ELSEVIER

Journal of Computational and Applied Mathematics 92 (1998) 109–133

JOURNAL OF COMPUTATIONAL AND APPLIED MATHEMATICS

A hybrid iterative method for symmetric indefinite linear systems

D. Calvetti [a,1], L. Reichel [b,*,2]

[a] *Department of Mathematics, Case Western Reserve University, Cleveland, OH 44106, USA*

[b] *Department of Mathematics and Computer Science, Kent State University, Kent, OH 44242, USA*

Received 20 November 1997; received in revised form 27 February 1998

Abstract

Hybrid iterative methods that combine a conjugate direction method with a simpler iteration scheme, such as Chebyshev or Richardson iteration, were first proposed in the 1950s. The ease with which Chebyshev and Richardson iteration can be implemented efficiently on a large variety of computer architectures has in recent years lead to renewed interest in iterative methods that use Chebyshev or Richardson iteration. This paper presents a new hybrid iterative method for the solution of linear systems of equations with a symmetric indefinite matrix. Our method combines the conjugate residual method with Richardson iteration. Special attention is paid to the determination of two real intervals, one on each side of the origin, that contain most of the eigenvalues of the matrix. These intervals are used to compute suitable iteration parameters for Richardson iteration. We also discuss when to switch between the methods. The hybrid scheme typically uses the Richardson method for most iterations, and this reduces the number of arithmetic vector operations significantly compared with the number of arithmetic vector operations required when only the conjugate residual method is used. Computed examples illustrate the competitiveness of the hybrid scheme.

Keywords: Conjugate residual method; Richardson iteration; Modified moments

1. Introduction

Many problems in science and engineering give rise to large linear systems of equations

$$Ax = b, \quad A \in \mathbb{R}^{n\times n}, \quad x, b \in \mathbb{R}^n, \tag{1.1}$$

with a sparse, nonsingular, symmetric, indefinite matrix. Several iterative methods have been designed specifically for the solution of such systems. Among the most well-known are the algorithms

* Corresponding author. E-mail: na.reichel@na-net.ornl.gov.

[1] Research supported in part by NSF grant DMS-9404692. E-mail: na.calvetti@na-net.ornl.gov.

[2] Research supported in part by NSF grant DMS-9404706.

0377-0427/98/$19.00
PII S0377-0427(98)00048-X

SYMMLQ and MINRES [20]. These methods are stable implementations of the Conjugate Gradient (CG) and Conjugate Residual (CR) methods, respectively. Recently, it was pointed out in [1] that the Orthodir implementation of the CR method also can be applied to the solution of symmetric indefinite linear systems of equations. This implementation computes the search directions from a three-term recurrence relation. Several semi-iterative methods are also available; see, e.g., [8, 14, 24]. A nice survey of iterative methods for symmetric indefinite systems is provided in [11].

Recently, semi-iterative methods, such as Chebyshev and Richardson iteration, have received renewed attention, because these methods are fairly easy to implement efficiently on modern vector and parallel computers; see, e.g., [9, 25] for implementation issues and timings. However, semi-iterative methods require that a set $\mathbb{I} := [a,b] \cup [c,d]$, $-\infty < a < b < 0 < c < d < \infty$, that contains all, or at least most, of the eigenvalues of the matrix A be explicitly known in order to determine suitable iteration parameters. For most matrices such a set is not explicitly known and this motivates the development of hybrid iteration schemes that comprise two iterative methods: a simple semi-iterative method and a more complicated iterative method, such as a conjugate direction method. The purpose of the latter method is to compute an improved approximate solution, as well as to determine orthogonal sections (defined below) of low order of the matrix A. Spectral information of A, gained by computing eigenvalues of the orthogonal sections, is used to determine a set $\mathbb{I} = [a,b] \cup [c,d]$ that contains most of the eigenvalues of A. This set is applied to compute iteration parameters for the semi-iterative method.

Hybrid methods for the solution of linear systems of equations with a symmetric positive-definite matrix were first suggested in the 1950s in [23], and have subsequently been discussed in [6,18], and more recently in [4]. These hybrid schemes use the CG method to "learn" about the spectrum of A.

The CG method can break down when applied to the iterative solution of symmetric indefinite linear systems. We propose a hybrid scheme for such systems that is based on the Orthodir implementation of the CR method and Richardson iteration. Typically, many more iterations are carried out by the Richardson method than by the CR method. This can yield a substantial reduction in the computational work necessary to solve (1.1), compared with the work required to solve (1.1) by the CR method only.

The numerical examples reported in this paper demonstrate the viability of hybrid iteration schemes for the solution of linear systems of equations with a symmetric indefinite matrix. Variants of the hybrid method proposed in the present paper can be easily developed; for instance it is straightforward to replace the Orthodir implementation of the CR method by the MINRES or SYMMLQ algorithms. We would expect these variants to perform similarly; our reason for using the Orthodir implementation of the CR method is that this provides an opportunity to derive some new relationships between tridiagonal matrices, zeros of residual polynomials and Gauss–Radau quadrature rules. These results complement previous investigations in [16, 19].

Hybrid schemes also have been proposed for the iterative solution of nonsymmetric linear systems; see [2, 10, 17]. We expect that the techniques used in the present paper will be helpful in further developing hybrid schemes for nonsymmetric linear systems.

An alternative to hybrid methods are so-called adaptive methods. The latter are based on one simple semi-iterative method only, such as Chebyshev or Richardson iteration, and seek to determine estimates of the spectrum by carrying out certain auxiliary computations, such as the evaluation of modified moments. An adaptive method for the iterative solution of symmetric indefinite linear systems is presented in [5]. A disadvantage of that method, compared with the hybrid scheme of the present paper, is that it requires the selection of an initial set $\mathbb{I}$ before the iterations can begin, and the total

number of iterations required depends on this selection. On the other hand, the computations required to update the set $\mathbb{I}$ are simpler in the adaptive method [5] than in the hybrid scheme of the present paper.

The following notation is used throughout this paper. For a real symmetric or complex symmetric $n \times n$ matrix C, we define the bilinear form

$$\langle \boldsymbol{u}, \boldsymbol{v} \rangle_C := \boldsymbol{u}^T C \boldsymbol{v} \tag{1.2}$$

and the functional

$$\|\boldsymbol{u}\|_C := |\langle \boldsymbol{u}, \boldsymbol{u} \rangle_C|^{1/2}. \tag{1.3}$$

The vectors $\boldsymbol{u}$ and $\boldsymbol{v}$ are either in $\mathbb{R}^n$ or $\mathbb{C}^n$. Note that the elements of complex vectors are not conjugated in (1.2). Moreover, $\|\cdot\|_C$ is not a norm; the triangle inequality does not hold. For instance, let $C := \operatorname{diag}[1, -1]$, $\boldsymbol{u} := [1, 1]^T$ and $\boldsymbol{v} := [1, -1]^T$. Then $\|\boldsymbol{u} + \boldsymbol{v}\|_C = 2$ and $\|\boldsymbol{u}\|_C = \|\boldsymbol{v}\|_C = 0$. Nevertheless, it will be convenient to use (1.3) to measure the "length" of a vector. We use the notation $\langle \cdot, \cdot \rangle = \langle \cdot, \cdot \rangle_I$ and $\|\cdot\| = \|\cdot\|_I$, where as usual I denotes the identity matrix.

We will use the spectral factorization

$$A = Q \Lambda Q^T, \tag{1.4}$$

where

$$Q^T Q = I, \qquad \Lambda = \operatorname{diag}[\lambda_1, \lambda_2, \ldots, \lambda_n], \quad \lambda_1 \leqslant \cdots \leqslant \lambda_\ell < 0 < \lambda_{\ell+1} \leqslant \cdots \leqslant \lambda_n. \tag{1.5}$$

Let $\boldsymbol{x}_0$ be an initial approximate solution of (1.1) and let $\boldsymbol{r}_0 := \boldsymbol{b} - A\boldsymbol{x}_0$ denote the associated residual vector. Apply m iterations by the CR method to the solution of the linear system

$$A\boldsymbol{y} = \boldsymbol{r}_0, \tag{1.6}$$

with initial approximate solution $\boldsymbol{y}_0 = \boldsymbol{0}$. This yields the approximate solution $\boldsymbol{y}_m$ of (1.6), and the approximate solution $\boldsymbol{x}_m := \boldsymbol{x}_0 + \boldsymbol{y}_m$ of (1.1). The residual vector associated with $\boldsymbol{x}_m$ can be expressed as

$$\boldsymbol{r}_m := \boldsymbol{b} - A\boldsymbol{x}_m = \boldsymbol{r}_0 - A\boldsymbol{y}_m = p_m(A)\boldsymbol{r}_0, \tag{1.7}$$

where the polynomial p_m is referred to as a residual polynomial. It is of the form

$$p_m(z) = \prod_{k=0}^{m-1} (1 - \delta_k z). \tag{1.8}$$

It is well known that in exact arithmetic the residual polynomial p_m determined by the CR method satisfies

$$\|p_m(A)\boldsymbol{r}_0\| = \min_{p \in \mathbb{P}_m^{(0)}} \|p(A)\boldsymbol{r}_0\|, \tag{1.9}$$

where $\mathbb{P}_m^{(0)}$ denotes the set of all polynomials p of degree at most m, normalized so that $p(0) = 1$; see, e.g., [1, 12]. Thus, among all iterative methods that yield a residual polynomial of the form (1.8), the CR methods yields a residual error of minimal norm.

Let the $n \times m$ matrix U_m, $n \geqslant m$, satisfy $U_m^T U_m = I$. Then $U_m^T A U_m$ is said to be an orthogonal section of A. Its eigenvalues can be used to estimate extreme eigenvalues of A. The connection

between the CG method and the Lanczos process is well known and makes it possible to determine an orthogonal section of A from the recursion coefficients generated by the CG method; see, e.g., [4, 6, 12].

The recursion coefficients generated by the CR method also can be used to determine orthogonal sections of the matrix A. The connection between the CR method and the Lanczos process with respect to the bilinear form $\langle \boldsymbol{u}, \boldsymbol{v} \rangle_A$ is discussed in [19]. Section 3 shows how two orthogonal sections of A can be computed from the recursion coefficients determined by the CR algorithm. These sections are real symmetric or complex symmetric tridiagonal matrices. Their eigenvalues are used to determine estimates of the eigenvalues $\lambda_1, \lambda_\ell, \lambda_{\ell+1}$ and λ_n of A. For reason of numerical stability these estimates are not computed from the eigenvalues of the orthogonal sections directly, but from Gaussian quadrature rules associated with these sections. The estimates are used to determine the endpoints of sets $\mathbb{I} = [a,b] \cup [c,d]$, with $a < b < 0 < c < d$. Typically, the sets $\mathbb{I}$ so obtained contain most of the eigenvalues of A, and they are used to determine iteration parameters for Richardson iteration.

