect to the density function (3.6). Besides sets of Leja points, also sets Fekete and of Fejér points for $\mathbf{K}$ are uniformly distributed on Ω; see [12, 34] for definitions and properties of the latter point sets. The fact that sets of Fejér, Fekete and Leja points satisfy Lemma 3.2 has made it attractive to use them in the construction of semiiterative methods for the solution of large linear systems of equations; see [7, 8, 9, 10, 24, 25] and references therein.

Example 3.5. Let $\mathbf{K} = [a, b]$ and define the Chebyshev polynomials

$$T_p(z) = \cos(p \arccos\left(\frac{2z - a_j - b_j}{b_j - a_j}\right)), \qquad p = 1, 2, 3 \dots\ ,$$

for $\mathbf{K}$. Then the zeros $\{\zeta_j^{(p)}\}_{j=1}^{p}$ of $T_p(z)$ are Fejér points for $\mathbf{K}$ and are uniformly distributed on $\mathbf{K}$ with respect to (3.7) as p increases. $\square$

Leja points for $\mathbf{K}$ have certain algorithmic advantages compared with zeros of Chebyshev polynomials. We will discuss this further in Section 4. Moreover, Leja points can be applied for sets $\mathbf{K}$ more general than an interval. This makes it possible to generalize the schemes of the present paper as outlined below.

Introduce the spectral decomposition

$$A = U\Lambda U^T, \tag{3.11}$$

where

$$\begin{aligned} \Lambda &= \operatorname{diag}(\lambda_1, \lambda_2, \dots \lambda_n), \qquad && \lambda_1 \le \lambda_2 \le \dots \le \lambda_n, \\ U &= (u_1, u_2, \dots, u_n), && U^T U = I. \end{aligned} \tag{3.12}$$

For definiteness, assume that we wish to determine the k smallest eigenvalues of A. We then seek to determine a polynomial filter ψ_m of degree m that is small on the undesired part of the spectrum and large on the wanted part. In view of that we do not know the undesired eigenvalues, we determine intervals $\mathbf{K} = [a, b]$ that contain many of the unwanted eigenvalues adaptively during the iterations with the IRL method, and we require the polynomial filter to be small on these intervals. In order to discuss the rate of convergence of our IRL method we consider the quotients $|\psi_m(\lambda_j)|/\max_{z\in\mathbf{K}}|\psi_m(z)|$. The next theorem sheds light on the decrease of this quotient as the degree m increases.

THEOREM 3.3. *Let the spectral decomposition of A be given by (3.11)-(3.12) and define the polynomials*

$$\psi_m(z) = \prod_{j=0}^{m-1}(z - z_j), \tag{3.13}$$

where the z_j are Leja points for the real interval $\mathbf{K} = [a, b]$, $a \ne b$. Let l be a positive integer such that $\lambda_l < a$. Then

$$\lim_{m\to\infty}\left(\frac{|\psi_m(\lambda_j)|}{\max_{z\in\mathbf{K}}|\psi_m(z)|}\right)^{1/m} = \exp(G(\lambda_j, 0)), \qquad 1 \le j \le l, \tag{3.14}$$

where the Green function G is given by (3.2). The limit is the smallest possible in the sense that there is no sequence of monic polynomials $\{\psi_j\}_{j=0}^{\infty}$, where ψ_j is of degree j, such that a limit smaller than the right-hand side of (3.14) is achieved.

Proof. The equality follows from Lemma 3.2. The left-hand side is minimized by Chebyshev polynomials for the interval $\mathbf{K}$. Explicit formulas for Chebyshev polynomials yield the same limit. $\square$

If we assume that the interval $\mathbf{K}$ is symmetric with respect to the origin, i.e., $\mathbf{K} = [-a, a]$ for some constant $a > 0$, then the expression (3.1) can be substituted into the right hand side of (3.14). We obtain, for $\lambda_l < -a$, that

$$\lim_{m\to\infty}\left(\frac{|\psi_m(\lambda_j)|}{\max_{z\in\mathbf{K}}|\psi_m(z)|}\right)^{1/m} = |\lambda_j/a| + (|\lambda_j/a|^2 - 1)^{1/2}, \qquad 1 \le j \le l. \tag{3.15}$$

It is easy to see that, for fixed λ_j, the right hand side of (3.15) decreases as a increases. Theorem 3.3 can be applied to determine the distance between the eigenvectors associated with the desired eigenvalues of A and the subspaces generated by the IRL method. This is the object of the next theorem. Related results have been shown by Rutishauser for the subspace iteration method [27, Theorem 2].

THEOREM 3.4. *Let l be a positive integer, let $\hat{\lambda}_1 < \hat{\lambda}_2 < \ldots < \hat{\lambda}_{l+1}$ be the $l+1$ smallest distinct eigenvalues of the matrix A and let λ_n be the largest eigenvalue. Let $\hat{u}_j$ denote an eigenvector associated with $\hat{\lambda}_j$. Assume that the vector v_1 contains components of all the eigenvectors $\hat{u}_1, \hat{u}_2, \ldots, \hat{u}_l$. Let the polynomial filter ψ_m be defined by (3.13) and assume that the zeros z_j of ψ_m are Leja points for the real interval $\mathbf{K} = [a, b]$, such that $a \neq b$ and $b \geq \lambda_n$. Let $v_m^+ = \psi_m(A)v_1$ and define the subspaces*

$$\mathbf{E}_l(A, v_m^+) = \operatorname{span}\{v_m^+, Av_m^+, \ldots, A^{l-1}v_m^+\}, \qquad m = 0, 1, 2, \ldots \ . \tag{3.16}$$

Then the distance between $\mathbf{E}_l(A, v_m^+)$ and $\hat{u}_j$ satisfies, for $1 \leq j \leq l$,

$$\lim_{m\to\infty} (\operatorname{dist}\{\mathbf{E}_l(A, v_m^+), \hat{u}_j\})^{1/m} \leq \begin{cases} \exp(G(\hat{\lambda}_{l+1}, 0) - G(\hat{\lambda}_j, 0)), & \text{if } \hat{\lambda}_j \leq \hat{\lambda}_{l+1} < a, \\ \\ \exp(-G(\hat{\lambda}_j, 0)), & \text{if } \hat{\lambda}_j \leq a \leq \hat{\lambda}_{l+1}. \end{cases}$$

