BIT manuscript No.
(will be inserted by the editor)

The structure of iterative methods for symmetric linear discrete ill-posed problems

L. Dykes · F. Marcellán · L. Reichel

Received: date / Accepted: date

Abstract The iterative solution of large linear discrete ill-posed problems with an error contaminated data vector requires the use of specially designed methods in order to avoid severe error propagation. Range restricted minimal residual methods have been found to be well suited for the solution of many such problems. This paper discusses the structure of matrices that arise in a range restricted minimal residual method for the solution of large linear discrete ill-posed problems with a symmetric matrix. The exploitation of the structure results in a method that is competitive with respect to computer storage, number of iterations, and accuracy.

Keywords ill-posed problem · iterative method · truncated iteration

Mathematics Subject Classification (2000) 65F10 · 65F22

Dedicated to Axel Ruhe on the occasion of his 70th birthday.

L. Dykes
Department of Mathematical Sciences
Kent State University, Kent, OH 44242, USA
and
University School, Hunting Valley, OH 44022, USA
E-mail: ldykes@math.kent.edu

F. Marcellán
Departamento de Matemáticas
Universidad Carlos III de Madrid
Avenida de la Universidad 30, 28911 Leganés, Spain
E-mail: pacomarc@ing.uc3m.es

L. Reichel
Department of Mathematical Sciences
Kent State University, Kent, OH 44242, USA
E-mail: reichel@math.kent.edu

1 Introduction

This paper describes range restricted iterative methods for the computation of approximate solutions of linear systems of equations

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

with a large symmetric matrix A with many eigenvalues of different orders of magnitude close to zero. Thus, A is very ill-conditioned and may be singular. Linear systems of equations (1.1) with a matrix of this kind are commonly referred to as linear discrete ill-posed problems. They are obtained, for instance, by the discretization of linear ill-posed problems, such as Fredholm integral equations of the first kind with a smooth kernel.

In linear discrete ill-posed problems that arise in science and engineering, the right-hand side vector b represents available data, which is contaminated by an error e that stems from measurement inaccuracies. Thus,

$$b = \hat{b} + e, \tag{1.2}$$

where $\hat{b} \in \mathbb{R}^m$ denotes the unknown error-free right-hand side associated with b. We refer to the error vector e as "noise."

Assume that the linear system of equations with the unknown error-free right-hand side

$$Ax = \hat{b} \tag{1.3}$$

is consistent and denote the solution of minimal Euclidean norm by $\hat{x}$. We would like to determine an accurate approximation of $\hat{x}$ by computing a suitable approximate solution of the available linear system of equations (1.1). Because of the severe ill-conditioning of the matrix A and the error e in b, the least-squares solution of minimal norm of (1.1) generally is so contaminated by propagated error that it does not furnish a useful approximation of $\hat{x}$.

A popular approach to determine a meaningful approximation of $\hat{x}$ is to apply a Krylov subspace iterative method to the solution of (1.1) and terminate the iterations sufficiently early. This way of computing an approximate solution is known as truncated iteration. It is easy to show that the sensitivity of the computed solution to the error e increases with the number of iterations. Let x_k denote the kth iterate determined by a minimal residual Krylov subspace iterative method. These methods are related to the conjugate gradient method; see below. Then the difference $x_k - \hat{x}$ typically decreases as k increases and is small, but increases rapidly with k when k is large. This behavior of the iterates is commonly referred to as semiconvergence. The growth of the difference $x_k - \hat{x}$ for large values of k is caused by severe propagation of the error e in b and of round-off errors introduced during the computations. It is important to terminate the iterations before the difference $x_k - \hat{x}$ grows with k.

When an estimate of the norm of e is available, the discrepancy principle can be used to determine how many iterations to carry out. We will use this stopping criterion in the computed examples of Section 4; however, other stopping rules also

can be applied in conjunction with the iterative method discussed, such as the quasi-optimality principle, generalized cross validation, and the L-curve; see, e.g., [1,13, 14,19] for discussions and references.

Let $\mathscr{R}(M)$ and $\mathscr{N}(M)$ denote the range and null space, respectively, of a matrix M. It is natural to apply iterative methods with iterates in $\mathscr{R}(A)$ to the approximate solution of (1.1), because these iterates are orthogonal to $\mathscr{N}(A)$. If a minimal residual iterative method with iterates in $\mathscr{R}(A)$ is applied to the solution of (1.3) with initial iterate $x_0 = 0$, then, in exact arithmetic, the method will determine the desired minimal-norm solution $\hat{x}$. We refer to iterative methods that generate approximate solutions in $\mathscr{R}(A)$ as *range restricted*. Hanke [9] analyzed the convergence properties of range restricted minimal residual Krylov subspace methods when used in conjunction with the discrepancy principle, and used an Ortodir-type implementation in the computed examples. This implementation was previously discussed in [5]. An improved implementation, that is less sensitive to round-off errors introduced during the computation, has recently been described in [15, Section 3]. This implementation was derived by first considering a range restricted GMRES-type method for the solution of linear discrete ill-posed problems with a square nonsymmetric matrix A, and then simplifying this method by exploiting the symmetry of A. It is the purpose of the present paper to further investigate the structure of the matrices of the reduced problems determined by range restricted minimal residual Krylov subspace methods. This results in a new efficient implementation that requires less storage than the implementation [16] of [15, Algorithm 3.1]. Our new implementation only requires storage of few m-vectors, whose number is bounded independently of the number of iterations carried out.

This paper is organized as follows. Section 2 discusses the structure of the matrices that arise in our range restricted minimal residual Krylov subspace methods. In Section 3, we exploit this structure to derive a new iterative method. Computed examples are presented in Section 4 and concluding remarks can be found in Section 5.

We will for notational simplicity assume that the right-hand side vector b in (1.1) is scaled to satisfy $\|b\| = 1$. Throughout this paper $\|\cdot\|$ denotes the Euclidean vector norm. The initial iterate for all iterative methods is chosen to be $x^{(0)} = 0$.