After having computed $\boldsymbol{r}_m$ and $\boldsymbol{x}_m$ by the CR method and determined a set $\mathbb{I} = [a,b] \cup [c,d]$, our hybrid scheme seeks to compute more accurate approximate solutions $\boldsymbol{x}_k$, $k > m$, by Richardson iteration

$$\begin{aligned} \boldsymbol{x}_{k+1} &:= \boldsymbol{x}_k + \delta_k \boldsymbol{r}_k, \\ & \qquad k := m, m+1, \ldots . \\ \boldsymbol{r}_k &:= \boldsymbol{b} - A\boldsymbol{x}_k, \end{aligned} \tag{1.10}$$

The iteration parameters δ_k are chosen as reciprocal values of Leja points, defined in Section 2, for the set $\mathbb{I}$. Our implementation uses a leapfrog variant of Richardson iteration, which is mathematically equivalent to, but computationally more efficient than, (1.10). Richardson iteration is continued until either a sufficiently accurate approximate solution has been found, or until slow convergence signals that the set $\mathbb{I}$ used is too small. In the latter case, we switch back to the CR method and improve the most recently computed approximate solution by m CR iterations. From the iteration parameters determined by the CR method, we compute new orthogonal sections of A. Gaussian quadrature rules associated with these sections are used to enlarge the set $\mathbb{I}$. Richardson iteration is then resumed with iteration parameters determined as reciprocal values of Leja points associated with the new larger set $\mathbb{I}$. The hybrid scheme switches between CR and Richardson iteration in the manner indicated until a sufficiently accurate approximate solution of (1.1) has been computed.

This paper is organized as follows. Section 2 reviews Richardson iteration based on Leja points for a set $\mathbb{I}$ consisting of two real intervals. The determination of orthogonal sections of A from recursion coefficients of the CR method is considered in Section 3, which also discusses the application of Gaussian quadrature rules to determine estimates of extreme eigenvalues of A. An algorithm for our hybrid scheme is presented in Section 4, and computed examples are described in Section 5. Concluding remarks can be found in Section 6.

2. Richardson iteration

This section first defines Leja points for sets of the form

$$\mathbb{I} := [a,b] \cup [c,d], \quad -\infty < a < b < 0 < c < d < \infty \tag{2.1}$$

and then reviews their application to Richardson iteration. Introduce the weight function

$$\omega(z) = |z| \tag{2.2}$$

and let

$$z_k := 1/\delta_k, \quad 0 \leqslant k < m, \tag{2.3}$$

where the δ_k are given by (1.8). Thus, the z_k, $0 \leqslant k < m$, are the zeros of the residual polynomial (1.8) determined by the CR method. For $k \geqslant m$, let z_k satisfy

$$\omega(z_k) \prod_{j=0}^{k-1} |z_k - z_j| = \max_{z \in \mathbb{I}} \omega(z) \prod_{j=0}^{k-1} |z - z_j|, \quad z_k \in \mathbb{I}. \tag{2.4}$$

In general, formula (2.4) does not determine the points z_k uniquely. We call any sequence of points $\{z_k\}_{k=0}^{\infty}$ that satisfies (2.3)–(2.4) a sequence of Leja points for $\mathbb{I}$, and we refer to the points in such a sequence as Leja points for $\mathbb{I}$. These point sequences for the weight function $\omega(z) = 1$ are studied in [15]. The asymptotic properties of the Leja points is the same for this weight functions and for the one given by (2.2). Computed examples in [21] show the latter weight function to be preferable in the context of Richardson iteration. We will therefore use the function (2.2) in the present paper.

Consider the residual errors

$$\boldsymbol{r}_k := \boldsymbol{b} - A\boldsymbol{x}_k, \quad k = m, m+1, m+2, \ldots,$$

where $\boldsymbol{x}_m$ is determined by the CR method from $\boldsymbol{x}_0$, and the $\boldsymbol{x}_k$, $k > m$, are computed by Richardson iteration (1.10). It follows from (1.7), (1.8) and (1.10) that

$$\boldsymbol{r}_k = p_k(A)\boldsymbol{r}_0, \quad k = m, m+1, m+2, \ldots, \tag{2.5}$$

where

$$p_k(z) = p_m(z) \prod_{j=m}^{k-1} (1 - \delta_j z), \quad k \geqslant m. \tag{2.6}$$

Here p_m is the residual polynomial (1.8) determined by the CR method. Let $\lambda(A)$ denote the spectrum of A, and assume for the moment that

$$\lambda(A) \subset \mathbb{I}. \tag{2.7}$$

Then it follows from (2.5) and (2.7) that

$$\overline{\lim_{k \to \infty}} \sup_{\boldsymbol{r}_0 \neq \boldsymbol{0}} \left(\frac{\|\boldsymbol{r}_k\|}{\|\boldsymbol{r}_0\|} \right)^{1/k} = \overline{\lim_{k \to \infty}} \max_{z \in \lambda(A)} |p_k(z)|^{1/k} \leqslant \overline{\lim_{k \to \infty}} \max_{z \in \mathbb{I}} |p_k(z)|^{1/k}. \tag{2.8}$$

The quantity on the right-hand side of (2.8) is referred to as the asymptotic convergence factor with respect to $\mathbb{I}$ of the iterative method defined by the iteration parameters $\{\delta_j\}_{j=0}^{\infty}$. Let the δ_j, $j \geqslant m$, be reciprocal values of Leja points for $\mathbb{I}$. Then it can be shown that

$$\overline{\lim_{k \to \infty}} \max_{z \in \mathbb{I}} |p_k(z)|^{1/k} = \overline{\lim_{k \to \infty}} \min_{p \in \mathbb{P}_k^{(0)}} \max_{z \in \mathbb{I}} |p(z)|^{1/k}, \tag{2.9}$$

i.e., the asymptotic convergence factor with respect to $\mathbb{I}$ is minimal when the iteration parameters are chosen as reciprocal values of Leja points for $\mathbb{I}$; see, e.g., [5, 21] for discussions. The quantity on the right-hand side of (2.9) is strictly smaller than unity and decreases when $\mathbb{I}$ is replaced by a subset. The smallest set $\mathbb{I}$ of the form (2.1) that satisfies (2.7) is given by

$$\mathbb{I}_0 = [\lambda_1, \lambda_\ell] \cup [\lambda_{\ell+1}, \lambda_n]. \tag{2.10}$$

Our hybrid scheme determines sets $\mathbb{I}$ that typically contain most but not all of the eigenvalues of A. Thus, generally, $\mathbb{I} \subsetneq \mathbb{I}_0$ and relation (2.7) is violated. Nevertheless, we use reciprocal values of Leja points for these sets $\mathbb{I}$ as iteration parameters for the Richardson method. The fact that (2.7) does not hold can result in slow or no convergence. We now describe a criterion for deciding when to enlarge a given set $\mathbb{I}$. It follows from (2.5) that

$$\|\boldsymbol{r}_k\| = \|p_k(A)\boldsymbol{r}_0\| \leqslant \max_{z\in\mathbb{I}_0} |p_k(z)| \|\boldsymbol{r}_0\|.$$

Now assume that

$$\boldsymbol{r}_k > \max_{z\in\mathbb{I}} |p_k(z)| \|\boldsymbol{r}_0\|. \tag{2.11}$$

Then clearly $\mathbb{I}_0 \not\subset \mathbb{I}$ and the rate of convergence of Richardson iteration may be improved by using a larger set $\mathbb{I}$. We therefore switch to CR iteration when (2.11) holds in order to determine a new larger set. A convergence proof of the hybrid method can be based on (1.9), (2.9) and this switching criterion. We omit the details.

3. CR iteration

The CR method is used to compute approximate solutions of systems

$$A\boldsymbol{y} = \boldsymbol{r}_k, \tag{3.1}$$

for certain values of k, where $\boldsymbol{r}_k := \boldsymbol{b} - A\boldsymbol{x}_k$, and to determine orthogonal sections of A. Algorithm 3.1 below carries out m iterations by the CR method with initial approximate solution $\boldsymbol{y}_0 = \boldsymbol{0}$. Let $\boldsymbol{y}_j$, $1 \leqslant j \leqslant m$, denote the approximate solutions of (3.1) determined by Algorithm 3.1, and define the associated residual vectors

$$\boldsymbol{s}_j := \boldsymbol{r}_k - A\boldsymbol{y}_j, \quad 0 \leqslant j \leqslant m. \tag{3.2}$$

Then $\boldsymbol{r}_{k+j} := \boldsymbol{s}_j$ is a residual vector of (1.1) associated with the approximate solution

$$\boldsymbol{x}_{k+j} := \boldsymbol{x}_k + \boldsymbol{y}_j. \tag{3.3}$$

Algorithm 3.1 (Orthodir algorithm for the CR(m) method for the solution of $A\boldsymbol{y} = \boldsymbol{r}_k$)
Input: m, q, $A, \boldsymbol{r}_k$, $\|\boldsymbol{r}_0\|$, ε;
Output if m iterations are completed: *Approximate solution* $\boldsymbol{y}_m$, *residual vector* $\boldsymbol{s}_m$, $\{\alpha_j\}_{j=0}^{m-1}$, $\{\gamma_j\}_{j=0}^{m-1}$, $\{\sigma_j\}_{j=1}^{m-1}$;
$\boldsymbol{y}_0 := \boldsymbol{d}_{-1} := \boldsymbol{d}'_{-1} := \boldsymbol{0}$; $\boldsymbol{d}_0 := \boldsymbol{s}_0 := \boldsymbol{r}_k$; $\boldsymbol{d}'_0 := A\boldsymbol{d}_0$;
for $j := 0, 1, \ldots, m-1$ **do**

if $\|s_j\|/\|r_0\| \leqslant \varepsilon$ **then** *found_solution* **endif**;
$\alpha_j := \eta_j^{-1}\langle s_j, d_j'\rangle$;
$y_{j+1} := y_j + \alpha_j d_j$; $s_{j+1} := s_j - \alpha_j d_j'$;
if $j = 0$ **then** $\sigma_j := 0$ **else** $\sigma_j := \eta_j/\eta_{j-1}$ **endif**;
$d_j'' := Ad_j'$; $\gamma_j := \eta_j^{-1}\langle d_j', d_j''\rangle$;
$d_{j+1} := d_j' - \gamma_j d_j - \sigma_j d_{j-1}$; $d_{j+1}' := d_j'' - \gamma_j d_j' - \sigma_j d_{j-1}'$;
endfor j;

The computations with Algorithm 3.1 are terminated before completion of m iterations if a computed residual vector s_j has been found to be of sufficiently small norm. We remark that the output specified in Algorithm 3.1 is only produced if m iterations are carried out. Otherwise, an approximate solution has been found. In the remainder of this paper, we will for notational convenience tacitly assume that every application of Algorithm 3.1 yields all the coefficients and vectors in the output statement of the algorithm. In order to secure that the $m \times m$ orthogonal section $\hat{B}_m$, defined below, exists, we require further that the residual vectors generated by the algorithm are linearly independent. If this requirement is violated, then we only can determine an orthogonal section $\hat{B}_k$ of some order $k < m$. Each iteration with Algorithm 3.1 requires 10 arithmetic operations with n-vectors. Here we count an inner product evaluation or a saxpy as one arithmetic operation with n-vectors. Also, the evaluation of the Euclidean norm is counted as one arithmetic vector operation, because it requires the computation of an inner product.

Throughout this paper $\tilde{e}_j$ denotes the jth column of the identity matrix of appropriate order, and $\mathcal{K}_m(A, c)$ denotes the Krylov subspace $\operatorname{span}\{c, Ac, \ldots, A^{m-1}c\}$, where A is the matrix in (1.1) and $c \in \mathbb{C}^n$.