Proof. First assume that the l smallest eigenvalues of A are distinct. Then $\hat{\lambda}_j = \lambda_j$ and $\hat{u}_j = u_j$ for $1 \leq j \leq l$. Let $s = (s_1, s_2, \ldots, s_n)^T = U^T v_1$, where $A = U\Lambda U^T$ is the spectral decomposition (3.11). By assumption, $s_j \neq 0$ for all j. Define the matrix $S = \operatorname{diag}(s_1, s_2, \ldots, s_l)$. Introduce the non-singular transposed Vandermonde matrix

$$W_l = \begin{pmatrix} 1 & \lambda_1 & \cdots & \lambda_1^{l-1} \\ 1 & \lambda_2 & \cdots & \lambda_2^{l-1} \\ \vdots & \vdots & & \vdots \\ 1 & \lambda_l & \cdots & \lambda_l^{l-1} \end{pmatrix}.$$

Then

$$\begin{aligned} \mathbf{E}_l(A, v_1) &= U \operatorname{span}\{s, \Lambda s, \ldots, \Lambda^{l-1}s\} = U\,S \operatorname{span}\left\{ \begin{matrix} W_l \\ M \end{matrix} \right\} && (3.17) \\ &= U\,S \operatorname{span}\left\{ \begin{matrix} I_l \\ MW_l^{-1} \end{matrix} \right\}, && (3.18) \end{aligned}$$

where M is an $(n-l) \times l$ matrix, and span$\{C\}$ denotes the span of the columns of the matrix C. Thus, with $\hat{M} = MW_l^{-1}$,

$$\mathbf{E}_l(A, v_m^+) = \psi_m(A)\mathbf{E}_l(A, v_1) = U\,S\,\psi_m(\Lambda) \operatorname{span}\left\{ \begin{matrix} I_l \\ \hat{M} \end{matrix} \right\}. \tag{3.19}$$

Let $\|\cdot\|_2$ denote the Euclidean vector norm. Then

$$
\begin{aligned}
\operatorname{dist}\{\mathbf{E}_l(A, v_m^+), u_j\} &= \min_{y\in\mathbf{R}^l} \left\| US\psi_m(\Lambda)\begin{pmatrix} I_l \\ \hat{M}\end{pmatrix} y - u_j \right\|_2 \\
&= \min_{y\in\mathbf{R}^l} \left\| S\psi_m(\Lambda)\begin{pmatrix} I_l \\ \hat{M}\end{pmatrix} y - e_j \right\|_2 \\
&\leq \left\| S\psi_m(\Lambda)\begin{pmatrix} e_j \\ \hat{M}e_j\end{pmatrix} (s_j\psi_m(\lambda_j))^{-1} - e_j \right\|_2 \\
&\leq d \max_{l+1\leq k\leq n}\{|\psi_m(\lambda_k)|\}\, |\psi_m(\lambda_j)|^{-1},
\end{aligned} \tag{3.20}
$$

where d is a constant independent of m. Note that the bound also holds for other norms with the constant d depending on the choice of norm. The theorem now follows from (3.14) or (3.9).

We turn to the proof when the $l+1$ smallest eigenvalues of A are not distinct, and consider the case when $l=2$ and $\lambda_1 = \lambda_2 < \lambda_3 < \lambda_4 \leq \ldots \leq \lambda_n$, i.e., $\hat{\lambda}_1 = \lambda_1$, $\hat{\lambda}_2 = \lambda_3$ and $\hat{\lambda}_3 = \lambda_4$. The proof for other values of l and other multiplicities of the eigenvalues is analogous. Similarly to (3.17), we obtain

$$
\mathbf{E}_2(A, v_1) = US \operatorname{span}\left\{ \begin{matrix} W_{3,2} \\ M' \end{matrix} \right\},
$$

where

$$
W_{3,2} = \begin{pmatrix} 1 & \lambda_1 \\ 1 & \lambda_2 \\ 1 & \lambda_3 \end{pmatrix},
$$

and M' is an $(n-3)\times 2$ matrix. Define

$$
I_{3,2} = \begin{pmatrix} 1 & 0 \\ 1 & 0 \\ 0 & 1 \end{pmatrix}.
$$

Analogously to (3.18), we obtain for a suitable matrix M'', that

$$
\begin{aligned}
\mathbf{E}_2(A, v_1) &= US \operatorname{span}\left\{ \begin{matrix} I_{3,2} \\ M'' \end{matrix} \right\} \\
&= (u_1 s_1 + u_2 s_2, u_3 s_3, \ldots, u_n s_n) \operatorname{span}\left\{ \begin{matrix} I_2 \\ M'' \end{matrix} \right\}.
\end{aligned} \tag{3.21}
$$

Introduce $\Lambda' = \operatorname{diag}(\lambda_2, \lambda_3, \ldots, \lambda_n)$. Then (3.21) yields

$$
\begin{aligned}
\mathbf{E}_2(A, v_m^+) &= \psi_m(A)\mathbf{E}_2(A, v_1) = US\psi_m(\Lambda) \operatorname{span}\left\{ \begin{matrix} I_{3,2} \\ M'' \end{matrix} \right\} \\
&= (u_1 s_1 + u_2 s_2, u_3 s_3, \ldots, u_n s_n)\psi_m(\Lambda') \operatorname{span}\left\{ \begin{matrix} I_2 \\ M'' \end{matrix} \right\}.
\end{aligned} \tag{3.22}
$$

Substitute $\hat{u}_1 = u_1 s_1 + u_2 s_2$ and $\hat{u}_2 = u_3$ into the right-hand side of (3.22). The the right-hand side obtained is of the same form as (3.19), and a bound similar to (3.20) can be shown. □

The proof of Theorem 3.4 shows that for a suitable initial vector v_1, the IRL method determines one vector in each subspace $\mathrm{Ker}(A - \hat{\lambda}_j I)$ for $j = 1, 2, \ldots, l$. The proof also shows that the IRL method determines at most one vector in each of these subspaces.