2 The structure of matrices in range restricted iterative methods

We consider iterative methods that determine iterates in $\mathscr{R}(A^\ell)$ for some integer $\ell \geq 1$. The choice $\ell = 1$ has been demonstrated to give better results than $\ell = 0$ in, e.g., [4,11]; recent computed examples in [6] show that it can be beneficial to let $\ell > 1$ in minimal residual methods for the solution of linear discrete ill-posed problems (1.1) with a nonsymmetric matrix A. We therefore, initially, will allow $\ell \geq 1$ in the present paper; however, the interesting case for linear discrete ill-posed problems with a symmetric matrix is $\ell = 1$.

Introduce the Krylov subspaces

$$\mathbb{K}_k(A, A^\ell b) = \text{span}\{A^\ell b, A^{\ell+1}b, \ldots, A^{\ell+k-1}b\}, \qquad k = 1,2,3,\ldots .$$

We are concerned with the minimal residual Krylov subspace method, whose kth iterate $x_k^{(\ell)}$ is characterized by

$$\|Ax_k^{(\ell)} - b\| = \min_{x \in \mathbb{K}_k(A,A^\ell b)} \|Ax - b\|, \quad x_k^{(\ell)} \in \mathbb{K}_k(A,A^\ell b). \tag{2.1}$$

The initial iterate is $x_0^{(\ell)} = 0$ and all iterates live in $\mathscr{R}(A^\ell)$. We refer to this method as MINRES(ℓ). The method is range restricted when $\ell \geq 1$. MINRES(0) is the standard MINRES method by Paige and Saunders [17]. Computed examples reported in [4] show that when the desired solution $\hat{x}$ is a discretization of a smooth function, MINRES(1) yields more accurate approximations of $\hat{x}$ than MINRES(0); see also Section 4 for an illustration. We therefore will not consider MINRES(0) in this paper. The following result is helpful to explore the structure of the matrices for reduced linear discrete ill-posed problems generated by the minimal residual Krylov subspace method of primary interest us.

Proposition 2.1 *Let $d\mu$ be a nontrivial probability measure supported on an infinite subset of points on the real line. Introduce the families of orthonormal polynomials $\{p_j\}_{j=0}^{\infty}$ and $\{\hat{p}_j\}_{j=0}^{\infty}$ such that*

$$\langle p_j, p_i \rangle := \int p_j(t) p_i(t) d\mu(t) = \begin{cases} 1, & j = i, \\ 0, & j \neq i, \end{cases} \tag{2.2}$$

$$\langle \hat{p}_j, \hat{p}_i \rangle_{t^\beta} := \int \hat{p}_j(t) \hat{p}_i(t) t^\beta d\mu(t) = \begin{cases} 1, & j = i, \\ 0, & j \neq i, \end{cases} \tag{2.3}$$

where $\beta \geq 1$ is a positive integer, and p_j and $\hat{p}_j$ are polynomials of degree j with positive leading coefficient. Generically, all polynomials $\hat{p}_j$ exist and

$$p_j(t) = \alpha_{j,j}\hat{p}_j(t) + \alpha_{j,j-1}\hat{p}_{j-1}(t) + \ldots + \alpha_{j,j-\beta}\hat{p}_{j-\beta}(t), \quad j = 0,1,2,\ldots, \tag{2.4}$$

where the $\hat{p}_j$s with $j < 0$ are defined to be the zero function and $\alpha_{j,i} = 0$ for $i < 0$. Moreover, $\alpha_{j,j} > 0$ for $j \geq 0$ as well as $\alpha_{j,j-\beta} > 0$ for $j \geq \beta$.

Proof When β is an even positive integer, (2.3) is an inner product and the orthonormal polynomials $\hat{p}_j$ of all degrees exist. This is also the case when $\beta > 0$ is odd and the support of the measure $d\mu$ lives on the positive real axis. If β is an odd positive integer and the support of $d\mu$ is not confined to the positive real axis, then (2.3) is a bilinear form and some orthonormal polynomials $\hat{p}_j$ might not exist. This may lead to a breakdown in the three-term recurrence relation for the $\hat{p}_j$; see, e.g., Brezinski et al. [2] for techniques to overcome breakdown. Generically all orthonormal polynomials $\hat{p}_j$ exist. We focus on this situation.

Let $p_j(t) = \sum_{i=0}^{j} \alpha_{j,i}\hat{p}_i(t)$ and note that the coefficient

$$\alpha_{j,i} = \langle p_j, \hat{p}_i \rangle_{t^\beta}$$

vanishes when the polynomial $\hat{p}_i(t)t^\beta$ is of degree less than j. The sign of the coefficients $\alpha_{j,j}$ and $\alpha_{j,j-\beta}$ follows from the fact that both p_j and $\hat{p}_j$ have positive leading coefficients. □

The orthonormal polynomials of Proposition 2.1 satisfy three-term recurrence relations. Let the entries of the symmetric (infinite) tridiagonal matrices T and $\hat{T}$ be the recurrence coefficients of the polynomials p_j and $\hat{p}_j$, respectively. If β is an even positive integer, then $\hat{T}$ can be determined from T by application of $\beta/2$ steps of the QR algorithm to T; see Kautsky and Golub [12] and Buhmann and Iserles [3] for discussions. When T is positive definite and $\beta > 0$ is an odd integer, then one step of the symmetric LR method is required to construct $\hat{T}$ from T (in addition to $(\beta-1)/2$ steps with the QR algorithm); see [12]. A recent discussion on how symmetric tridiagonal matrices whose entries are recursion coefficients for orthonormal polynomials are changed when the measure is modified by a polynomial or rational function is provided by Gautschi [7].