Theorem 3.2. *Let the coefficients* $\{\gamma_j\}_{j=0}^{m-1}$ *and* $\{\sigma_j\}_{j=1}^{m-1}$ *be determined by Algorithm* 3.1. *Then the tridiagonal matrix*

$$\hat{T}_m := \begin{bmatrix} \gamma_0 & \sigma_1^{1/2} & & & 0 \\ \sigma_1^{1/2} & \gamma_1 & \sigma_2^{1/2} & & \\ & \sigma_2^{1/2} & \gamma_2 & & \\ & & & \ddots & \\ & & & \ddots & \sigma_{m-1}^{1/2} \\ 0 & & & \sigma_{m-1}^{1/2} & \gamma_{m-1} \end{bmatrix} \tag{3.4}$$

is an orthogonal section of the matrix A.

Proof. Introduce the $n \times m$ matrix

$$D_m := [d_0, d_1, \ldots, d_{m-1}]. \tag{3.5}$$

The three-term recurrence relation for the direction vectors d_j in Algorithm 3.1 can be written

$$D_m T_m = AD_m - d_m \tilde{e}_{km}^{\mathrm{T}}, \tag{3.6}$$

where

$$T_m := \begin{bmatrix} \gamma_0 & \sigma_1 & & & 0 \\ 1 & \gamma_1 & \sigma_2 & & \\ & 1 & \gamma_2 & & \\ & & & \ddots & \\ & & \ddots & \ddots & \sigma_{m-1} \\ 0 & & & 1 & \gamma_{m-1} \end{bmatrix}.$$

A direction vectors $\boldsymbol{d}_k$ in Algorithm 3.1 vanishes if and only if the solution $\boldsymbol{y}^*$ to $A\boldsymbol{y}=\boldsymbol{r}_k$ has been found. We may therefore assume that $\boldsymbol{d}_j \neq \boldsymbol{0}$ for $0 \leqslant j < m$. It is well known that $\langle \boldsymbol{d}_j, \boldsymbol{d}_k \rangle_{A^2} = 0$ for $j \neq k$. Thus, the matrix

$$N_m := D_m^{\mathrm{T}} A^2 D_m \tag{3.7}$$

is diagonal. Since A^2 is positive definite, N_m is positive definite. Its entries are computed by Algorithm 3.1; we have

$$N_m = \operatorname{diag}[\eta_0, \eta_1, \ldots, \eta_{m-1}].$$

It follows from (3.7) that the columns of the matrix $V_m := AD_m N_m^{-1/2}$ form an orthonormal basis of the Krylov subspace $\mathcal{K}_m(A, A\boldsymbol{d}_0)$ with respect to the inner product $\langle \cdot, \cdot \rangle$ and associated norm $\|\cdot\|$. In view of that $D_m^{\mathrm{T}} A^2 \boldsymbol{d}_m = \boldsymbol{0}$, we obtain from (3.6) that

$$N_m T_m = D_m^{\mathrm{T}} A^3 D_m.$$

Therefore

$$\hat{T}_m := N_m^{1/2} T_m N_m^{-1/2} = V_m^{\mathrm{T}} A V_m \tag{3.8}$$

is an orthogonal section of A. By Algorithm 3.1, $\sigma_j = \eta_j / \eta_{j-1}$, and it follows that the entries of the matrix (3.8) can be written in the form (3.4). □

Corollary 3.3. *Let* $\{\hat{\lambda}_j\}_{j=1}^m$ *denote the eigenvalues of the matrix* $\hat{T}_m$ *and assume that they are ordered according to*

$$\hat{\lambda}_1 < \hat{\lambda}_2 < \cdots < \hat{\lambda}_m. \tag{3.9}$$

Then

$$\lambda_1 \leqslant \hat{\lambda}_1, \qquad \hat{\lambda}_m \leqslant \lambda_n, \tag{3.10}$$

where the λ_j *are eigenvalues of* A *ordered according to* (1.5).

Proof. We first note that the eigenvalues of the real symmetric tridiagonal matrix $\hat{T}_m$ are pairwise distinct because all subdiagonal elements $\sigma_j^{1/2}$ are nonvanishing. The inequalities (3.10) follow from

the fact that $\hat{T}_m$ is an orthogonal section of A. For instance,

$$\lambda_1 = \min_{u \neq 0} \frac{u^T A u}{u^T u} \leqslant \min_{u \neq 0} \frac{u^T V_m^T A V_m u}{u^T u} = \hat{\lambda}_1. \qquad \square$$

We now consider the recurrence relation for the residual vectors s_j. It is well known that the residual vectors satisfy

$$\langle s_j, s_k \rangle_A = 0, \quad j \neq k. \tag{3.11}$$

Introduce the $n \times m$ matrix

$$S_m := [s_0, s_1, \ldots, s_{m-1}]. \tag{3.12}$$

The residual vectors satisfy a three-term recurrence relation. The following lemma displays this relation in matrix form.

Lemma 3.4. *The residual vectors s_j generated by Algorithm* 3.1 *satisfy*

$$S_m B_m = A S_m F_m - s_m \tilde{e}_m^T, \tag{3.13}$$

provided that the bilinear form $\langle \cdot, \cdot \rangle_A$ does not vanish for certain vectors. Here S_m is given by (3.12), *B_m is a tridiagonal matrix of the form*

$$B_m = \begin{bmatrix} \beta_{00} & \beta_{10} & & & 0 \\ 1 & \beta_{11} & \beta_{21} & & \\ & 1 & \beta_{22} & & \\ & & & \ddots & \\ & & \ddots & \ddots & \beta_{m-1,m-2} \\ 0 & & & 1 & \beta_{m-1,m-1} \end{bmatrix},$$

and F_m is a diagonal matrix

$$F_m = \operatorname{diag}[\phi_0, \phi_1, \ldots, \phi_{m-1}].$$

The entries of B_m and F_m are given by

$$\beta_{j,j-1} := (\alpha_j^2 / \alpha_{j-1}^2)\sigma_j, \quad 1 \leqslant j < m,$$

$$\beta_{jj} := \begin{cases} -1, & j = 0, \\ -1 - (\alpha_j^2 / \alpha_{j-1}^2)\sigma_j, & 1 \leqslant j < m, \end{cases} \tag{3.14}$$

$$\phi_j := \begin{cases} -\alpha_0, & j = 0, \\ \alpha_j / \alpha_{j-1}, & 1 \leqslant j < m, \end{cases}$$

where the α_j and σ_j are defined in Algorithm 3.1.

Proof. The lemma follows from (3.11). The proof requires that the bilinear form $\langle \cdot, \cdot \rangle_A$ does not vanish for certain vectors. For instance, one has to require that $\langle \boldsymbol{s}_j, \boldsymbol{s}_j \rangle_A \neq 0$ and $\langle \boldsymbol{s}_j, \boldsymbol{d}_j \rangle_A \neq 0$ for $0 \leqslant j < m$. □

Introduce the matrix

$$M_m := S_m^{\mathrm{T}} A S_m,$$

and assume that M_m is nonsingular. Then the columns of the matrix

$$W_m := S_m M_m^{-1/2} \tag{3.15}$$

form an orthonormal basis of the Krylov subspace $\mathcal{K}_m(A, \boldsymbol{s}_0)$ with respect to the bilinear form $\langle \cdot, \cdot \rangle_A$ and associated "length" $\| \cdot \|_A$.

Lemma 3.5. *The matrix M_m is diagonal with nontrivial entries*

$$\tilde{\boldsymbol{e}}_1^{\mathrm{T}} M_m \tilde{\boldsymbol{e}}_1 = \alpha_0 \eta_0,$$

$$\tilde{\boldsymbol{e}}_{j+1}^{\mathrm{T}} M_m \tilde{\boldsymbol{e}}_{j+1} = -\alpha_{j-1} \alpha_j \eta_j, \quad 1 \leqslant j < m.$$

Proof. The matrix M_m is diagonal due to the orthogonality (3.11). The diagonal entries can be determined by using (3.11) and the fact that $\langle \boldsymbol{s}_j, \boldsymbol{d}_k \rangle_A = 0$ for $j > k$. □

Theorem 3.6. *Let the matrices B_m, F_m and S_m be as in Lemma* 3.4. *Then the symmetric tridiagonal matrix*

$$\hat{B}_m = [\hat{\beta}_{ij}]_{i,j=0}^{m-1} := M_m^{1/2} B_m F_m^{-1} M_m^{-1/2} \tag{3.16}$$

is an orthogonal projection of A onto the Krylov subspace $\mathcal{K}(A, A\boldsymbol{s}_0)$ with respect to the bilinear form $\langle \cdot, \cdot \rangle_A$ and associated "length" $\| \cdot \|_A$, i.e.,

$$\hat{B}_m = \langle W_m, A W_m \rangle_A, \tag{3.17}$$

where W_m is given by (3.15). *The nontrivial entries $\hat{\beta}_{ij}$ are given by*

$$\begin{aligned} \hat{\beta}_{jj} &:= \begin{cases} 1/\alpha_0, & j = 0, \\ -\alpha_{j-1}/\alpha_j - (\alpha_j/\alpha_{j-1})\sigma_j, & 1 \leqslant j < m, \end{cases} \\ \hat{\beta}_{j+1,j} &:= \begin{cases} -((-\alpha_1)^{1/2}/\alpha_0)\sigma_1^{1/2}, & j = 0, \\ ((\alpha_{j-1}\alpha_{j+1})^{1/2}/\alpha_j)\sigma_{j+1}^{1/2}, & 1 \leqslant j \leqslant m-2, \end{cases} \end{aligned} \tag{3.18}$$

where the coefficients α_j and σ_j are given by Algorithm 3.1.

Proof. Multiplication of (3.13) from the left by $S_m^{\mathrm{T}} A$ and from the right by F_m^{-1}, using that $S_m^{\mathrm{T}} A \boldsymbol{s}_m = \boldsymbol{0}$, yields

$$M_m B_m F_m^{-1} = S_m^{\mathrm{T}} A^2 S_m.$$

Therefore

$$M_m^{1/2} B_m F_m^{-1} M_m^{-1/2} = M_m^{-1/2} S_m^{\mathrm{T}} A^2 S_m M_m^{-1/2}. \tag{3.19}$$

The right-hand side of (3.19) can be written as $W_m^{\mathrm{T}} A^2 W_m$ and (3.17) follows. Formulas (3.18) are now obtained from (3.16) and Lemmas 3.4 and 3.5. □

Depending on the signs of the coefficients α_j determined by Algorithm 3.1, the matrix $\hat{B}_m$ is either real symmetric or complex symmetric.

Corollary 3.7. *Assume that the matrix A is positive definite. Then the matrix $\hat{B}_m$ defined in Theorem 3.6 has only real entries.*

Proof. Lemma 3.5 yields

$$0 < s_j^{\mathrm{T}} A s_j = \tilde{e}_{j+1}^{\mathrm{T}} M_m \tilde{e}_{j+1} = \begin{cases} \alpha_0 \eta_0, & j = 0, \\ -\alpha_{j-1}\alpha_j \eta_j, & 1 \leqslant j < m. \end{cases}$$

Since $\eta_j > 0$ for all j, it follows that $\operatorname{sign}(\alpha_j) = (-1)^j$. Thus, the elements $\hat{\beta}_{j+1,j}$ are real. □

Corollary 3.8. *Let the eigenvalues $\{\hat{\mu}_j\}_{j=1}^m$ of the matrix $\hat{B}_m$ be ordered so that*

$$\frac{1}{\hat{\mu}_1} \leqslant \frac{1}{\hat{\mu}_2} \leqslant \cdots \leqslant \frac{1}{\hat{\mu}_m}.$$

Assume that $\hat{\mu}_1 < 0$ and $\hat{\mu}_m > 0$. Then

$$\hat{\mu}_1 \leqslant \lambda_\ell, \qquad \lambda_{\ell+1} \leqslant \hat{\mu}_m, \tag{3.20}$$

where λ_ℓ and $\lambda_{\ell+1}$ are the largest negative and smallest positive eigenvalues of A, respectively, see (1.5).