Now assume that we want to determine the k smallest distinct eigenvalues of A. The next result shows how to select the interval $\mathbf{K} = [a, b]$ in Theorem 3.4 in order to make the distance between the space $\mathbf{E}_l(A, v_m^+)$ and the associated eigenvectors $\hat{u}_1, \hat{u}_2, \ldots, \hat{u}_k$ small.

THEOREM 3.5. *Let the assumptions of Theorem 3.4 be valid, and let k be a positive integer not larger than l. Then the minimum, with respect to the endpoints of the interval $\mathbf{K} = [a, b]$, of the right hand side of*

$$\lim_{m\to\infty} (\mathrm{dist}\{\mathbf{E}_l(A, v_m^+), \hat{u}_k\})^{1/m} \le \begin{cases} \exp(G(\hat{\lambda}_{l+1}, 0) - G(\hat{\lambda}_k, 0)), & \text{if } \hat{\lambda}_k \le \hat{\lambda}_{l+1} < a, \\ \\ \exp(-G(\hat{\lambda}_k, 0)), & \text{if } \hat{\lambda}_k \le a \le \hat{\lambda}_{l+1}. \end{cases}$$

is achieved for $a = \hat{\lambda}_{l+1}$ and $b = \lambda_n$, where we minimize over all $a \ge \hat{\lambda}_k$ and $b \ge \lambda_n$.

Proof. In this proof we denote the Green function for the interval $\mathbf{K} = [a, b]$ by $G(s, t; a, b)$. We may assume that the interval $\mathbf{K}$ is symmetric with respect to the origin; otherwise we shift the interval by $-\frac{1}{2}(a + b)$ and the matrix A by $-\frac{1}{2}(a + b)I$. Thus, $\mathbf{K} = [-\alpha, \alpha]$ with $-\alpha = a$. Consider the function

$$h(\alpha) = G(\hat{\lambda}_{l+1}, 0; -\alpha, \alpha) - G(\hat{\lambda}_k, 0; -\alpha, \alpha), \qquad \hat{\lambda}_k \le \hat{\lambda}_{l+1} \le -\alpha.$$

We want to show that $h(\alpha)$ is minimal when $\alpha = -\hat{\lambda}_{l+1}$. Let $r = |\hat{\lambda}_{l+1}/\alpha|$ and $s = |\hat{\lambda}_k/\alpha|$. Then $s \ge r \ge 1$ and by (3.1) we have

$$h(\alpha) = \ln(r + \sqrt{r^2 - 1}) - \ln(s + \sqrt{s^2 - 1}).$$

It follows that

$$h'(\alpha) = -\frac{1}{\alpha}\left(\frac{r}{\sqrt{r^2 - 1}} - \frac{s}{\sqrt{s^2 - 1}}\right) \le 0.$$

Therefore, the minimal value of $h(\alpha)$ is achieved at $\alpha = -\hat{\lambda}_{l+1}$ and

$$h(-\hat{\lambda}_{l+1}) = -G(\hat{\lambda}_k, 0; \hat{\lambda}_{l+1}, -\hat{\lambda}_{l+1}).$$

Define the function

$$g(\beta) = -G(\hat{\lambda}_k, 0; \hat{\lambda}_{l+1}, \beta), \qquad \lambda_n \le \beta.$$

It follows from (3.2) that $g(\beta)$ decreases as β decreases. A proof of the analogous result for Green functions for more general sets $\mathbf{\Omega}$ is presented by Eiermann et al. [7, Proposition 3]. Related results can also be found in [8, 9]. Thus, $g(\beta)$ is minimal for $\beta = \lambda_n$. This proves the theorem for $\hat{\lambda}_k \le \hat{\lambda}_{l+1} \le a$. We turn to the case when $\hat{\lambda}_k \le a \le \hat{\lambda}_{l+1}$ and consider the function

$$f(a) = -G(\hat{\lambda}_k, 0; a, b), \qquad \hat{\lambda}_k \le a \le \hat{\lambda}_{l+1}.$$

By (3.2) or [7, Proposition 3] it follows that f increases as a decreases. This completes the proof of the theorem. $\square$

We now use the fact that the Lanczos method is a Rayleigh-Ritz procedure in order to obtain error bounds for the computed eigenvalues.

THEOREM 3.6. *Let* $\{v_j\}_{j=1}^{s}$ *be an orthonormal basis of* $\mathbf{E}_s(A, v_m^+)$ *for* $1 \le s \le l$ *and define* $V_s = [v_1, v_2, \ldots, v_s]$. *Let* m_0 *be an integer sufficiently large. Then, for* $m \ge m_0$, *the eigenvalues* $\theta_1 \le \theta_2 \le \ldots \le \theta_l$ *of* $H = V_l^T A V_l$ *satisfy*

$$\lambda_j \le \theta_j \le \lambda_j + d_\delta \exp(-2mG(\lambda_j, 0) + \delta), \qquad 1 \le j \le l, \tag{3.23}$$

for any constant $\delta > 0$ *and a constant* d_δ *that depends on* δ, *but is independent of* $m \ge m_0$ *and* j.

Proof. Let $\mathbf{G}_j$ denote a subspace of dimension j. Then

$$\theta_j = \min_{\mathbf{G}_j \subset \mathbf{E}_l(A, v_m^+)} \max_{g \in \mathbf{G}_j} \frac{g^T A g}{g^T g}$$

and

$$\begin{aligned}
\lambda_j &= \min_{\mathbf{G}_j \subset \mathbf{R}^n} \max_{g \in \mathbf{G}_j} \frac{g^T A g}{g^T g} \le \min_{\mathbf{G}_j \subset \mathbf{E}_l(A, v_m^+)} \max_{g \in \mathbf{G}_j} \frac{g^T A g}{g^T g} = \theta_j \\
&\le \max_{g \in \mathbf{E}_j(A, v_m^+)} \frac{g^T A g}{g^T g} = \max_{\hat{g} \in U^T \mathbf{E}_j(A, v_m^+)} \frac{\hat{g}^T \Lambda \hat{g}}{\hat{g}^T \hat{g}}.
\end{aligned} \tag{3.24}$$

From the proof of Theorem 3.4 we obtain that

$$\hat{g} = S\psi_m(\Lambda) \begin{pmatrix} I_l \\ \hat{M} \end{pmatrix} (s_j \psi_m(\lambda_j))^{-1}$$

yields the upper bound in (3.23), and all elements of $U^T \mathbf{E}_j(A, v_m^+)$ satisfy a bound of this form. □

To obtain an IRL algorithm based on Leja shifts, we have to specify how to choose the endpoints a and b of the interval $\mathbf{K}$. The next section describes two strategies for determining the endpoints in an adaptive manner during the computations.