The remainder of this section discusses how the property of the polynomial families described by Proposition 2.1 predicts the structure of the matrices of the reduced linear discrete ill-posed problems generated by the MINRES(1) method during the iterations. Introduce the spectral factorization

$$A = U\Lambda U^T,$$

where $\Lambda = \mathrm{diag}[\lambda_1, \lambda_2, \ldots, \lambda_m] \in \mathbb{R}^{m\times m}$ and $U \in \mathbb{R}^{m\times m}$ is orthogonal. Define the unit vector $\breve{b} = [\breve{b}_1, \breve{b}_2, \ldots, \breve{b}_m]^T := U^T b$. Then

$$(f,g) := b^T f(A)g(A)b = \breve{b}^T f(\Lambda)g(\Lambda)\breve{b} = \sum_{j=1}^{m} f(\lambda_j)g(\lambda_j)\breve{b}_j^2 \tag{2.5}$$

is an inner product for all polynomials f and g of sufficiently small degree. Precisely, the sum of the degrees of f and g has to be strictly smaller than the number of distinct eigenvalues λ_j associated with positive weights $\breve{b}_j^2$. This requirement holds for linear discrete ill-posed problems (1.1) of interest.

Application of k steps of the symmetric Lanczos process to the matrix A with initial vector b yields the decomposition

$$AV_k = V_{k+1}T_{k+1,k}, \tag{2.6}$$

where the columns of the matrix $V_{k+1} = [v_1, v_2, \ldots, v_{k+1}] \in \mathbb{R}^{m\times(k+1)}$ form an orthonormal basis for the Krylov subspace $\mathscr{K}_{k+1}(A,b) = \mathrm{span}\{b, Ab, \ldots, A^k b\}$ with $v_1 = b$, the matrix $V_k \in \mathbb{R}^{m\times k}$ is made up of the first k columns of V_{k+1}, and the tridiagonal matrix $T_{k+1,k} \in \mathbb{R}^{(k+1)\times k}$ has a leading $k\times k$ symmetric tridiagonal submatrix T_k, positive subdiagonal entries, and a last row proportional to the vector e_k^T. Throughout this paper $e_j = [0,\ldots,0,1,0,\ldots,0]^T$ denotes the jth canonical basis vector of appropriate dimension. We refer to, e.g., Golub and Van Loan [8] or Saad [20] for details on the symmetric Lanczos process. Here we assume that the number of steps, k, is small enough so that the Lanczos decomposition (2.6) with the stated properties exists; otherwise the relation (2.6) simplifies to

$$AV_k = V_k T_k \tag{2.7}$$

for some k, where T_k is a symmetric tridiagonal matrix. If T_k is fairly well-conditioned, then we determine the exact solution of (1.1), given by $x = V_k T_k^{-1} e_1$. When T_k is very

ill-conditioned or singular, the small linear system of equations $T_k y = e_1\|b\|$ should be regularized before solution. We will not dwell on this further, because it is very unusual that the relation (2.7) holds for $k \ll m$.

Equation (2.6) is a recursion formula for the columns v_j of the matrix V_{k+1} and shows that, for $j = 1,2,\ldots,k+1$,

$$v_j = p_{j-1}(A)b \tag{2.8}$$

for some polynomial p_{j-1} of degree $j-1$. Therefore,

$$v_j^T v_i = b^T p_{j-1}(A)p_{i-1}(A)b = (p_{j-1}, p_{i-1}), \qquad 1 \le j,i \le k+1,$$

i.e., the polynomials p_j, $j = 0,1,\ldots,k$, are orthonormal with respect to the inner product (2.5), which is analogous to the inner product (2.2), but with only a discrete finite point set on the real axis as support.

Similarly, k steps of the symmetric Lanczos process applied to A with initial vector $A^\ell b/\|A^\ell b\|$ determines the decomposition

$$AV_k^{(\ell)} = V_{k+1}^{(\ell)} T_{k+1,k}^{(\ell)}, \tag{2.9}$$

where the columns of $V_{k+1}^{(\ell)} = [v_1^{(\ell)}, v_2^{(\ell)}, \ldots, v_{k+1}^{(\ell)}] \in \mathbb{R}^{m\times(k+1)}$ form an orthonormal basis for the Krylov subspace $\mathscr{K}_{k+1}(A, A^\ell b) = \text{span}\{A^\ell b, A^{\ell+1}b, \ldots, A^{\ell+k}b\}$. Moreover, $v_1^{(\ell)} = A^\ell b/\|A^\ell b\|$, the matrix $V_k^{(\ell)} \in \mathbb{R}^{m\times k}$ is made up of the first k columns of $V_{k+1}^{(\ell)}$, and $T_{k+1,k}^{(\ell)} \in \mathbb{R}^{(k+1)\times k}$ has a leading $k\times k$ symmetric tridiagonal submatrix, which we denote by $T_k^{(\ell)}$, and a last row proportional to the vector e_k^T. We assume that k is small enough so that the decomposition (2.9) with the stated properties exists.

Analogously to (2.8), we obtain from (2.9) that, for $j = 1,2,\ldots,k+1$,

$$v_j^{(\ell)} = \hat{p}_{j-1}(A)A^\ell b, \tag{2.10}$$

for some polynomial $\hat{p}_{j-1}$ of degree $j-1$. Therefore,

$$\begin{aligned}(v_j^{(\ell)})^T v_i^{(\ell)} &= b^T A^\ell \hat{p}_{j-1}(A)\hat{p}_{i-1}(A)A^\ell b = \tilde{b}^T \hat{p}_{j-1}(\Lambda)\hat{p}_{i-1}(\Lambda)\Lambda^{2\ell}\tilde{b} \\ &= \sum_{j=1}^{m} \hat{p}_{j-1}(\lambda_j)\hat{p}_{i-1}(\lambda_j)\lambda_j^{2\ell}\tilde{b}_j^2 =: (\hat{p}_{j-1}, \hat{p}_{i-1})_{A^{2\ell}}, \qquad 1 \le j,i \le k+1,\end{aligned}$$

i.e., the polynomials $\hat{p}_j$, $j = 0,1,\ldots,k$, are orthonormal with respect to an inner product analogous to (2.3) but with a finite discrete subset of the real axis as support.