Proof. We first establish that the eigenvalues $\hat{\mu}_j$ are real. Let $\boldsymbol{u} \in \mathbb{R}^m \setminus \{\boldsymbol{0}\}$ and consider the quotient

$$\frac{\boldsymbol{u}^{\mathrm{T}} \hat{B}_m \boldsymbol{u}}{\boldsymbol{u}^{\mathrm{T}} \boldsymbol{u}} = \frac{\boldsymbol{u}^{\mathrm{T}} M_m^{-1/2} S_m^{\mathrm{T}} A^2 S_m M_m^{-1/2} \boldsymbol{u}}{\boldsymbol{u}^{\mathrm{T}} \boldsymbol{u}} = \frac{\boldsymbol{v}^{\mathrm{T}} S_m^{\mathrm{T}} A^2 S_m \boldsymbol{v}}{\boldsymbol{v}^{\mathrm{T}} M_m \boldsymbol{v}}, \tag{3.21}$$

where $\boldsymbol{v} = M_m^{-1/2} \boldsymbol{u}$. Comparing the left-hand side and the right-hand side of (3.21) shows that the eigenvalues of $\hat{B}_m$ are eigenvalues of the generalized eigenvalue problem

$$S_m^{\mathrm{T}} A^2 S_m \hat{\boldsymbol{v}} = \hat{\mu} M_m \hat{\boldsymbol{v}}. \tag{3.22}$$

The matrices $S_m^{\mathrm{T}} A^2 S_m$ and M_m are real and symmetric, and $S_m^{\mathrm{T}} A^2 S_m$ is positive definite. Therefore the eigenvalues $\hat{\mu}$ in (3.22) are real. Thus, the spectrum of $\hat{B}_m$ is real. Let $\operatorname{span}\{U\}$ denote the span of the columns of the matrix U. Then

$$\frac{1}{\lambda_\ell} = \min_{\boldsymbol{u} \in \mathbb{R}^n \setminus \{\boldsymbol{0}\}} \frac{\langle \boldsymbol{u}, A^{-1}\boldsymbol{u} \rangle}{\langle \boldsymbol{u}, \boldsymbol{u} \rangle} \leqslant \min_{\boldsymbol{u} \in \operatorname{span}\{S_m\} \setminus \{\boldsymbol{0}\}} \frac{\langle \boldsymbol{u}, \boldsymbol{u} \rangle_A}{\langle \boldsymbol{u}, A\boldsymbol{u} \rangle_A} = \min_{\boldsymbol{u} \in \mathbb{R}^m \setminus \{\boldsymbol{0}\}} \frac{\boldsymbol{u}^{\mathrm{T}} M_m \boldsymbol{u}}{\boldsymbol{u}^{\mathrm{T}} M_m^{1/2} \hat{B}_m M_m^{1/2} \boldsymbol{u}} = \frac{1}{\hat{\mu}_1}.$$

Assume that $\hat{\mu}_1 < 0$. Then it follows that $\hat{\mu}_1 \leqslant \lambda_\ell$. Similarly,

$$\frac{1}{\lambda_{\ell+1}} = \max_{u \in \mathbb{R}^n \setminus \{0\}} \frac{\langle u, A^{-1}u \rangle}{\langle u, u \rangle} \geqslant \max_{u \in \operatorname{span}\{S_m\} \setminus \{0\}} \frac{\langle u, u \rangle_A}{\langle u, Au \rangle_A} = \max_{u \in \mathbb{R}^m \setminus \{0\}} \frac{u^{\mathrm{T}} M_m u}{u^{\mathrm{T}} M_m^{1/2} \hat{B}_m M_m^{1/2} u} = \frac{1}{\hat{\mu}_m}.$$

Assume that $\hat{\mu}_m > 0$. Then $\hat{\mu}_m \leqslant \lambda_{\ell+1}$. □

Corollary 3.8 establishes that the complex symmetric tridiagonal matrices $\hat{B}_m$ only has real eigenvalues. We remark that many complex symmetric tridiagonal matrices have eigenvalues with non-vanishing imaginary part. For instance, the matrix

$$\begin{bmatrix} 0 & \mathrm{i} \\ \mathrm{i} & 0 \end{bmatrix}, \quad \mathrm{i} := \sqrt{-1},$$

has eigenvalues $\pm \mathrm{i}$.

Theorem 3.9. *The eigenvalues of the matrix $\hat{B}_m$, defined in Theorem 3.6, are the zeros of the residual polynomial p_m that is implicitly determined by the CR method.*

Proof. Write the relation (3.13) in the form

$$S_m B_m F_m^{-1} = A S_m - s_m \tilde{e}_m^{\mathrm{T}} \phi_{m-1}^{-1}, \tag{3.23}$$

and introduce the vector $p_m(z) = [p_0(z), p_1(z), \ldots, p_{m-1}(z)]^{\mathrm{T}}$, where the p_j are residual polynomials that are implicitly determined by the CR method. It follows from (3.23) that

$$p_m^{\mathrm{T}}(z) B_m F_m^{-1} = z p_m^{\mathrm{T}}(z) - p_m(z) \phi_m^{-1} \tilde{e}_m^{\mathrm{T}}.$$

Thus, the zeros of $p_m(z)$ are eigenvalues of $B_m F_m^{-1}$ and vice versa. The matrix $B_m F_m^{-1}$ is similar to $\hat{B}_m$. □

The following theorems are concerned with the sensitivity of the eigenvalues of the matrices $\hat{T}_m$ and $\hat{B}_m$ to perturbations in the computations for Algorithm 3.1 caused by round-off errors. In exact arithmetic the residual vectors generated by the CR method are orthogonal with respect to the bilinear form $\langle \cdot, \cdot \rangle_A$. However, due to round-off, orthogonality can be lost. We wish to determine how this loss of orthogonality affects the computed eigenvalues of $\hat{B}_m$. Let

$$w = [w_1, w_2, \ldots, w_n]^{\mathrm{T}} := \frac{1}{\|s_0\|_A} \Lambda^{1/2} Q^{\mathrm{T}} s_0$$

and define the measure

$$\mathrm{d}\sigma_n(z) := \sum_{j=1}^{n} \delta(z - \lambda_j) w_j^2, \tag{3.24}$$

where Q is the eigenvector matrix, Λ the eigenvalue matrix and λ_j the eigenvalues of A, see (1.4)–(1.5), and δ denotes the Dirac δ-function. For future reference we note that

$$\langle w, w \rangle = \int_{\mathbb{R}} \mathrm{d}\sigma_n(z) = 1. \tag{3.25}$$

We assume for notational simplicity that the eigenvalues λ_j are pairwise distinct and that all weights w_j^2 are nonvanishing. Note that the weights w_j^2 can be negative. Introduce the bilinear form

$$(f,g)_z := \int_{\mathbb{R}} f(z)g(z)\,\mathrm{d}\sigma_n(z)$$

for polynomials f and g of degree less than n. The residual polynomials p_j generated by the CR method satisfy $s_j = p_j(A)s_0$ and are orthogonal with respect to this bilinear form; we have

$$\begin{aligned}(p_j, p_k)_z &= \int_{\mathbb{R}} p_j(z)p_k(z)\,\mathrm{d}\sigma_n(z) = w^{\mathrm{T}} p_j(\Lambda)\Lambda p_k(\Lambda)w \\ &= \frac{1}{\|s_0\|_A^2} s_0^{\mathrm{T}} p_j(A)Ap_k(A)s_0 = \frac{1}{\|s_0\|_A^2}\langle s_j, s_k\rangle_A \\ &= \begin{cases} 0, & j \neq k, \\ \frac{\|s_j\|_A^2}{\|s_0\|_A^2}, & j = k. \end{cases}\end{aligned} \tag{3.26}$$

Theorem 3.10. *Assume that the matrix $\hat{B}_m$ has pairwise distinct eigenvalues $\hat{\mu}_j$ and that the eigenvector matrix in the spectral factorization*

$$\hat{B}_m = \hat{Q}_m \hat{\Lambda}_m \hat{Q}_m^{\mathrm{T}},$$

satisfies $\hat{Q}_m^{\mathrm{T}}\hat{Q}_m = I$, where $\hat{\Lambda}_m = \mathrm{diag}[\hat{\mu}_1, \hat{\mu}_2, \ldots, \hat{\mu}_m]$. Let

$$\hat{w}_j^2 := c_m(\tilde{e}_1^{\mathrm{T}}\hat{Q}_m\tilde{e}_j)^2, \quad 1 \leqslant j \leqslant m, \tag{3.27}$$

with the scaling factor $c_m > 0$ chosen so that

$$\sum_{j=1}^{m} \hat{w}_j^2 = 1. \tag{3.28}$$

Then

$$G_m(f) := \sum_{j=1}^{m} f(\hat{\mu}_j)\hat{w}_j^2 \tag{3.29}$$

is the m-point Gaussian quadrature rule associated with $\mathrm{d}\sigma_n$. It satisfies, for $\varphi_j(z) := z^j$,

$$G_m(\varphi_j) = (\varphi_j, \varphi_0)_z, \quad 0 \leqslant j < 2m.$$

Proof. Let $\tilde{Q}_m \in \mathbb{C}^{m\times m}$ be an eigenvector matrix associated with $\hat{B}_m$, i.e., the columns of $\tilde{Q}_m$ are linearly independent eigenvectors of $\hat{B}_m$. The complex symmetry of $\hat{B}_m$ yields that $\tilde{Q}_m^{\mathrm{T}}\tilde{Q}_m = \tilde{C}_m$, where $\tilde{C}_m$ is a diagonal matrix. Assuming that $\tilde{C}_m^{-1}$ exists, we can define $\hat{Q}_m := \tilde{Q}_m\tilde{C}_m^{-1/2}$. The matrix $\hat{Q}_m$ satisfies the conditions of the theorem. Its columns are uniquely determined up to a factor ± 1. In particular, the quantities $(\tilde{e}_1^{\mathrm{T}}\hat{Q}_m\tilde{e}_j)^2$ in (3.27) are uniquely defined.

The fact that $\hat{B}_m$ is a complex symmetric tridiagonal matrix with real-valued diagonal entries and real eigenvalues, makes it possible to interpret the elements of $\hat{B}_m$ as recursion coefficients of orthonormal polynomials with respect to an indefinite weight function with support on (part of) the

real axis; see, e.g., [22]. The entries of the eigenvector matrix $\hat{Q}_m$ can be interpreted as values of the orthonormal polynomials at the nodes $\hat{\mu}_j$. The theorem can now be shown in exactly the same manner as analogous results for Gaussian quadrature rules associated with positive measures. The latter are discussed, e.g., in [13, 27]. Gaussian quadrature rules with negative weights are considered in [28]. □

Introduce the modified moments

$$v_j := (p_j, p_0)_z = \frac{1}{\|s_0\|_A^2}\langle s_j, s_0\rangle_A, \quad j \geqslant 0. \tag{3.30}$$

If the residual vectors were generated by the CR method without round-off errors, then $v_0 = 1$ and $v_j = 0$ for $j > 0$. Due to round-off errors, the residual vectors s_j actually computed by Algorithm 3.1 might not be orthogonal with respect to the bilinear form $\langle \cdot, \cdot \rangle_A$, i.e., the modified moments v_j, $j \geqslant 1$, might be contaminated by errors $\mathrm{d}v_j$. These errors give rise to errors in the entries of the matrix $\hat{B}_m$ and in its eigenvalues $\hat{\mu}_j$. The following theorem shows how perturbations $\mathrm{d}v_j$ propagate and give rise to perturbations $\mathrm{d}\hat{\mu}_j$ in $\hat{\mu}_j$.