To conclude this section, we briefly indicate some possible extensions of the results of the present section. First let $\mathbf{K}$ be the union of two real intervals, i.e., $\mathbf{K} = [a, c] \cup [d, b]$, and let the endpoints satisfy

$$a \le \lambda_1 \le \ldots \le \lambda_j \le c < \lambda_{j+1} \le \ldots \le \lambda_{j+k} < d \le \lambda_{j+k+1} \le \ldots \le \lambda_n \le b.$$

Then Lemma 3.2 remains valid and results analogous to Theorems 3.3-3.6 can be shown. It therefore would appear possible to generalize the computational schemes based on Leja shifts of the present paper to the computation of k nonextreme eigenvalues $\lambda_{j+1} \le \lambda_{j+2} \le \ldots \le \lambda_{j+k}$ of a symmetric matrix A.

Now let instead A be a large nonsymmetric matrix, and let $\mathbf{K}$ be a compact set in $\mathbf{C}$ that contains the undesired eigenvalues of A, but not the desired ones. Analogues of Theorems 3.3-3.6 can be shown and indicate that the computational scheme of the present paper can be modified to be applicable to the determination of a few selected eigenvalues of a nonsymmetric matrix. These generalizations of the schemes of the present paper are being investigated.

4. Algorithms based on the IRL method.

We describe three IRL algorithms for computing the k smallest distinct eigenvalues $\{\hat{\lambda}_j\}_{j=1}^{k}$ and associated eigenvectors $\{\hat{u}_j\}_{j=1}^{k}$ of a large symmetric matrix A. The algorithms differ in the selection of shifts. Their performances are compared in Section 5. Throughout this section v_1 denotes the first vector of the initial Krylov subspace used. The dimension of the Krylov subspaces generated is varied between k and $k+p$. We let $\{\theta_j, y_j\}_{j=1}^{k+p}$ be eigenvalue-eigenvector pairs of the symmetric tridiagonal matrix $H_{k+p} \in \mathbf{R}^{(k+p)\times(k+p)}$ defined by (2.5), and assume that the eigenvalues are ordered according to

$$\theta_1 < \theta_2 < \ldots < \theta_{k+p}. \tag{4.1}$$

We remark that we may assume that the off-diagonal elements β_j of H_{k+p} are non-vanishing, because otherwise we have found an invariant subspace, and therefore the eigenvalues θ_j are distinct. Let x_j be a Ritz vector of the matrix A, associated with the Ritz value θ_j. Then, analogously with (2.3), we obtain that

$$\|Ax_j - x_j\theta_j\| = |\beta_{k+p}e_{k+p}^T y_j|, \qquad 1 \le j \le k+p,$$

where β_{k+p} is defined by (2.5). In our IRL algorithms the computations are terminated as soon as

$$\max_{1\le j\le k+p} |\beta_{k+p}e_{k+p}^T y_j| \le \varepsilon|\theta_j|, \tag{4.2}$$

where ε is a given positive constant. Our first algorithm uses exact shifts.

Algorithm 4.1. IRL-ES*:*

```
Input: $A$, $k$, $p$, $v_1$, $\varepsilon$;
Output: $\{\hat{\lambda}_j\}_{j=1}^{k}$, $\{\hat{u}_j\}_{j=1}^{k}$;
  1. Determine the Lanczos factorization (2.5);
  2. Compute the eigenvalues (4.1) of the matrix $H_{k+p} \in \mathbf{R}^{(k+p)\times(k+p)}$;
  3. If inequality (4.2) is satisfied then stop;
  4. Apply shifts $\mu = \theta_j$, $j = k+1, k+2, \ldots, k+p$, according to (2.6)-(2.12);
  5. Advance Lanczos factorization $p$ steps in order to obtain (2.5). Go to 2; □
```

The following two algorithms use Leja shifts for sets $\mathbf{K}$ and implement different strategies for choosing $\mathbf{K}$ adaptively during the computations. These algorithms differ from Algorithm IRL-ES only in Step 3. The first of these schemes, Algorithm IRL-LSNI, defined below, uses Leja Shifts for a sequence of Nested Intervals $\mathbf{K}_j = [a_j, b_j]$ that are determined during the iterations as follows. The first time we determine eigenvalues (4.1) of a $(k+p)\times(k+p)$ symmetric tridiagonal matrix H_{k+p} in the algorithm, we define the initial interval $\mathbf{K}_0 = [a_0, b_0]$ with endpoints

$$a_0 = \theta_{k+1}, \qquad b_0 = \theta_{k+p}. \tag{4.3}$$

In the course of the iterations new $(k+p)\times(k+p)$ symmetric tridiagonal matrices H_{k+p} are generated and their eigenvalues are computed. For each new set of eigenvalues (4.1) computed, we update $\mathbf{K}_j$, i.e., we replace $\mathbf{K}_j$ by $\mathbf{K}_{j+1} = [a_{j+1}, b_{j+1}]$, where

$$a_{j+1} = \min\{a_j, \theta_{k+1}\}, \qquad b_{j+1} = \max\{b_j, \theta_{k+p}\}, \qquad j = 0, 1, 2, \ldots\ . \tag{4.4}$$

Our motivation for this choice of intervals $\mathbf{K}_j$ is furnished by the Cauchy interlacing theorem (see, e.g., [20]), which states that the eigenvalues (4.1) of any one of the matrices H_{k+p} generated satisfy

$$\lambda_k \le \theta_k, \qquad \theta_{k+p} \le \lambda_n.$$

Therefore, the sequence of intervals $\mathbf{K}_j = [a_j, b_j]$, $j \geq 0$, defined by (4.3)-(4.4) satisfies

$$\mathbf{K}_j \subset \mathbf{K}_{j+1} \subset [\lambda_{k+1}, \lambda_n]. \tag{4.5}$$

Filter polynomials determined by Leja points for these intervals damp components of eigenvectors in v_1 associated with eigenvalues in the undesired part of the spectrum.