The matrix $V_k^{(1)}$ in the Lanczos decomposition (2.9) with $\ell = 1$ can be computed from the decomposition (2.6) without evaluating additional matrix-vector products with A as follows. Introduce the QR factorization of the tridiagonal matrix $T_{k+1,k}$ in (2.6),

$$T_{k+1,k} = Q_{k+1}^{(1)} R_{k+1,k}^{(1)}, \tag{2.11}$$

where $Q_{k+1}^{(1)} \in \mathbb{R}^{(k+1)\times(k+1)}$ is orthogonal and $R_{k+1,k}^{(1)} \in \mathbb{R}^{(k+1)\times k}$ has a leading $k \times k$ upper triangular submatrix, $R_k^{(1)}$, and a vanishing last row. The matrix $Q_{k+1}^{(1)}$ can be expressed as a product of k Givens rotations,

$$Q_{k+1}^{(1)} = G_1 G_2 \cdots G_k, \tag{2.12}$$

where $G_j \in \mathbb{R}^{(k+1)\times(k+1)}$ is a rotation in the planes j and $j+1$. Thus, G_j is the identity matrix except for a 2×2 block in the rows and columns j and $j+1$. The representation (2.12) shows that $Q_{k+1}^{(1)}$ is upper Hessenberg. Moreover, the matrix $R_{k+1,k}^{(1)}$ is banded with right half-bandwidth 2. The QR factorization (2.11) can be computed in only $\mathcal{O}(k)$ arithmetic floating point operations (flops).

Let the matrix $Q_{k+1,k}^{(1)} \in \mathbb{R}^{(k+1)\times k}$ consist of the first k columns of $Q_{k+1}^{(1)}$ and introduce

$$W_k^{(1)} = V_{k+1} Q_{k+1,k}^{(1)}. \tag{2.13}$$

Since both the matrices V_{k+1} and $Q_{k+1,k}^{(1)}$ have orthonormal columns, so does $W_k^{(1)}$. We obtain from (2.6) and (2.11) that

$$AV_k = W_k^{(1)} R_k^{(1)}, \tag{2.14}$$

which shows that $\mathcal{R}(W_k^{(1)}) = \mathbb{K}_k(A, Ab)$.

Proposition 2.2 *Let the matrices $W_k^{(1)}$ and $V_k^{(1)}$ be defined by (2.13) and (2.9), respectively. Then*

$$W_k^{(1)} e_j = \pm V_k^{(1)} e_j, \qquad 1 \le j \le k,$$

where the sign may vary with j.

Proof The leading upper triangular submatrix $R_k^{(1)} = [r_{i,j}^{(1)}] \in \mathbb{R}^{k\times k}$ of $R_{k+1,k}^{(1)}$ in (2.11) has nonvanishing diagonal entries, because all subdiagonal entries of the tridiagonal matrix $T_{k+1,k}$ are positive. Identifying the first columns of the right-hand side and left-hand side of (2.14) shows that $W_k^{(1)} e_1 r_{1,1}^{(1)} = Av_1$. It follows that $W_k^{(1)} e_1 = \pm Av_1/\|Av_1\|$. Identifying the second columns of the right-hand side and left-hand side of (2.14) yields that $W_k^{(1)} e_2$ lives in $\mathcal{K}_2(A, Ab)$. Moreover, this vector is orthogonal to $Av_1/\|Av_1\|$. These properties are shared by the unit vector $v_2^{(1)}$, the second column of $V_k^{(1)}$. It follows that $W_k^{(1)} e_2 = \pm v_2^{(1)}$. We may proceed in this manner, column by column, to show the proposition. □

It follows from (2.14) that

$$V_k^T A W_k^{(1)} = L_k^{(1)}, \qquad L_k^{(1)} := (R_k^{(1)})^T. \tag{2.15}$$

Letting the polynomials p_{j-1} and $\hat{p}_{i-1}$ be defined by (2.8) and (2.10) with $\ell = 1$, respectively, yields the coefficients

$$\alpha_{j-1,i-1} = v_j^T A v_i^{(1)} \tag{2.16}$$

in the expansion (2.4). Fixing i, we obtain from Proposition 2.1 with $\beta = 2$ that

$$Av_i^{(1)} \in \text{span}\{v_i, v_{i+1}, v_{i+2}\}.$$

Proposition 2.2 and (2.15) now yield

$$AW_k^{(1)} = V_{k+2} L_{k+2,k}^{(1)}, \tag{2.17}$$

where $L_{k+2,k}^{(1)} \in \mathbb{R}^{(k+2)\times k}$ is a lower triangular matrix with left half-bandwidth 2. The matrix $L_k^{(1)}$ in (2.15) is the leading $k \times k$ submatrix of $L_{k+2,k}^{(1)}$. The nontrivial entries of the matrix are the nonvanishing coefficients (2.16). Specifically, we have

$$L_{k+2,k}^{(1)} = \begin{bmatrix} \alpha_{0,0} & & & & & & & \text{O} \\ \alpha_{1,0} & \alpha_{1,1} & & & & & & \\ \alpha_{2,0} & \alpha_{2,1} & \alpha_{2,2} & & & & & \\ & \alpha_{3,1} & \alpha_{3,2} & \alpha_{3,3} & & & & \\ & & \alpha_{4,2} & \alpha_{4,3} & \alpha_{4,4} & & & \\ & & & \ddots & \ddots & \ddots & & \\ & & & & \alpha_{k,k-2} & \alpha_{k,k-1} & \alpha_{k,k} & \\ & & & & & \alpha_{k+1,k-1} & \alpha_{k+1,k} & \\ \text{O} & & & & & & & \alpha_{k+2,k} \end{bmatrix}. \tag{2.18}$$

Our solution method for the least-squares problem (2.1) for $\ell = 1$ uses the relation (2.17). It differs from the solution method described in [15] in that it exploits both the lower triangularity and the small bandwidth of $L_{k+2,k}^{(1)}$.