Theorem 3.11. *Let* $\mathrm{d}y$ *denote the differential of* y. *Then*

$$\mathrm{d}\hat{\mu}_j = \hat{w}_j^{-2}(p_m'(\hat{\mu}_j))^{-2} \sum_{k=1}^{2m-1} \left(\frac{\|s_0\|_A}{\|s_k\|_A}\right)^2 (p_m p_{m-1,j}, p_k)_z \, \mathrm{d}v_k, \quad 1 \leqslant j \leqslant m, \tag{3.31}$$

where $p_{m-1,j}(z) := p_m(z)/(z - \hat{\mu}_j)$.

Proof. The theorem can be shown as in Lemma 3.3 of [4]. □

The theorem shows that an eigenvalue $\hat{\mu}_j$ associated with a tiny Gaussian weight $\hat{w}_j^2$ can be very sensitive to perturbations in the modified moments (3.30). Thus, eigenvalues can be quite sensitive to loss of orthogonality in the residual vectors s_k computed by the CR method. In order to make sure that the residual vectors generated by the CR method are close to orthogonal, we choose a fairly small value of the parameter m in Algorithm 3.1. Moreover, when updating the set $\mathbb{I}$ for Richardson iteration, we ignore those computed eigenvalues $\hat{\mu}_j$ of the matrix $\hat{B}_m$ for which the associated Gaussian weights $\hat{w}_j^2$ are tiny. In this manner we seek to achieve that even in the presence of round-off errors, the sets used to determine iteration parameters for Richardson iteration satisfy $\mathbb{I} \subset \mathbb{I}_0$, where $\mathbb{I}_0$ is defined by (2.10). Details of how $\mathbb{I}$ is updated are described in Section 4.

We turn to the sensitivity of the eigenvalues of the matrix $\hat{T}_m$. The CR method (Algorithm 3.1) generates vectors d_j' and scalars η_j. Introduce the normalized vectors $\tilde{d}_j := \eta_j^{-1/2} d_j'$ and define the matrix

$$\tilde{D}_m = [\tilde{d}_0, \tilde{d}_1, \ldots, \tilde{d}_{m-1}].$$

Then

$$\hat{T}_m = \tilde{D}_m^{\mathrm{T}} A \tilde{D}_m;$$

see the proof of Theorem 3.2. If the computations were carried out in exact arithmetic, then $\tilde{D}_m^{\mathrm{T}} \tilde{D}_m = I$. Due to round-off the orthogonality of the vectors $\tilde{d}_j$ can be lost. We wish to determine

how this loss of orthogonality affects the computed eigenvalues of $\hat{T}_m$. We proceed analogously as above. Thus, let

$$\boldsymbol{u} = [u_1, u_2, \ldots, u_n]^{\mathrm{T}} := \frac{1}{\|\tilde{\boldsymbol{d}}_0\|} Q^{\mathrm{T}} \tilde{\boldsymbol{d}}_0$$

and analogously to (3.24), we define the measure

$$\mathrm{d}\tilde{\sigma}_n(z) := \sum_{j=1}^{n} \delta(z - \lambda_j) u_j^2.$$

Then

$$\langle \boldsymbol{u}, \boldsymbol{u} \rangle = \int_{\mathbb{R}} \mathrm{d}\tilde{\sigma}_n(z) = 1. \tag{3.32}$$

We assume for notational simplicity that the eigenvalues λ_j are pairwise distinct and that all weights u_j^2 are positive. Introduce the inner product

$$(f, g) := \int_{\mathbb{R}} f(z) g(z) \, \mathrm{d}\tilde{\sigma}_n(z) \tag{3.33}$$

for polynomials f and g of degree less than n. The polynomials $\tilde{p}_j$, such that

$$\tilde{\boldsymbol{d}}_j = \tilde{p}_j(A) \tilde{\boldsymbol{d}}_0, \quad j \geqslant 0,$$

are implicitly defined by Algorithm 3.1. They are orthogonal with respect to the inner product (3.33); we have,

$$\begin{aligned}(\tilde{p}_j, \tilde{p}_k) &= \int_{\mathbb{R}} \tilde{p}_j(z) \tilde{p}_k(z) \, \mathrm{d}\tilde{\sigma}_n(z) = \boldsymbol{u}^{\mathrm{T}} \tilde{p}_j(\Lambda) \tilde{p}_k(\Lambda) \boldsymbol{u} \\ &= \frac{1}{\|\tilde{\boldsymbol{d}}_0\|^2} \tilde{\boldsymbol{d}}_0^{\mathrm{T}} \tilde{p}_j(A) \tilde{p}_k(A) \tilde{\boldsymbol{d}}_0 = \frac{1}{\|\tilde{\boldsymbol{d}}_0\|^2} \langle \tilde{\boldsymbol{d}}_j, \tilde{\boldsymbol{d}}_k \rangle \\ &= \begin{cases} 0, & j \neq k, \\ \dfrac{\|\tilde{\boldsymbol{d}}_j\|^2}{\|\tilde{\boldsymbol{d}}_0\|^2}, & j = k. \end{cases}\end{aligned} \tag{3.34}$$

The m-point Gaussian quadrature rule associated with $\mathrm{d}\tilde{\sigma}_n$ is given by

$$\tilde{G}_m(f) := \sum_{j=1}^{m} f(\hat{\lambda}_j) \hat{u}_j^2. \tag{3.35}$$

It satisfies, for $\varphi_j(z) := z^j$,

$$\tilde{G}_m(\varphi_j) = (\varphi_j, \varphi_0), \quad 0 \leqslant j < 2m.$$

The nodes $\hat{\lambda}_j$ in (3.35) are the eigenvalues of $\hat{T}_m$ and, because of (3.32), the weight $\hat{u}_k^2$ associated with $\hat{\lambda}_k$ is the square of the first component of the eigenvector of unit length of $\hat{T}_m$ associated with $\hat{\lambda}_k$. Details are discussed in [13], who describe an efficient algorithm for computing the nodes $\hat{\lambda}_j$ and weights $\hat{u}_j^2$ of the Gaussian quadrature rule (3.35) from $\hat{T}_m$. It follows from (3.32) that

$$\sum_{j=1}^{m} \hat{u}_j^2 = 1. \tag{3.36}$$

Define the modified moments

$$\tilde{v}_j := (\tilde{p}_j, \tilde{p}_0) = \frac{1}{\|\tilde{\boldsymbol{d}}_0\|^2} \langle \tilde{\boldsymbol{d}}_j, \tilde{\boldsymbol{d}}_0 \rangle, \quad j \geqslant 0. \tag{3.37}$$

If the vectors $\boldsymbol{d}_j'$ were computed in exact arithmetic, then $\tilde{v}_0 = 1$ and $\tilde{v}_j = 0$ for $j > 0$. Due to round-off errors, the vectors $\boldsymbol{d}_j'$ actually computed by Algorithm 3.1 might not be orthogonal with respect to the inner product $\langle \cdot, \cdot \rangle$, i.e., the modified moments $\tilde{v}_j$, $j \geqslant 1$, might be contaminated by errros $\mathrm{d}\tilde{v}_j$. These errors give rise to errors in the entries of the matrix $\hat{T}_m$ and in its eigenvalues $\hat{\lambda}_j$. The following theorem shows how perturbations $\mathrm{d}\tilde{v}_j$ propagate and give rise to perturbations $\mathrm{d}\hat{\lambda}_j$ in $\hat{\lambda}_j$.

Theorem 3.12 ([4, Lemma 3.3]). *Let* dy *denote the differential of* y. *Then*

$$\mathrm{d}\hat{\lambda}_j = \hat{u}_j^{-2} (\tilde{p}_m'(\hat{\lambda}_j))^{-2} \sum_{k=1}^{2m-1} \left(\frac{\|\tilde{\boldsymbol{d}}_0\|}{\|\tilde{\boldsymbol{d}}_k\|} \right)^2 (\tilde{p}_m \tilde{p}_{m-1,j}, \tilde{p}_k)\, \mathrm{d}\tilde{v}_k, \quad 1 \leqslant j \leqslant m,$$

where $\tilde{p}_{m-1,j}(z) := \tilde{p}_m(z)/(z - \hat{\lambda}_j)$.

The above theorem shows that eigenvalues $\hat{\lambda}_j$ associated with tiny Gaussian weights $\hat{u}_j^2$ can be very sensitive to perturbations in the modified moments (3.37), i.e., they can be quite sensitive to loss of orthogonality in the vectors $\boldsymbol{d}_k'$ computed by Algorithm 3.1. We therefore ignore eigenvalues $\hat{\lambda}_k$ associated with tiny Gaussian weights $\hat{u}_k$ when updating the set $\mathbb{I}$ used by the Richardson method. This is discussed in the next section.

4. The hybrid scheme

Before presenting the algorithm that defines our hybrid scheme, we discuss some details on how to generate iteration parameters for Richardson iteration and how to update the set $\mathbb{I}$ used to generate these parameters. Let $\mathbb{I} := [a_j, b_j] \cup [c_j, d_j]$ satisfy $\mathbb{I} \subset \mathbb{I}_0$, where the eigenvalues λ_j of A are ordered according to (1.5). Assume that we have carried out $k \geqslant m$ iterations by the CR and Richardson methods. Thus, we have determined an approximate solution $\boldsymbol{x}_k$ of (1.1) from the initial approximate solution $\boldsymbol{x}_0$. The associated residual vector $\boldsymbol{r}_k := \boldsymbol{b} - A\boldsymbol{x}_k$ can be written in terms of the residual polynomial $\boldsymbol{r}_k = p_k(A)\boldsymbol{r}_0$. Let the parameters $\{\delta_\ell\}_{\ell=0}^{k-1}$ of p_k be explicitly known. We then determine new relaxation parameters δ_ℓ, $\ell \geqslant k$, for Richardson iteration (1.10) as reciprocal values

of Leja points for the set $\mathbb{I}$ in the presence of the points $z_\ell := 1/\delta_\ell$, $0 \leqslant \ell < k$. The computation of a large number of Leja points by (2.4) can be cumbersome. We therefore replace $\mathbb{I}$ by a finite point set consisting of the union of zeros of two high degree Chebyshev polynomials, one polynomial for the interval $[a_j, b_j]$ and one for $[c_j, d_j]$. The degrees are chosen sufficiently large to make the discretization error negligible. It follows from

$$\prod_{j=0}^{k-1}(z - z_j) = p_k(z)\prod_{j=0}^{k-1}(-z_j),$$

that the Leja point z_k for $\mathbb{I}$ can be determined by maximizing $|p_k(z)|$ over $\mathbb{I}$. The value $\max_{z\in\mathbb{I}}|p_k(z)|$ is used in our criterion (2.11) for deciding when to update the set $\mathbb{I}$ and is part of the output of the following algorithm.