When we determine Leja points for $\mathbf{K}_j$, we distribute them taking previously allocated points into account by keeping the latter in the product that we maximize; see Algorithm 4.2 below. This feature is discussed further after Algorithm 4.2. We use the weight function

$$w(z) = |z - \theta_{k+1}| \tag{4.6}$$

when generating Leja points. This choice of weight function satisfies (3.3). Numerical experiments showed this weight function to yield faster convergence during the first iterations of our IRL algorithms based on Leja points than the trivial weight function $w(z) = 1$. A discussion on a related weight function used when generating Leja points for application in a Richardson iteration method for the solution of large linear systems of equations can be found in [25].

ALGORITHM 4.2. Compute p Leja points for $\mathbf{K}_j$ given points $\{z_k\}_{k=0}^{r-1}$:

```
Input: $a_j$, $b_j$, $\theta_{k+1}$, $r$, $\{z_k\}_{k=0}^{r-1}$;
Output: $\{z_k\}_{k=r}^{p+r-1}$;
 1. k=r;
 2. if k=0 then
        $z_0 = b_j$
    else
        Determine $z_k \in \mathbf{K}_j$, such that

        $w(z_k)\prod_{l=0}^{k-1} |z_k - z_l| = \max_{z\in\mathbf{K}_j} w(z)\prod_{i=0}^{k-1} |z - z_i|$,

        where w(z) is defined by (4.6);
    endif;
 3. $k = k + 1$;
 4. if $k < p + r$ then go to 2 else stop; □
```

When p new Leja points $\{z_k\}_{k=r}^{p+r-1}$ are computed by Algorithm 4.2, their distribution is affected by previously determined points $\{z_k\}_{k=0}^{r-1}$. For instance, if the points $\{z_k\}_{k=0}^{r-1}$ are Leja points for an interval $\mathbf{K}' = [a', b']$ and Algorithm 4.2 is applied to determine p new Leja points $\{z_k\}_{k=r}^{p+r-1}$ for a larger interval $\mathbf{K} = [a, b]$, with $a < a'$ and $b > b'$, then the new points will be distributed so that the density function for all the $r+p$ points $\{z_k\}_{k=0}^{p+r-1}$ approximates the density function (3.7) well. In particular, many of the p new points might be allocated in the set $\mathbf{K}\backslash\mathbf{K}' = [a, a'] \cup [b', b]$. It is easy to see that if we would ignore previously allocated points $\{z_k\}_{k=0}^{r-1}$ when determining new points $\{z_k\}_{k=r}^{p+r-1}$, then the distribution function for the $r + p$ points $\{z_k\}_{k=0}^{p+r-1}$ might be very different from (3.7), unless $p \gg r$. This, in turn, could reduce the rate of convergence of the IRL algorithms based on Leja points considerably. Keeping previously determined Leja points when determining new ones provides a "memory" of which eigenvector components of the initial vector v_1 already have been damped.

The determination of each Leja points, except z_0, by Algorithm 4.2 requires the maximization of a product over an interval $[a_j, b_j]$. In order to reduce the computational effort necessary to determine Leja points, we discretize the interval $[a_j, b_j]$

using grid points $\{t_k\}_{k=1}^{\ell}$, for some l sufficiently large. In the computed examples of Section 5, we choose the grid points to be zeros of a Chebyshev polynomial of the first kind of degree ℓ for the interval $[a_j, b_j]$. We are in a position to describe the first of the algorithms based on Leja points.

Algorithm 4.3. IRL-LSNI*:*

```
Input: $A$, $k$, $p$, $v_1$, $\varepsilon$;
Output: $\{\hat{\lambda}_j\}_{j=1}^{k}$, $\{\hat{u}_j\}_{j=1}^{k}$;
 1.  Determine the Lanczos factorization (2.5); j=0;
 2.  Compute the eigenvalues (4.1) of the matrix $H_{k+p} \in \mathbf{R}^{(k+p)\times(k+p)}$;
 3.  If inequality (4.2) is satisfied then stop;
 4a. If $j=0$ then define the interval $\mathbf{K}_j = [a_j, b_j]$ by (4.3) else by (4.4);
 4b. Compute p Leja points $\{z_k\}_{k=jp}^{(j+1)p-1}$ for $\mathbf{K}_j$ by Algorithm 4.2 with r=jp and the
     weight function (4.6) defined by the $(k+1)$st smallest eigenvalue of $H_{k+p}$;
 4c. Apply shifts $\mu = z_k$, $k = jp, jp+1, \ldots, (j+1)p-1$, according to (2.6)-(2.12);
 5.  Advance Lanczos factorization p steps in order to obtain (2.5); $j = j+1$;
     Go to 2; □
```

We remark that the convergence analysis developed in Section 3 applies to this algorithm if for some integer $l \geq 0$, $\mathbf{K}_j = \mathbf{K}_l$ for all $j \geq l$. We can then set $\mathbf{K} = \mathbf{K}_l$ in Theorems 3.4-3.6.