We may proceed similarly to determine an analogue of the expression (2.17) with $W_k^{(1)}$ replaced by a matrix $W_k^{(2)} \in \mathbb{R}^{n\times k}$, whose columns form an orthonormal basis for $\mathscr{K}_k(A, A^2b)$, and the matrix $L_{k+2,k}^{(1)}$ replaced by a matrix with small bandwidth. This analogue of (2.17) would be useful for the implementation of the MINRES(2) method. However, we will not dwell on the details, because MINRES(2) generally does not provide better approximations of $\hat{x}$ than MINRES(1). This is illustrated in Section 4. We finally remark that it is, of course, possible to derive the decomposition (2.17) by linear algebra techniques only, i.e., without invoking Proposition 2.1. However, the proposition provides insight into why a decomposition of the form (2.17) can be expected to exist.

3 The MINRES(1) method

Substituting the decomposition (2.17) into (2.1) yields

$$\begin{aligned} \min_{y\in\mathbb{R}^k} \|AW_k^{(1)} y - b\| &= \min_{y\in\mathbb{R}^k} \|V_{k+2} L_{k+2,k}^{(1)} y - b\| \\ &= \min_{y\in\mathbb{R}^k} \|L_{k+2,k}^{(1)} y - e_1 \|b\|\ \|, \end{aligned} \tag{3.1}$$

where the last equality follows from the fact that $V_{k+2}e_1 = b/\|b\|$. We first provide an outline of a progressive scheme for the computation of a sequence of iterates $x_1^{(1)}, x_2^{(1)}, x_3^{(1)}, \ldots$. Subsequently, we present an algorithm with details of the computations. The initial iterate, $x_0^{(1)}$, is the zero vector.

Define the QR factorization

$$L_{k+2,k}^{(1)} = \widetilde{Q}_{k+2,k}\widetilde{R}_k, \tag{3.2}$$

where $\widetilde{Q}_{k+2,k} \in \mathbb{R}^{(k+2)\times k}$ has orthonormal columns and $\widetilde{R}_k \in \mathbb{R}^{k\times k}$ is upper triangular and banded with upper half-bandwidth 2. This factorization can be computed in only $\mathcal{O}(k)$ flops by applying a judiciously chosen sequence of elementary rotations. Substituting (3.2) into (3.1) shows that this least-squares problem can be solved in only $\mathcal{O}(k)$ flops provided that the matrix $L_{k+2,k}^{(1)}$ is available. The computations can be carried out "progressively", i.e., the iterate $x_k^{(1)}$ can be computed by updating the previous iterate $x_{k-1}^{(1)}$ in such a manner that only a few of the most recently generated columns of the matrices V_{k+2} and $W_k^{(1)}$ are required. Therefore, only a few m-vectors have to be stored at any time during the iterations. In particular, the storage requirement of MINRES(1) can be bounded independently of the number of iterations. This is similar as for the MINRES algorithm by Paige and Saunders [17] and follows from the fact that the matrix $L_{k+2,k}^{(1)}$ is banded. The derivation requires the use of suitable auxiliary vectors and is analogous to the derivation of the conjugate gradient method from the Lanczos method described by Saad [20, p. 188] but somewhat more complicated, because the matrix (2.18) has nontrivial subdiagonal and sub-subdiagonal entries, while the tridiagonal matrix determined by the Lanczos method only has nontrivial subdiagonal elements. We omit the details. Further comments on the storage requirement are provided below.

The entry (j,k) of a matrix M is denoted by $M_{j,k}$. We use MATLAB-inspired notation. Thus, $b_{i:j}$ denotes the subvector of the vector b with entries $b_i, b_{i+1}, \ldots, b_j$, where we assume that $i \leq j$. Similarly, $L_{i:j,h:k}$ denotes the submatrix of the matrix $L \in \mathbb{R}^{n\times n}$ made up of the entries $L_{s,t}$, $i \leq s \leq j$, $h \leq t \leq k$. Moreover, $L_{i:j,:}$ denotes the submatrix consisting of the entries $L_{s,t}$ with $i \leq s \leq j$ and $1 \leq t \leq n$. Let $\alpha, \beta, \gamma, \delta$ be real scalars. Then $[\alpha,\ \beta]$ denotes a row vector, $[\alpha;\ \beta]$ a column vector, and $[\alpha,\ \beta;\ \gamma,\ \delta]$ stands for the matrix

$$\begin{bmatrix} \alpha & \beta \\ \gamma & \delta \end{bmatrix}.$$

The symmetric tridiagonal matrix T generated by the Lanczos process has the diagonal entries $T_{j,j} = \alpha_j$ and subdiagonal entries $T_{j+1,j} = \beta_j$, $j = 1, 2, \ldots$. Thus, the $(k+1)$st column of T has the nontrivial entries (from top to bottom) $\beta_k, \alpha_{k+1}, \beta_{k+1}$. The matrix V_k has the columns $v_1, v_2, \ldots, v_k$. Finally, $0_{1:h,1:k}$ denotes the zero matrix of size $h \times k$ and $0_{1:k}$ stands for the zero column vector with k entries. The algorithm does not assume that $\|b\| = 1$.

Algorithm 3.1 *MINRES*(1)