Algorithm 4.1 (Computation of relaxation parameter δ_k)
Input: a_j, b_j, c_j, d_j, $k>0$, $\{z_\ell\}_{\ell=0}^{k-1}$ $(z_\ell = 1/\delta_\ell)$;
Output: δ_k, $\rho_k := \max_{z\in\mathbb{I}}|p_k(z)|$, where $\mathbb{I} := [a_j, b_j] \cup [c_j, d_j]$;
Determine $z_k \in \mathbb{I}$, such that $|p_k(z_k)|\omega(z_k) = \max_{z\in\mathbb{I}}|p_k(z)|\omega(z)$;
$\delta_k := 1/z_k$; $\rho_k := |p_k(z_k)|$;

The points $z_0, z_1, \ldots, z_{k-1}$ serve as memory of previous iterations when the new relaxation parameter δ_k is determined. The presence of this memory has the effect that relaxation parameters δ_ℓ determined just after the set $\mathbb{I}$ has been increased in size, are distributed so that eigenvector components associated with eigenvalues in $\mathbb{I}$ in the residual error, that have not been damped before, will be damped more heavily than other eigenvector components for a couple of iterations.

We carry out Richardson iteration with iteration parameters determined by Algorithm 4.1 using the set $\mathbb{I} = [a_j, b_j] \cup [c_j, d_j]$ until a sufficiently accurate approximate solution has been found, or until inequality (2.11) holds. In the latter case, $\mathbb{I}$ does not contain all eigenvalues of A, and we seek to increase the rate of convergence of Richardson iteration by replacing this set by a larger set $\mathbb{I} := [a_{j+1}, b_{j+1}] \cup [c_{j+1}, d_{j+1}]$. We would like the endpoints a_{j+1}, b_{j+1}, c_{j+1} and d_{j+1} of the new set to satisfy

$$\lambda_1 \leqslant a_{j+1} \leqslant a_j, \qquad b_j \leqslant b_{j+1} \leqslant \lambda_\ell, \qquad \lambda_{\ell+1} \leqslant c_{j+1} \leqslant c_j, \qquad d_j \leqslant d_{j+1} \leqslant \lambda_n,$$

in order to secure that $[a_{j+1}, b_{j+1}] \cup [c_{j+1}, d_{j+1}] \subset \mathbb{I}_0$, We determine the new set by first carrying out m iterations by the CR method (Algorithm 3.1). This yields a new approximate solution $\boldsymbol{x}_{k+m}$, the associated residual vector $\boldsymbol{r}_{k+m}$, as well as $m \times m$ symmetric tridiagonal matrices $\hat{T}_m$ and $\hat{B}_m$ with associated Gaussian quadrature rules $\{\hat{\lambda}_j, \hat{w}_j^2\}_{j=1}^m$ and $\{\hat{\mu}_j, \hat{w}_j^2\}_{j=1}^m$, respectively. In view of the inequalities (3.10) and (3.20), an obvious way to define the new endpoints is given by

$$a_{j+1} := \min\{a_j, \hat{\lambda}_1\}, \qquad b_{j+1} := \max\{b_j, \hat{\mu}_1\}, \tag{4.1}$$

$$c_{j+1} := \min\{c_j, \hat{\mu}_m\}, \qquad d_{j+1} := \max\{d_j, \hat{\lambda}_m\}, \tag{4.2}$$

where we have assumed that the $\hat{\lambda}_j$ and $\hat{\mu}_j$ are ordered as indicated in Corollaries 3.3 and 3.8, respectively, and that $\hat{\mu}_1 < 0$ and $\hat{\mu}_m > 0$. However, computational experience from the solution of numerous problems indicates, that when the Gaussian weight $\hat{w}_k^2$ associated with $\hat{\mu}_k$ is tiny, the computed value of $\hat{\mu}_k$ may be contaminated by a large error and should not be used to update

the interval; cf. Theorem 3.11. Similarly, we ignore eigenvalues $\hat{\lambda}_k$ associated with tiny weights $\hat{u}_k^2$ when updating the set $\mathbb{I}$. Thus, we replace the updating formulas (4.1) and (4.2) by

$$a_{j+1} := \min\left\{a_j, \min_{1\leqslant k\leqslant m}\{\hat{\lambda}_k : \hat{u}_k^2 \geqslant \varepsilon_w\}\right\},$$

$$b_{j+1} := \max\left\{b_j, \max_{1\leqslant k\leqslant m}\{\hat{\mu}_k : \hat{\mu}_k < 0,\ \hat{w}_k^2 \geqslant \varepsilon_w\}\right\},$$

$$c_{j+1} := \min\left\{c_j, \min_{1\leqslant k\leqslant m}\{\hat{\mu}_k : \hat{\mu}_k > 0,\ \hat{w}_k^2 \geqslant \varepsilon_w\}\right\},$$

$$d_{j+1} := \max\left\{d_j, \max_{1\leqslant k\leqslant m}\{\hat{\lambda}_k : \hat{u}_k^2 \geqslant \varepsilon_w\}\right\},$$

where we assume that the weights are normalized according to (3.28) and (3.36). The choice of the tolerance ε_w is not very critical. When the rate of convergence can be increased significantly by letting a_{j+1} be smaller than a_j, the Gaussian weight $\hat{u}_1^2$ typically is quite large. Similarly, when the rate of convergence can be increased significantly by letting b_{j+1} be larger than b_j, the weight $\hat{w}_1^2$ typically is large. We have made similar observations with regard to the endpoints c_{j+1} and d_{j+1}.

In order to reduce the number of arithmetic operations with n-vectors required by Richardson iteration, we implement a leapfrog variant. Leapfrog implementations of iterative methods were first proposed in [25]. Let $\boldsymbol{x}_k$ be an approximate solution of (1.1) with associated residual vector $\boldsymbol{r}_k$. The computation of the approximate solutions $\boldsymbol{x}_{k+1}$ and $\boldsymbol{x}_{k+2}$, as well as of the associated residual vectors $\boldsymbol{r}_{k+1}$ and $\boldsymbol{r}_{k+2}$ by (1.10) would require 4 saxpy operations and 2 evaluations of matrix–vector products with the matrix A. On the other hand, using the recursion formulas (1.10), we obtain

$$\begin{aligned} \boldsymbol{x}_{k+2} &= \boldsymbol{x}_k + (\delta_{k+1} + \delta_k)\boldsymbol{r}_k - \delta_{k+1}\delta_k A\boldsymbol{r}_k, \\ \boldsymbol{r}_{k+2} &= \boldsymbol{b} - A\boldsymbol{x}_{k+2}. \end{aligned} \tag{4.3}$$

The computation of the left-hand side vectors in (4.3) requires only 3 saxpy operations and the evaluation of 2 matrix–vector products with the matrix A. We use the 2-step leapfrog formula (4.3) in our implementation of Richardson iteration.

We are in a position to present an algorithm for our hybrid scheme. In order to keep the number of inner product computations fairly small, we only evaluate the norm of the residual vector every 2 leapfrog Richardson steps (4.3). The set used to determine iteration parameters for Richardson iteration is in the algorithm denoted by $[a,b]\cup[c,d]$.

Algorithm 4.2 (Hybrid iteration scheme)
Input: $A\in\mathbb{R}^{n\times n}$, $\boldsymbol{b},\boldsymbol{x}_0\in\mathbb{R}^n$, $m\in\mathbb{N}$, $\varepsilon>0$, $\varepsilon_w>0$;
Output: *Approximate solution* $\boldsymbol{x}_k$, *endpoints of set* $[a,b]\cup[c,d]$;
$\boldsymbol{r}_0 := \boldsymbol{b} - A\boldsymbol{x}_0$; $\rho_0 := 0$; $k := 0$;
while $\|\boldsymbol{r}_k\|/\|\boldsymbol{r}_0\| > \varepsilon$ **do**
  if $\|\boldsymbol{r}_k\| > \rho_k\|\boldsymbol{r}_0\|$ **then**
    CR iteration:

Apply Algorithm 3.1 *to determine approximate solution* $\boldsymbol{y}_m$ *and associated residual vector* $\boldsymbol{s}_m := \boldsymbol{r}_k - A\boldsymbol{y}_m$ *of* $A\boldsymbol{y} = \boldsymbol{r}_k$, *as well as coefficients* α_j, γ_j *and* σ_j.
$\boldsymbol{x}_{k+m} := \boldsymbol{x}_k + \boldsymbol{y}_m$; $\boldsymbol{r}_{k+m} := \boldsymbol{s}_m$;
if $\|\boldsymbol{r}_{k+m}\|/\|\boldsymbol{r}_0\| \leqslant \varepsilon$ **then** *found_solution* **endif**;
Form the tridiagonal matrices $\hat{T}_m$ *and* $\hat{B}_m$ *defined by Theorems* 3.2 *and* 3.6 *from the coefficients* α_j, γ_j *and* σ_j *determined by Algorithm* 3.1. *Compute the Gaussian quadrature rules* $\{\hat{\mu}_j, \hat{w}_j^2\}_{j=1}^m$ *and* $\{\hat{\lambda}_j, \hat{u}_j^2\}_{j=1}^m$ *associated with* $\hat{B}_m$ *and* $\hat{T}_m$, *respectively*;
if $k = 0$ **then**

$$a := \min_{1\leqslant j\leqslant m}\{\hat{\lambda}_j : \hat{u}_j^2 \geqslant \varepsilon_w\};\ b := \max_{1\leqslant j\leqslant m}\{\hat{\mu}_j : \hat{\mu}_j < 0,\ \hat{w}_j^2 \geqslant \varepsilon_w\};$$

$$c := \min_{1\leqslant j\leqslant m}\{\hat{\mu}_j : \hat{\mu}_j > 0,\ \hat{w}_j^2 \geqslant \varepsilon_w\};\ d := \max_{1\leqslant j\leqslant m}\{\hat{\lambda}_j : \hat{u}_j^2 \geqslant \varepsilon_w\}$$

else

$$a := \min\left\{a, \min_{1\leqslant j\leqslant m}\{\hat{\lambda}_j : \hat{u}_j^2 \geqslant \varepsilon_w\}\right\};\ b := \max\left\{b, \max_{1\leqslant j\leqslant m}\{\hat{\mu}_j : \hat{\mu}_j < 0,\ \hat{w}_j^2 \geqslant \varepsilon_w\}\right\};$$

$$c := \min\left\{c, \min_{1\leqslant j\leqslant m}\{\hat{\mu}_j : \hat{\mu}_j > 0,\ \hat{w}_j^2 \geqslant \varepsilon_w\}\right\};\ d := \max\left\{d, \max_{1\leqslant j\leqslant m}\{\hat{\lambda}_j : \hat{u}_j^2 \geqslant \varepsilon_w\}\right\}$$

endif;
for $\ell := 1, 2, \ldots, m$ **do** $\delta_{k+\ell-1} := 1/\hat{\lambda}_\ell$ **endfor** ℓ;
$k := k + m$;
Compute δ_k *and* ρ_k *by Algorithm* 4.1 *for the set* $[a,b] \cup [c,d]$
else
Leapfrog Richardson iteration:
Compute $\{\delta_\ell\}_{\ell=k+1}^{k+4}$ *and* $\{\rho_\ell\}_{\ell=k+1}^{k+4}$ *by Algorithm* 4.1 *for the set* $[a,b] \cup [c,d]$;
$\boldsymbol{x}_{k+2} := \boldsymbol{x}_k + (\delta_{k+1} + \delta_k)\boldsymbol{r}_k - \delta_{k+1}\delta_k A\boldsymbol{r}_k$; $k := k + 2$;
$\boldsymbol{x}_{k+2} := \boldsymbol{x}_k + (\delta_{k+1} + \delta_k)\boldsymbol{r}_k - \delta_{k+1}\delta_k A\boldsymbol{r}_k$; $k := k + 2$;
endif
endwhile

In our implementation of the algorithm used for the numerical experiments in Section 5, we only require that one of the intervals $[a,b]$ or $[c,d]$ be nonempty.