Assume for the moment that A has distinct eigenvalues and that

$$\lambda_{k+1} \ll \lambda_{k+p+1}. \tag{4.7}$$

Algorithm IRL-LSNI then determines a sequence of intervals $\mathbf{K}_j$ that satisfy (4.5) and we assume for simplicity that $\mathbf{K}_j = [\lambda_{k+1}, \lambda_n]$ for $j \geq 0$. Theorem 3.4 with $l = k+p$ shows that it would suffice to allocate Leja points on the interval $[\lambda_{k+p+1}, \lambda_n]$. Because of (4.7), Theorem 3.5 leads us to expect that allocating Leja points on $[\lambda_{k+p+1}, \lambda_n]$ would yield a higher rate of convergence than allocating Leja points on $[\lambda_{k+1}, \lambda_n]$. However, it is difficult to determine approximations of the interval $[\lambda_{k+p+1}, \lambda_n]$ from the matrices H_{k+p} computed by the Lanczos processes. Algorithm IRL-LSFLE, defined below, presents an approach to obtain intervals $\mathbf{K}_j$ that are smaller than those determined by Algorithm IRL-LSNI. In Algorithm IRL-LSFLE we update the endpoints of the intervals $\mathbf{K}_j = [a_j, b_j]$ according to

$$a_{j+1} = \theta_{k+1}, \qquad b_{j+1} = \max\{b_j, \theta_{k+p}\}, \qquad j = 0, 1, 2, \ldots . \tag{4.8}$$

Formula (4.8) lets the left endpoint "free" in the sense that the a_j are not required to be monotonically decreasing. This inspired the acronym LSFLE for Leja Shifts with Free Left Endpoint. The sets $\mathbf{K}_j$ obtained in this manner are typically smaller than the sets determined by Algorithm IRL-LSNI, and computed examples, some of which are presented in Section 5, show that Algorithm IRL-LSFLE generally requires fewer iterations than Algorithm IRL-LSNI.

Algorithm 4.4. IRL-LSFLE*:*

```
Input: $A$, $k$, $p$, $v_1$, $\varepsilon$;
Output: $\{\hat{\lambda}_j\}_{j=1}^{k}$, $\{\hat{u}_j\}_{j=1}^{k}$;
1.  Determine the Lanczos factorization (2.5); j=0;
2.  Compute the eigenvalues (4.1) of the matrix $H_{k+p} \in \mathbf{R}^{(k+p)\times(k+p)}$;
3.  If inequality (4.2) is satisfied then stop;
4a. If $j = 0$ then define the interval $\mathbf{K}_j = [a_j, b_j]$ by (4.3) else by (4.8);
4b. Compute p Leja points $\{z_k\}_{k=jp}^{(j+1)p-1}$ for $\mathbf{K}_j$ by Algorithm 4.2 with r=jp and the
    weight function (4.6) defined by the $(k+1)$st smallest eigenvalue of $H_{k+p}$;
4c. Apply shifts $\mu = z_k$, $k = jp, jp+1, \ldots, (j+1)p-1$, according to (2.6)-(2.12);
5.  Advance Lanczos factorization p steps in order to obtain (2.5); $j = j+1$;
    Go to 2; □
```

The IRL algorithms of this section are designed to determine the k smallest eigenvalues of the matrix A. It is straightforward to modify these algorithms if instead the k largest eigenvalues of A are desired.

Instead of using Leja shifts in Algorithms IRL-LSNI and IRL-LSFLE, one could for each j use the zeros of the Chebyshev polynomial $T_p(z)$ for the interval $[a_j, b_j]$ as shifts. However, these "Chebyshev shifts" do not yield convergence nearly as rapidly as Leja shifts. This depends on that the Chebyshev shifts for the interval $[a_j, b_j]$ do not take into account the location of the shifts previously applied. Thus, Chebyshev shifts lack the "memory" of Leja shifts.

Richardson iteration is a popular iterative method for the solution of large linear systems of equations; see [1, 8, 9, 10, 24, 25]. The IRL method can be viewed as a modification of Richardson iteration applicable to the determination of desired eigenvalues and eigenvectors. In the context of Richardson iteration, the shifts are often referred to as relaxation parameters. Computational experience with the Richardson iteration method is indicative of the performance of the IRL method. In particular, shift selection strategies that yield slow convergence in the context of Richardson iteration typically also yield poor convergence of the IRL method. Illustrative computed examples that demonstrate the superiority of Leja shifts, compared with Chebyshev shifts, in the context of adaptive Richardson iteration have been presented in [1]. We will therefore not consider Chebyshev shifts further.

5. Computed examples.

This section describes numerical experiments that compare the performance of the IRL algorithms presented in Section 4 for determining a few of the smallest eigenvalues and associated eigenvectors. The computations were carried out on an IBM RISC 6000/550 workstation, using single precision arithmetic, i.e., with approximately eight significant digits. In all examples the input parameter ε in the IRL algorithms was chosen to be $1 \cdot 10^{-8}$ and the initial vector v_1 was obtained by first determining a vector $\tilde{v}$ with uniformly distributed random entries in the interval $(-1, 1)$, and then letting $v_1 = \tilde{v}/\|\tilde{v}\|_2$.

Example 5.1. The entries of the symmetric 100×100 matrix A considered in this example are random numbers uniformly distributed in the interval $(-1, 1)$. We want to compute the four smallest eigenvalues and associated eigenvectors using a Krylov subspace of dimension at most eight. The input parameters are given by $k = 4$ and $p = 4$. Algorithm IRL-ES[1] required 185 matrix-vector products, Algorithm IRL-LSNI

[1] In all numerical examples we used the implementation in ARPACK of Algorithm IRL-ES. This implementation contains various improvements compared with Algorithm 4.1 in Section 4 in order to enhance convergence. ARPACK can be requested by sending e-mail to sorensen@rice.edu

ETNA
Kent State University
etna@mcs.kent.edu

required 144 and Algorithm IRL-LSFLE required 124. □

In all examples Algorithm IRL-LSFLE determined the desired eigenpairs using fewer matrix-vector multiplications than Algorithm IRL-LSNI. We therefore do not report the performance of Algorithm IRL-LSNI in the remaining examples.