Input: A, b, number of iterations s.
Output: Approximate solutions $x_1, x_2, \ldots, x_s$ of (1.1) and norm of the associated residual vectors $\sigma_k = \|b - Ax_k\|$, $k = 1, 2, \ldots, s$.
% Initialization and first step of the Lanczos method
$G := 0_{1:2s,1:4}$; $L := 0_{1:s+2,1:s+1}$; $R := 0_{1:s+2,1:s}$; $\tilde{b} = [\|b\|; 0_{1:s+2}]$;
$v_1 := b/\|b\|$; $\hat{v} := Av_1$;
$\alpha_1 := v_1^T \hat{v}$; $\hat{v} := \hat{v} - \alpha_1 v_1$;
$\beta_1 := \|\hat{v}\|$; $v_2 := \hat{v}/\beta_1$;
$T_{1,1} := \alpha_1$; $T_{2,1} = \beta_1$; $L_{1:2,1} := T_{1:2,1}$;
for $k = 1, 2, \ldots, s$ *do*
  % Compute new Lanczos vector v_{k+2} and column $k+1$ of T
  $\hat{v} = Av_{k+1} - \beta_k v_k$;
  $\alpha_{k+1} := v_{k+1}^T \hat{v}$; $\hat{v} := \hat{v} - \alpha_{k+1} v_{k+1}$;
  $\beta_{k+1} := \|\hat{v}\|$; $v_{k+2} := \hat{v}/\beta_{k+1}$;
  $L_{k:k+2,k+1} := T_{k:k+2,k+1}$;
  % Apply Givens rotation to columns k and $k+1$ of T to determine the
  % corresponding columns of L
  $\rho := (L_{k,k}^2 + L_{k,k+1}^2)^{1/2}$; $c := L_{k,k}^2/\rho$; $s := L_{k,k+1}^2/\rho$;
  $L_{k,k:k+1} = [cL_{k,k} + sL_{k,k+1}; 0]$; $L_{k+1:k+2,k:k+1} = L_{k+1:k+2,k:k+1}[c,\ -s;\ s,\ c]$;
  % Form vectors w_k and $\hat{w}_k$, where w_k is the kth column of the matrix $W_k^{(1)}$ in (2.14) and
  % $\hat{w}_k$ accumulates the Givens rotations of the previous columns
  if $k = 1$ *then*
    $w_1 := V_{:,1:2}[c;\ s]$; $\hat{w}_1 := V_{:,1:2}[-s;\ c]$;
  else
    $w_k := c\hat{w}_{k-1} + sV_{:,k+1}$; $\hat{w}_k := -s\hat{w}_{k-1} + cV_{:,k+1}$;
  end
  % Compute QR factorization of L
  % Update new column $L_{:,k}$ by previous Givens rotations
  if $k = 2$ *then*
    $L_{2:3,2} := [G_{1,1},\ G_{1,2};\ -G_{1,2},\ G_{1,1}]L_{2:3,2}$;
    $L_{1:2,2} := [G_{1,3},\ G_{1,4};\ -G_{1,4},\ G_{1,3}]L_{1:2,2}$;
  else
    if $k > 2$ *then*
      for $i = k-2 : k-1$ *do*
        $L_{i+1:i+2,k} := [G_{i,1},\ G_{i,2};\ -G_{i,2},\ G_{i,1}]L_{i+1:i+2,k}$;
        $L_{i:i+1,k} := [G_{i,3},\ G_{i,4};\ -G_{i,4},\ G_{i,3}]L_{i:i+1,k}$;
      end
    end
  end
  $R_{1:k,k} := L_{1:k,k}$;
  % Apply Givens rotations to rows $k+1$ and $k+2$ of L to determine the
  % corresponding rows of R
  $\mu := (L_{k+1,k}^2 + L_{k+2,k}^2)^{1/2}$; $c := L_{k+1,k}/\mu$; $s := L_{k+2,k}/\mu$; $G_{k,1:2} := [c,\ s]$;
  $R_{k+1,k} := cL_{k+1,k} + sL_{k+2,k}$;
  $\mu := (R_{k,k}^2 + R_{k+1,k}^2)^{1/2}$; $c := R_{k,k}/\mu$; $s := L_{k+1,k}/\mu$; $G_{k,3:4} := [c,\ s]$;
  $R_{k:k+1,k} := [cR_{k,k} + sR_{k+1,k};\ 0]$;
  % Update right-hand side $\tilde{b}$
  $\tilde{b}_{k:k+1} := [c,\ s;\ -s,\ c]\tilde{b}_{k:k+1}$;
  % Compute the approximate solutions x_1 or x_k when $k > 1$. The latter
  % approximate solution is computed by updating x_{k-1}
  if $k = 1$ *then*
    $z_1 := w_1/R_{1,1}$; $x_1 := \tilde{b}_k z_1$;
  elseif $k = 2$ *then*
    $z_k := (w_k - R_{k-1,k}z_{k-1})/R_{k,k}$;
  else
    $z_k := (w_k - R_{k-2,k}z_{k-2} - R_{k-1,k}z_{k-1})/R_{k,k}$;

```
        end
    end
    if k > 1 then
        x_k := x_{k-1} + b̃_k z_k;
    end
    σ_k := |b̃_{k+1}|;
end
```

Many of the vectors in the algorithm can be overwritten to save storage. For instance, all iterates x_k can share the same storage. A careful implementation of the above algorithm requires storage of at most nine m-vectors, only, independently of the number of iterations s. When the algorithm is applied with the discrepancy principle, defined in the following section, it is not known a priori how many iterations will be carried out. We then can choose a sufficiently large value of s when allocating storage in the initialization phase of the algorithm. For most problems of practical interest $s = 30$ suffices. This is illustrated by the computed examples of Section 4. Alternatively, one easily can modify the algorithm to allocate storage dynamically during execution, and in this manner avoid the necessity to allocate zero matrices before execution. Moreover, some of the zero matrices can be reduced or avoided all together by reusing storage.

4 Computed examples

This section presents a few numerical examples produced with Algorithm 3.1. We compare this algorithm with the MATLAB code sym_rrgmres_dp from [16]. This software is described in Section 3 of [15]. Comparisons of sym_rrgmres_dp to the Orthodir implementation used in [9] and to an implementation described in [4] are presented in [15], and show the software in [16] to often require fewer iterations to determine an approximation of $\hat{x}$ with desired accuracy than the other methods. The reason for the superior performance of the software sym_rrgmres_dp depends on that the method is less sensitive to round-off errors introduced during the computation than the other methods in the comparison. The computed examples in this section illustrate that Algorithm 3.1 performs as well as the software [16] and, therefore, is competitive with the methods described in [4,9]. Moreover, Algorithm 3.1 requires less computer storage and fewer arithmetic floating point operations than the implementation [16]. The storage demand of the implementations sym_rrgmres_dp and sym_rrgmres_iter from [16] of Algorithm 3.1 in [15] is proportional to sm, where s is the number of iterations carried out. The lower storage requirement of Algorithm 3.1[1] is its most attractive feature, but we point out that its flop count for carrying out s iterations is somewhat smaller than for the software sym_rrgmres_dp and sym_rrgmres_iter. The main purpose of the computed examples is to illustrate that the accuracy achieved and the number of iterations required by Algorithm 3.1 is about the same as with the software in [16]. All computations were carried out in MATLAB with about 15 significant decimal digits.