5. Computed examples

This section presents a few numerical examples that illustrate the performance of our hybrid iteration scheme (Algorithm 4.2). The computer programs used were written in FORTRAN 77 and the numerical experiments were carried out on an HP 9000/777 workstation using double precision arithmetic, i.e., computations were carried out with approximately 16 significant decimal digits.

Results of our numerical experiments are displayed by figures that show the relative residual error $\log_{10}(\|\boldsymbol{r}_k\|/\|\boldsymbol{r}_0\|)$ as a function of either the number of arithmetic operations with n-vectors, or k, the number of matrix–vector product evaluations with A, where the residual vector $\boldsymbol{r}_k$ is computed by Algorithm 4.2 or by the CR method (Algorithm 3.1). We report the endpoints of the last computed set $\mathbb{I}_{\text{last}} = [a_{\text{last}}, b_{\text{last}}] \cup [c_{\text{last}}, d_{\text{last}}]$.

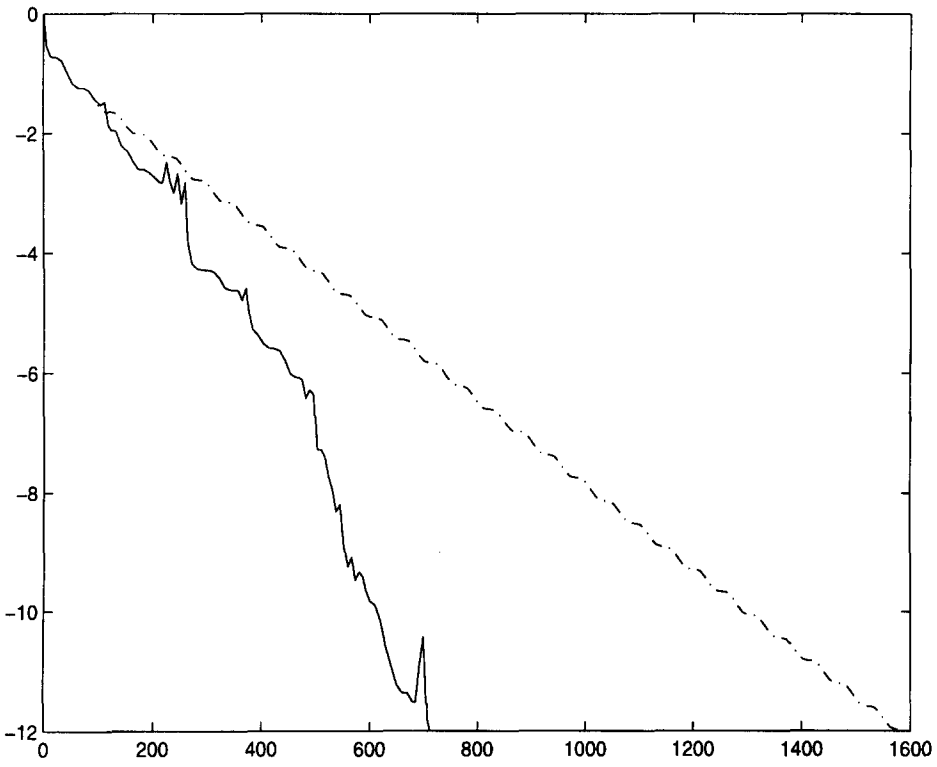

Fig. 1. Example 5.1: $\log_{10}(\|r_k\|/\|r_0\|)$ versus arithmetic vector operations with n-vectors. Legend: (–) hybrid method, (–.) CR method.

In all examples, the right-hand side vector $\boldsymbol{b}$ is chosen so that $\boldsymbol{x}=[1,1,\ldots,1]^{\mathrm{T}}$ solves (1.1). The initial approximate solution $\boldsymbol{x}_0$ has uniformly distributed random entries in $[-1,1]$. We chose $m=10$ in Algorithm 4.2 in order to keep the number of arithmetic vector operations small, and we let $\varepsilon_w := 1 \times 10^{-4}$ in order to make the computed set $\mathbb{I}$ fairly insensitive to round-off errors introduced during the computations.

Our present experimental code uses the subroutine COMQR [26] to compute the quadrature rule associated with the matrices $\hat{B}_m$. More efficient implementations can be based on modifications of the QR algorithm tailored for the computation of the Gaussian quadrature rule associated with complex symmetric tridiagonal matrices with real eigenvalues; see [3, 7] for related algorithms.

Example 5.1. Let $A=\mathrm{diag}[a_{11},a_{22},\ldots,a_{nn}]$ with half the diagonal entries equidistant in the interval $[-1/10,-1/20]$ and the other entries equidistant in $[1/20,1]$. The iterations were terminated when $\|r_k\|/\|r_0\| \leqslant 1 \times 10^{-12}$. Fig. 1 shows that Algorithm 4.2 requires significantly fewer arithmetic operations with n-vectors than Algorithm 3.1, and Fig. 2 displays that the latter algorithm requires somewhat fewer matrix–vector product evaluations with A in order to satisfy the stopping criterion. The fact that Algorithm 3.1 requires a smaller number of matrix–vector product evaluations is not surprising in view of that Eq. (1.9) holds for the residual polynomial generated by the CR method.

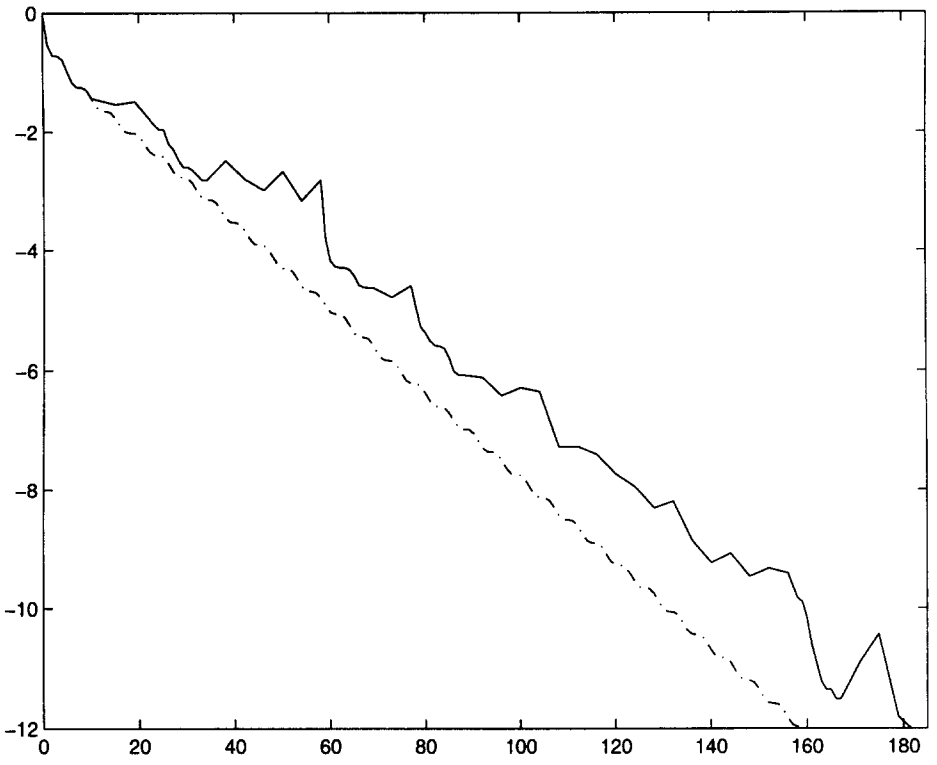

Fig. 2. Example 5.1: $\log_{10}(\|r_k\|/\|r_0\|)$ versus matrix–vector product evaluations with A. Legend: (–) hybrid method, (–.) CR method.

Figs. 1 and 2 show that if the matrix A is very sparse, so that each evaluation of a matrix–vector product is equivalent to only a few arithmetic vector operations, then the hybrid scheme requires less arithmetic work to satisfy the stopping criterion than the CR method. Algorithm 4.2 determined 5 pairs of matrices $\hat{B}_m$ and $\hat{T}_m$ and computed $[a_{\text{last}}, b_{\text{last}}] = [-9.99 \times 10^{-2}, -5.08 \times 10^{-2}]$ and $[c_{\text{last}}, d_{\text{last}}] = [5.02 \times 10^{-2}, 1.00]$. □

Example 5.2. Let A be the matrix obtained by discretizing the 2-dimensional negative Helmholtz operator $-\Delta - \tau$ on the unit square by the standard 5-point stencil with Dirichlet boundary conditions. A uniform grid with 30 grid points in each coordinate direction in the interior of the unit square yields a 900×900 matrix with diagonal entries $4 - (1/961)\tau$ and extreme eigenvalues $\lambda_1 = -2.11 \times 10^{-2}$ and $\lambda_{900} = 7.94$. We iterate until $\|r_k\|/\|r_0\| \leqslant 1 \times 10^{-2}$. Algorithm 4.2 computed $\mathbb{I}_{\text{last}} = [4.00 \times 10^{-1}, 7.82]$, i.e., the algorithm only determined one interval on the real axis. Table 1 shows the performance of Algorithms 3.1 and 4.2. In the present example, the matrix A has 5 non-vanishing diagonals, and each evaluation of a matrix–vector product with A is roughly equivalent with 5 saxpy operations. Thus, counting each matrix–vector product evaluation as 5 arithmetic vector operations, we obtain from Table 1 that Algorithm 4.2 requires 180 arithmetic vector operations while Algorithm 3.1 requires 188. This example illustrates that Algorithm 3.2 can be competitive also when only very few iterations have to be carried out. □

Table 1
Example 5.2: Arithmetic work for CR method and Algorithm 4.2

Iterative method	# matrix–vector products	# arithmetic vector oper.
Algor. 4.2	15	105
CR method	13	123

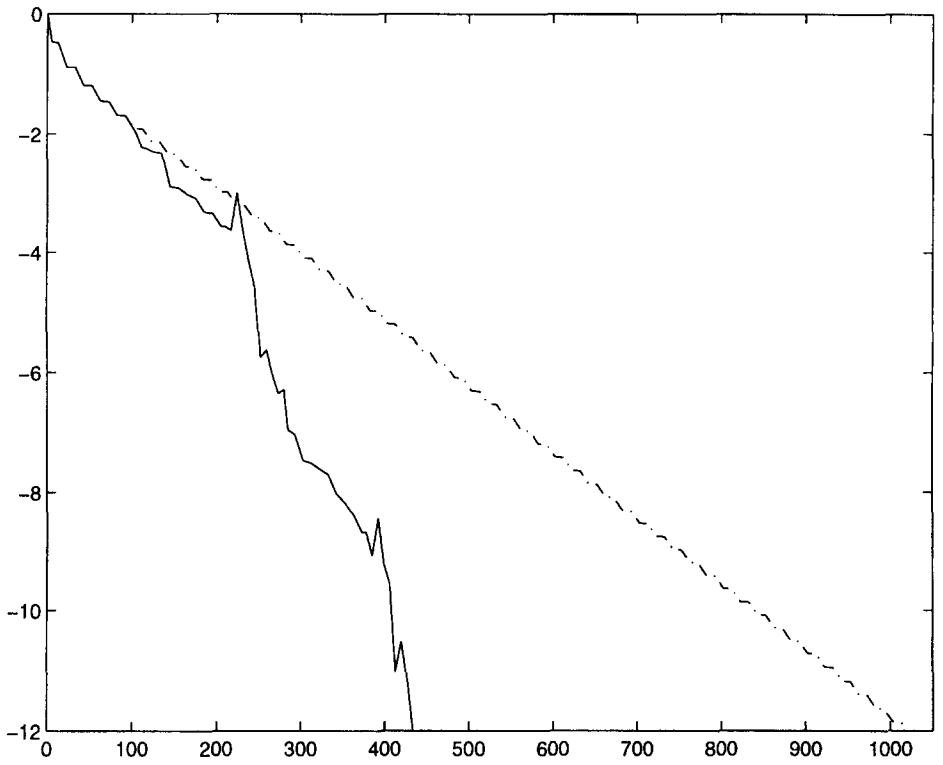

Fig. 3. Example 5.3: $\log_{10}(\|r_k\|/\|r_0\|)$ versus arithmetic vector operations with n-vectors. Legend: (–) hybrid method, (–.) CR method.