Example 5.2. Let A be a 100×100 matrix of the form $U^T DU$, where U is a randomly generated orthogonal matrix and D is a diagonal matrix with evenly spaced entries in the interval $[\mathbf{u}, 1]$. Here $\mathbf{u}$ denotes the unit roundoff. We want to compute the two smallest eigenvalues and associated eigenvectors using a Krylov subspace of dimension at most four, i.e., we chose the input parameters $k = 2$ and $p = 4$. Algorithm IRL-ES failed to find any of the desired eigenvalues and eigenvectors within 80 iterations, which require about 480 matrix-vector multiplications. This may be attributed to the fact that approximations of the two largest eigenvalues of A kept reappearing as eigenvalues of the matrices H_{k+p} generated. Algorithm IRL-LSFLE, on the other hand, computed the desired pair of eigenvalues and eigenvectors using 92 matrix-vector multiplications. □

Example 5.3. The matrix of this example comes from a technique described in [23] for solving a three-dimensional quantum mechanical scattering problem within the context of the the Adiabatically adjusting Principal axis Hyperspherical (APH) coordinate approach. In this approach the total wave function is expanded in a basis of sector adiabatic surface functions

$$\Psi^{JMpn} = 4 \sum_{t,\Lambda} \rho^{-5/2} \psi^{Jpn}_{t,\Lambda}(\rho) \Phi^{Jp}_{t,\Lambda}(\theta, \chi; \rho_\zeta) \hat{D}^{Jp}_{\Lambda M}(\alpha_Q, \beta_Q, \gamma_Q).$$

Here the surface functions $\Phi_t = \Phi^{Jp}_{t,\Lambda}$ are bound state eigenfunctions of the two-dimensional surface Hamiltonian, i.e.,

$$\mathcal{H}(\theta, \chi; \rho_\zeta) \oplus_\sqcup (\theta, \chi; \rho_\zeta) = \mathcal{E}_\sqcup(\rho_\zeta) \oplus_\sqcup (\theta, \chi; \rho_\zeta), \tag{5.1}$$

where θ and χ are the two hyperspherical angles and ρ is the scattering coordinate. We wish to determine a few of the smallest eigenvalues and eigenvectors of the Hamiltonian operator (5.1). The Hamiltonian operator is discretized by a Galerkin method that uses products of normalized Legendre polynomials in $\cos 2\theta$ and trigonometric polynomials in χ as test and trial functions. The discretized Hamiltonian operator $\mathcal{H}^{DVR}$ obtained in this manner, where DVR stands for Discrete Variable Representation, can be written in tensor product form. Using the notation of [23], one obtains

$$\mathcal{H}^{DVR} = h^{DVR}_\theta \otimes I_\chi + f^{DVR}_\theta \otimes h^{DVR}_\chi + \tilde{V}^{DVR},$$

where h^{DVR}_θ and h^{DVR}_χ are the kinetic energy operators for the θ and χ degrees of freedom. The matrix $\mathcal{H}^{DVR}$ so obtained has a regular and sparse structure. It can be written as

$$\mathcal{H}^{DVR} = \mathcal{B} + \mathcal{K},$$

where $\mathcal{B}$ is a block-diagonal matrix and the the matrix $\mathcal{K}$ has diagonal blocks. More precisely, we have

$$\mathcal{K} = \begin{pmatrix} \mathcal{K}_{11} & \mathcal{K}_{12} & \mathcal{K}_{13} & \dots & \mathcal{K}_{1n_\theta} \\ \mathcal{K}_{21} & \mathcal{K}_{22} & \mathcal{K}_{23} & \dots & \mathcal{K}_{2n_\theta} \\ \mathcal{K}_{31} & \mathcal{K}_{32} & \mathcal{K}_{33} & \dots & \mathcal{K}_{3n_\theta} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \mathcal{K}_{n_\theta 1} & \mathcal{K}_{n_\theta 2} & \mathcal{K}_{n_\theta 3} & \dots & \mathcal{K}_{n_\theta n_\theta} \end{pmatrix},$$

		ES	ES	LSFLE	LSFLE
k	$k+p$	# iterations	# mat-vec mult	# iterations	# mat-vec mult
6	12	127	491	65	396
6	10	431	975	94	382
6	8	> 1500		193	392
4	8	177	496	91	368
4	6	> 1500		150	304

Table 5.1

where each submatrix $\mathcal{K}_{jl}$ is a multiple of the identity matrix. For this eigenvalue problem one is typically interested in about 3% of the smallest eigenvalues. We refer to [23] for details.

In the present example we determined the 4-6 smallest eigenvalues and associated eigenvectors of the matrix $\mathcal{H}^{DVR}$ of order 200 by Algorithms IRL-ES and IRL-LSFLE. Table 1 shows that as the dimension of the Krylov subspace decreases, the number of matrix-vector products required by Algorithm IRL-ES to compute the desired eigenvalues and eigenvectors increased sharply, with the number of matrix-vector multiplications nearly doubling as the dimension of the Krylov subspace was reduced from 12 to 10. In fact, when a Krylov subspace of dimension 8 was used, Algorithm IRL-ES failed to determine the desired eigenvalues and eigenvectors within 1500 iterations. The number of matrix-vector products required by Algorithm IRL-LSFLE to compute the six smallest eigenvalues and eigenvectors, on the other hand, remained essentially unaffected by the change in dimension of the Krylov subspace. Similar behavior could be observed when we wanted to compute the four smallest eigenvalues and associated eigenvector of the matrix $\mathcal{H}^{DVR}$. $\square$

6. Conclusion.

This paper describes the IRL methods, analyzes Leja shifts and compares two IRL algorithms based on Leja shifts with an IRL algorithm that uses exact shifts. The numerical examples shown, as well as many other computed examples, indicate that if p is sufficiently large, then all three algorithms yield about the same rate of convergence. A reduction in p for fixed k typically increases the number of iterations necessary for convergence for all three algorithm, and in many examples the number of iterations required by Algorithm IRL-ES grows faster than the number of iterations requires by the algorithms based on Leja shifts. Among the latter, Algorithm IRL-LSFLE performed best. Thus, when p is large any of the three algorithms described can be chosen, but when p is small, e.g., because of limitations in fast computer storage available, Algorithm IRL-LSFLE is preferred. When a value of p should be considered large depends on the distribution of eigenvalues of A and on the size of k.

Acknowledgement. Work on this paper was carried out while D.C. and L.R. visited Stanford University. They would like to thank Gene Golub for making this visit possible.

REFERENCES

[1] D. Calvetti and L. Reichel, *Adaptive Richardson iteration based on Leja points*, Math. Comp., to appear.