[1] Algorithm 3.1 refers to the algorithm in Section 3 unless explicitly stated otherwise.

implementation	ν	# iterations k	$\|x_k - \hat{x}\|/\|\hat{x}\|$
Algorithm 3.1	$1 \cdot 10^{-1}$	4	$1.67 \cdot 10^{-1}$
Software from [16]	$1 \cdot 10^{-1}$	4	$1.67 \cdot 10^{-1}$
Algorithm 3.1	$1 \cdot 10^{-2}$	5	$1.31 \cdot 10^{-1}$
Software from [16]	$1 \cdot 10^{-2}$	5	$1.31 \cdot 10^{-1}$
Algorithm 3.1	$1 \cdot 10^{-4}$	10	$3.67 \cdot 10^{-2}$
Software from [16]	$1 \cdot 10^{-4}$	10	$3.67 \cdot 10^{-2}$
Algorithm 3.1	$1 \cdot 10^{-6}$	15	$1.95 \cdot 10^{-2}$
Software from [16]	$1 \cdot 10^{-6}$	15	$1.95 \cdot 10^{-2}$
Algorithm 3.1	$1 \cdot 10^{-8}$	26	$7.16 \cdot 10^{-3}$
Software from [16]	$1 \cdot 10^{-8}$	26	$7.16 \cdot 10^{-3}$
Algorithm 3.1	$1 \cdot 10^{-10}$	38	$3.68 \cdot 10^{-3}$
Software from [16]	$1 \cdot 10^{-10}$	38	$3.68 \cdot 10^{-3}$

Table 4.1 Example 4.1: Number of iterations k and relative error in iterate x_k determined with the discrepancy principle for Algorithm 3.1 and the software sym_rrgmres_dp from [16] for different noise levels ν.

Example 4.1. Let the matrix A be obtained by discretizing the integral equation

$$\int_{-\pi/2}^{\pi/2} \kappa(\tau,\sigma)x(\sigma)d\sigma = b(\tau), \qquad -\frac{\pi}{2} \leq \tau \leq \frac{\pi}{2}, \tag{4.1}$$

where

$$\kappa(\sigma,\tau) = (\cos(\sigma)+\cos(\tau))\left(\frac{\sin(\xi)}{\xi}\right)^2, \qquad \xi = \pi(\sin(\sigma)+\sin(\tau)).$$

The right-hand side function $b(\tau)$ is chosen so that the solution $x(\sigma)$ is the sum of two Gaussian functions. This integral equation is discussed by Shaw [21]. We discretize it with the code shaw from [10], using a quadrature rule with 200 nodes. This yields a symmetric matrix $A \in \mathbb{R}^{200\times 200}$ and a scaled discretized solution $\hat{x} \in \mathbb{R}^{200}$ of (4.1), from which we determine $\hat{b} = A\hat{x}$. A "noise vector" $e \in \mathbb{R}^{200}$ with normally distributed random entries with zero mean, and scaled to correspond to a specified noise level

$$\nu = \frac{\|e\|}{\|\hat{x}\|}$$

is added to $\hat{b}$ to give the contaminated right-hand side in (1.1); cf. (1.2).

The iterations are terminated with the discrepancy principle, i.e., as soon as an iterate x_k has been determined that satisfies

$$\|Ax_k - b\| \leq \|e\|.$$

Table 4.1 shows the number of iterations required as well as the relative error in the computed approximate solution x_k of (1.1). The table reports results both for Algorithm 3.1 and for the method implemented by the software [16]. The number of iterations and the error in the computed approximate solutions x_k determined by these schemes is seen to be almost the same. □

implementation	ν	# iterations k	$\|x_k - \hat{x}\|/\|\hat{x}\|$
Algorithm 3.1	$1\cdot 10^{-2}$	4	$2.59\cdot 10^{-2}$
Software from [16]	$1\cdot 10^{-2}$	4	$2.59\cdot 10^{-2}$
Algorithm 3.1	$1\cdot 10^{-3}$	8	$1.16\cdot 10^{-2}$
Software from [16]	$1\cdot 10^{-3}$	8	$1.16\cdot 10^{-2}$
Algorithm 3.1	$1\cdot 10^{-4}$	11	$5.45\cdot 10^{-3}$
Software from [16]	$1\cdot 10^{-4}$	11	$5.45\cdot 10^{-3}$
Algorithm 3.1	$1\cdot 10^{-6}$	29	$7.65\cdot 10^{-4}$
Software from [16]	$1\cdot 10^{-6}$	29	$7.65\cdot 10^{-4}$
Algorithm 3.1	$1\cdot 10^{-8}$	95	$1.04\cdot 10^{-4}$
Software from [16]	$1\cdot 10^{-8}$	94	$1.07\cdot 10^{-4}$
Algorithm 3.1	$1\cdot 10^{-10}$	201	$3.85\cdot 10^{-5}$
Software from [16]	$1\cdot 10^{-10}$	200	$3.87\cdot 10^{-5}$

Table 4.2 Example 4.2: Number of iterations k and relative error in iterate x_k determined with the discrepancy principle for Algorithm 3.1 and the software sym_rrgmres_dp from [16] for different noise levels ν.

Example 4.2. Consider the Fredholm integral equation of the first kind

$$\int_{-6}^{6} \kappa(t,s)x(s)ds = b(t), \qquad -6 \le t \le 6, \tag{4.2}$$

discussed by Phillips [18]. Its solution, kernel, and right-hand side are given by

$$\begin{aligned} x(s) &= \begin{cases} 1+\cos(\frac{\pi}{3}s), & \text{if } |s|<3, \\ 0, & \text{otherwise}, \end{cases} \\ \kappa(t,s) &= x(t-s), \\ b(t) &= (6-|t|)(1+\frac{1}{2}\cos(\frac{\pi}{3}t)) + \frac{9}{2\pi}\sin(\frac{\pi}{3}|t|). \end{aligned}$$

We discretize this integral equation by a Galerkin method using orthonormal box functions. This discretization is computed with the function phillips from [10], which determines a symmetric matrix $A \in \mathbb{R}^{200\times 200}$ and a scaled discretization $\hat{x} \in \mathbb{R}^{200}$ of the solution of (4.2), with which we determine the error-free right-hand side vector $\hat{b} = A\hat{x}$. The contaminated right-hand side $b \in \mathbb{R}^{200}$ is defined analogously as in Example 4.1.