Example 5.3. Let

$$A = \begin{pmatrix} I & M \\ M^{\mathrm{T}} & 0 \end{pmatrix} \in \mathbb{R}^{4000\times 4000}. \tag{5.1}$$

Equilibrium problems give rise to matrices of this form. Here we let $M = \mathrm{diag}[m_{11}, m_{22}, \ldots, m_{2000,2000}]$ with entries m_{jj} equidistant in the interval $[\frac{1}{2}, 2]$. Then A has 2000 positive and 2000 negative eigenvalues given by

$$\hat{\lambda}_{\pm j} := \frac{1}{2} \pm \sqrt{\frac{1}{4} + m_{jj}^2}, \quad 1 \leqslant j \leqslant 2000.$$

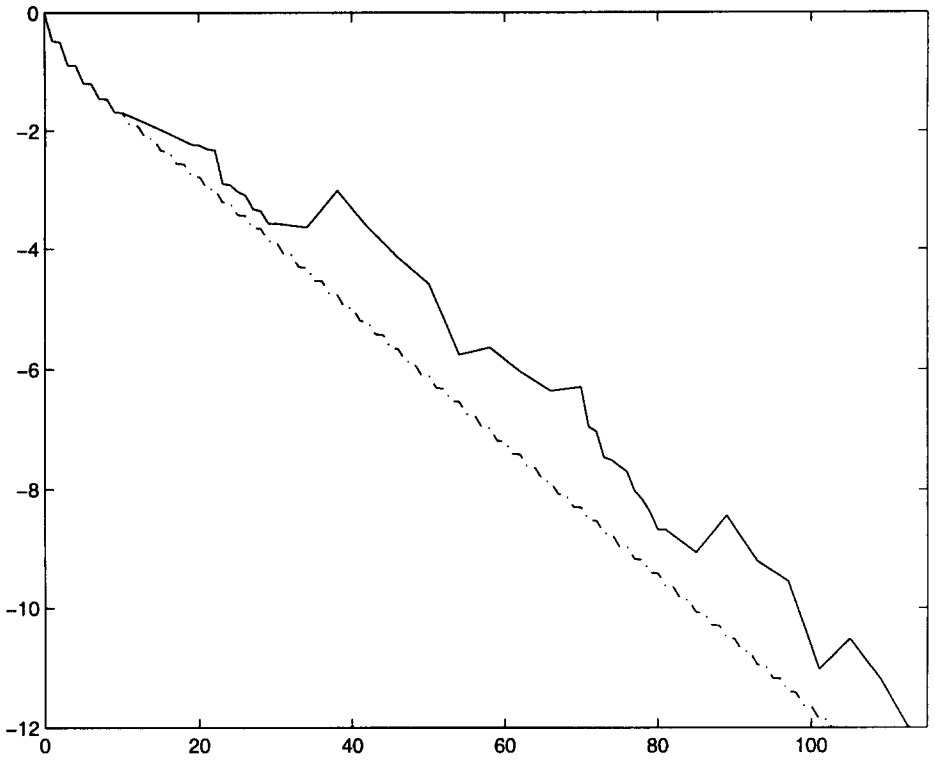

Fig. 4. Example 5.3: $\log_{10}(\|r_k\|/\|r_0\|)$ versus matrix–vector product evaluations with A. Legend: (–) hybrid method, (–.) CR method.

The iterations were terminated when $\|r_k\|/\|r_0\| \leqslant 1 \times 10^{-12}$. Fig. 3 shows that Algorithm 4.2 requires significantly fewer arithmetic operations with n-vectors than Algorithm 3.1, and Fig. 4 displays that the latter algorithm requires somewhat fewer matrix–vector product evaluations with A in order to satisfy the stopping criterion. Figs. 3 and 4 show that if the matrix A is very sparse, so that each evaluation of a matrix–vector product is equivalent to only a few arithmetic operations with n-vectors, then the hybrid scheme requires less arithmetic work to satisfy the stopping criterion than the CR method. Algorithm 4.2 computed 3 pairs of matrices $\hat{B}_m$ and $\hat{T}_m$ and determined $[a_{\text{last}}, b_{\text{last}}] = [-1.56, -2.12 \times 10^{-1}]$ and $[c_{\text{last}}, d_{\text{last}}] = [1.21, 2.56]$.

6. Conclusion

We have demonstrated that a hybrid iteration scheme based on the CR method and Richardson iteration can require significantly fewer arithmetic vector operations than the CR method to achieve a desired reduction of the residual error. The number of matrix–vector product evaluations with the matrix A for the hybrid and CR methods are almost the same. Numerous numerical experiments suggest the hybrid method to be most competitive when the symmetric indefinite matrix in the linear system (1.1) is large, has a structure that allows rapid computation of matrix–vector products and is

not very ill-conditioned. Matrices for which matrix–vector products can be evaluated quickly include sparse matrices, Toeplitz matrices and Toeplitz-plus-Hankel matrices.

Most of the iterations by the hybrid scheme are leapfrog Richardson iterations (4.3). The simplicity of the latter makes them attractive for implementation on parallel computers; see, e.g., [25] for a discussion. Therefore the hybrid scheme would appear to be attractive to implement on parallel computers.

References

[1] S.F. Ashby, T.A. Manteuffel, P.E. Saylor, A taxonomy for conjugate gradient methods, SIAM J. Numer. Anal. 27 (1990) 1542–1568.
[2] C. Brezinski, M. Redivo-Zaglia, Hybrid procedures for solving linear systems, Numer. Math. 67 (1994) 1–19.
[3] A. Bunse-Gerstner, An analysis of the HR algorithm for computing the eigenvalues of a matrix, Linear Algebra Appl. 35 (1981) 155–173.
[4] D. Calvetti, L. Reichel, A hybrid iterative method for symmetric positive definite linear systems of equations, Numer. Algorithms 11 (1996) 79–98.
[5] D. Calvetti, L. Reichel, An adaptive Richardson iterative method for symmetric indefinite linear systems, Numer. Algorithms 12 (1996) 125–149.
[6] P. Concus, G.H. Golub, D.P. O'Leary, A generalized conjugate gradient method for the numerical solution of elliptic partial differential equations, in: J.R. Bunch, D.J. Rose (Eds.), Sparse Matrix Computations, Academic Press, New York, 1976, pp. 309–332.
[7] J.K. Cullum, R.A. Willoughby, A QL procedure for computing the eigenvalues of complex symmetric tridiagonal matrices, SIAM J. Matrix Anal. Appl. 17 (1996) 83–109.
[8] C. de Boor, J.R. Rice, Extremal polynomials with application to Richardson iteration for indefinite linear systems, SIAM J. Sci. Stat. Comput. 3 (1982) 47–57.
[9] J.J. Dongarra, I.S. Duff, D.C. Sorensen, H.A. van der Vorst, Solving Linear Systems on Vector and Shared Memory Computers, SIAM, Philadelphia, 1991.
[10] H.C. Elman, Y. Saad, P.E. Saylor, A hybrid Chebyshev Krylov subspace algorithm for solving nonsymmetric systems of linear equations, SIAM J. Sci. Statist. Comput. 7 (1986) 840–855.
[11] B. Fischer, Polynomial Based Iteration Methods for Symmetric Linear Systems, Wiley, Chichester, 1996.
[12] G.H. Golub, C.F. Van Loan, Matrix Computations, 2nd ed., Johns Hopkins University Press, Baltimore, 1989.
[13] G.H. Golub, J.H. Welsch, Calculation of Gauss quadrature rules, Math. Comp. 23 (1969) 221–230.
[14] W.B. Gragg, L. Reichel, On the application of orthogonal polynomials to the iterative solution of linear systems of equations with indefinite or non-Hermitian matrices, Linear Algebra Appl. 88–89 (1987) 349–371.
[15] F. Leja, Sur certaines suits liées aux ensembles plans et leur application à la representation conforme, Ann. Polon. Math. 4 (1957) 8–13.
[16] R.B. Morgan, Computing interior eigenvalues of large matrices, Linear Algebra Appl. 154–156 (1991) 289–309.
[17] N.M. Nachtigal, L. Reichel, L.N. Trefethen, A hybrid GMRES algorithm for nonsymmetric linear systems, SIAM J. Matrix Anal. Appl. 13 (1992) 796–825.
[18] D.P. O'Leary, Hybrid conjugate gradient algorithms, Ph.D. Thesis, Computer Science Department, Stanford University, Stanford, CA, 1975.
[19] C.C. Paige, B.N. Parlett, H.A. van der Vorst, Approximate solutions and eigenvalue bounds from Krylov subspaces, Numer. Linear Algebra Appl. 2 (1995) 115–135.
[20] C.C. Paige, M.A. Saunders, Solution of sparse indefinite systems of linear equations, SIAM J. Numer. Anal. 12 (1975) 617–624.
[21] L. Reichel, The application of Leja points to Richardson iteration and polynomial preconditioning, Linear Algebra Appl. 154–156 (1991) 389–414.
[22] L. Reichel, Construction of polynomials that are orthogonal with respect to a discrete bilinear form, Adv. Comput. Math. 1 (1993) 241–258.

[23] H. Rutishauser, Theory of gradient methods, in: M. Engeli, Th. Ginsburg, H. Rutishauser, E. Stiefel (Eds.), Refined Iterative Methods for Computation of the Solution and the Eigenvalues of Self-Adjoint Boundary Value Problems, Birkhäuser, Basel, 1959, pp. 24–49.
[24] Y. Saad, Iterative solution of indefinite symmetric linear systems by methods using orthogonal polynomials over two disjoint intervals, SIAM J. Numer. Anal. 29 (1983) 784–811.
[25] P.E. Saylor, Leapfrog variants of iterative methods for linear algebraic equations, J. Comput. Appl. Math. 24 (1988) 169–193.
[26] B. Smith, J. Boyle, Y. Ikebe, V. Klema, C. Moler, Matrix Eigensystem Routines: EISPACK Guide, 2nd ed., Springer, Berlin, 1976.
[27] J. Stoer, R. Bulirsch, Introduction to Numerical Analysis, Springer, New York, 1980.
[28] G.W. Struble, Orthogonal polynomials: variable-signed weight functions, Numer. Math. 5 (1963) 88–94.