[2] J. Cullum, *The simultaneous computation of a few of the algebraically largest and smallest eigenvalues of a large, symmetric sparse matrix*, BIT, 18 (1978), pp. 265–275.

[3] J. Cullum and R.A. Willoughby, *Lanczos Algorithms for Large Symmetric Eigenvalue Computations*, Vol. 1, Birkhäuser, Boston, MA, 1985.

[4] J. Daniel, W.B. Gragg, L. Kaufman and G.W. Stewart, *Reorthogonalization and stable algorithms for updating the Gram-Schmidt QR factorization*, Math. Comp., 30 (1976), pp. 772–795.

[5] J.J. Dongarra, I.S. Duff, D.C. Sorensen and H.A. van der Vorst, *Solving Linear Systems on Vector and Shared Memory Computers*, SIAM, Philadelphia, PA, 1991.

[6] A. Edrei, *Sur les déterminants récurrents et les singularités d'une fonction donnée par son développement de Taylor*, Composito Math., 7 (1939), pp. 20–88.

[7] M. Eiermann, X. Li and R.S. Varga, *On hybrid semi-iterative methods*, SIAM J. Numer. Anal., 26 (1989), pp. 152–168.

[8] M. Eiermann and W. Niethammer, *On the construction of semiiterative methods*, SIAM J. Numer. Anal., 20 (1983), pp. 1153–1160.

[9] M. Eiermann, W. Niethammer and R.S. Varga, *A study of semi-iterative methods for nonsymmetric systems of linear equations*, Numer. Math., 47 (1985), pp. 505–533.

[10] B. Fischer and L. Reichel, *A stable Richardson iteration method for complex linear systems*, Numer. Math., 54 (1988), pp. 225–242.

[11] D.A. Flanders and G. Shortley, *Numerical determination of fundamental modes*, J. Appl. Phys., 21 (1950), pp. 1326–1332.

[12] D. Gaier, *Lectures on Complex Approximation*, Birkhäuser, Basel, 1987.

[13] R.G. Grimes, J.L. Lewis and H.D. Simon, *A shifted block Lanczos algorithm for solving sparse symmetric generalized eigenproblems*, SIAM J. Matrix Anal., 15 (1994), pp. 228–272.

[14] G.H. Golub, R. Underwood and J.H. Wilkinson, *The Lanczos algorithm for the symmetric $Ax = \lambda Bx$ problem*, Report STAN-CS-72-270, Department of Computer Science, Stanford University, Stanford, CA, 1972.

[15] D. Ho, F. Chatelin and M. Bennani, *Arnoldi-Tchebyshev procedure for large scale nonsymmetric matrices*, Math. Model. Numer. Anal., 24 (1990), pp. 53–65.

[16] W. Karush, *An iterative method for finding characteristic vectors of a symmetric matrix*, Pacific J. Math., 1 (1951), pp. 233–248.

[17] F. Leja, *Sur certaines suits liées aux ensemble plan et leur application à la representation conforme*, Ann. Polon. Math., 4 (1957), pp. 8–13.

[18] R. Natarajan and D. Vanderbilt, *A new iterative scheme for obtaining eigenvectors of large, real-symmetric matrices*, J. Comp. Phys., 82 (1989), pp. 218–228.

[19] C.C. Paige, *Computational variants of the Lanczos method for the eigenproblem*, J. Inst. Math. Appl., 10 (1972), pp. 373–381.

[20] B.N. Parlett, *The Symmetric Eigenvalue Problem*, Prentice-Hall, Englewood Cliffs, NJ, 1980.

[21] B.N. Parlett and B. Nour-Omid, *Towards a black box Lanczos program*, Comput. Phys. Comm., 53 (1989), pp. 169–179.

[22] B.N. Parlett and D.S. Scott, *The Lanczos algorithm with selective orthogonalization*, Math. Comp., 33 (1979), pp. 311–328.

[23] P. Pendergast, Z. Darakjian, E.F. Hayes and D.C. Sorensen, *Scalable algorithms for three-dimensional reactive scattering: evaluation of a new algorithm for obtaining surface functions*, J. Comp. Phys., to appear.

[24] L. Reichel, *Polynomials by conformal mapping for the Richardson iteration method for complex linear systems*, SIAM J. Numer. Anal., 25 (1988), pp. 1359–1368.

[25] L. Reichel, *The application of Leja points to Richardson iteration and polynomial preconditioning*, Linear Algebra Appl., 154-156 (1991), pp. 389–414.

[26] L. Reichel and W.B. Gragg, *Algorithm 686: FORTRAN subroutines for updating the QR decomposition of a matrix*, ACM Trans. Math. Software, 16 (1990), pp. 369–377.

[27] H. Rutishauser, *Computational aspects of F.L. Bauer's simultaneous iteration method*, Numer. Math., 13 (1969), pp. 4–13.

[28] Y. Saad, *Chebyshev acceleration techniques for solving nonsymmetric eigenvalue problems*, Math. Comp., 42 (1984), pp. 567–588.

[29] Y. Saad, *Numerical Methods for Large Eigenvalue Problems*, Halstead Press, New York, 1992.

[30] D.C. Sorensen, *Implicit application of polynomial filters in a k-step Arnoldi method*, SIAM J. Matrix Anal. Appl., 13 (1992), pp. 357–385.

[31] D.B. Szyld, *Criteria for combining inverse and Rayleigh quotient iteration*, SIAM J. Numer. Anal., 25 (1988), pp. 1369–1375.
[32] X. Yang, T.K. Sarkar and E. Arvas, *A survey of conjugate gradient algorithms for solution of extreme eigen-problems of a symmetric matrix*, IEEE Trans. Acoust. Speech Signal Proc., 37 (1989), pp. 1550–1555.
[33] H.F. Walker, *Implementation of the GMRES method using Householder transformations*, SIAM J. Sci. Statist. Comput., 9 (1988), pp. 152–163.
[34] J.L. Walsh, *Interpolation and Approximation by Rational Functions in the Complex Domain*, 4th ed., Amer. Math. Soc., Providence, RI, 1965.