Table 4.2 displays the performance of Algorithms 3.1 and the software [16]. The iterations are terminated by the discrepancy principle. Algorithm 3.1 is seen to require about the same number of iterations as the implementation sym_rrgmres_dp from [16]. □

Example 4.3. We consider the restoration of a 300×300-pixel image that has been contaminated by blur and noise. The pixel values are stored column-wise in a vector $b \in \mathbb{R}^{90000}$. The blur is Gaussian and described by the matrix $A \in \mathbb{R}^{90000\times 90000}$, which is generated with the function blur from [10] using the parameters sigma$=1$ and band$=7$. Here sigma controls the width of the Gaussian point spread function and band specifies the bandwidth. The matrix A is a symmetric block Toeplitz matrix with Toeplitz blocks. Let the vector $e \in \mathbb{R}^{90000}$ represent the noise in b. We let e have normally distributed random entries with zero mean, and be scaled to correspond to

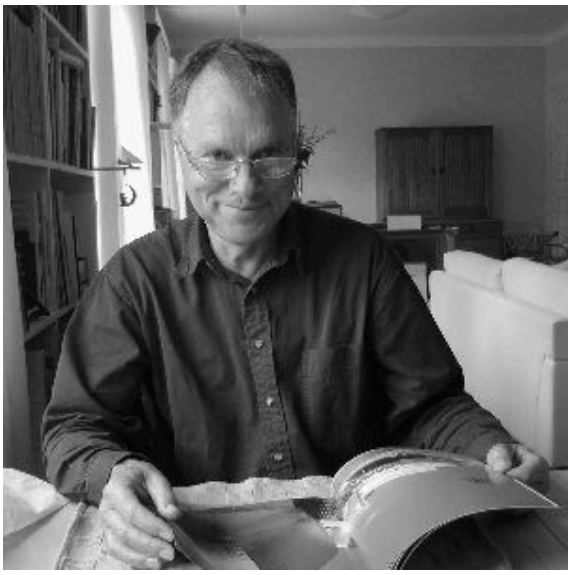

Fig. 4.1 Desired blur- and noise-free image “Axel”.

Fig. 4.2 Blur- and noise-contaminated image.

the noise level 0.1%. Finally, let $\hat{x} \in \mathbb{R}^{90000}$ represent the (unknown) desired image. Then b satisfies

$$b = A\hat{x} + e.$$

Figure 4.1 displays the desired blur and noise-free image represented by the vector $\hat{x}$ and Figure 4.2 shows the available blur- and noise-contaminated image represented by the vector b.

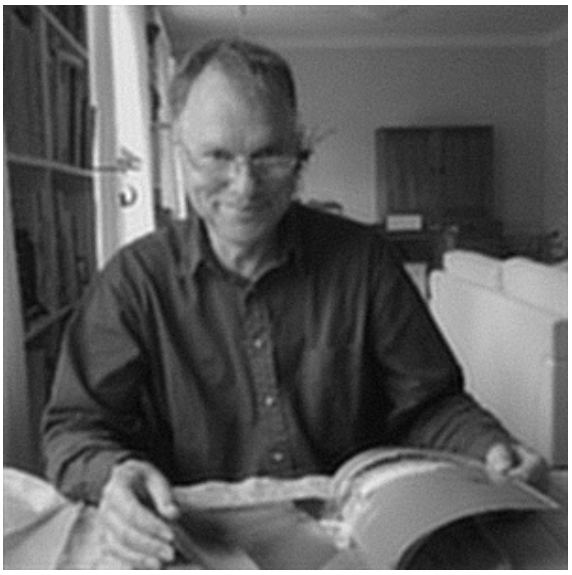

Fig. 4.3 Image restored by MINRES(1).

The iterative solution of (1.1) with A and b defined as described yields approximations x_k, $k = 1,2,3,\ldots$, of the desired image $\hat{x}$. The discrepancy principle yields the iterate x_{18} as our approximation of $\hat{x}$. Figure 4.3 shows the image represented by x_{18}.

iterative method	relative error in iterates
MINRES(0)	$\|x_{11}^{(0)} - \hat{x}\|/\|\hat{x}\| = 9.08 \cdot 10^{-2}$
MINRES(1)	$\|x_{18}^{(1)} - \hat{x}\|/\|\hat{x}\| = 8.54 \cdot 10^{-2}$
MINRES(2)	$\|x_{25}^{(2)} - \hat{x}\|/\|\hat{x}\| = 8.61 \cdot 10^{-2}$

Table 4.3 Relative errors in iterates determined with the discrepancy principle.

We conclude this example with an illustration of the MINRES(ℓ) iterative methods for $\ell \neq 1$; cf. (2.1). The number of iterations required by MINRES(ℓ) to produce an iterate that satisfies the discrepancy principle typically increases with ℓ. This is illustrated by Table 4.3. The MINRES(0) iterates determined by the discrepancy principle typically do not approximate $\hat{x}$ as well as the corresponding iterates computed with MINRES(1). This depends on that the iterates computed by MINRES(1) are orthogonal to $\mathcal{N}(A)$, while iterates determined by MINRES(0) generally are not.

Hanke [9] provides an analysis in a Hilbert space setting that shows that MINRES(1) is a regularization method in a well-defined sense, while MINRES(0) is not. The iterates generated by MINRES(ℓ) with $\ell > 1$ also are orthogonal to $\mathcal{N}(A)$, but Tables 4.3 and 4.4, as well as other computed examples, indicate that the quality of the computed approximate solutions, generally, is not improved by using MINRES(ℓ) with $\ell > 1$ instead of MINRES(1).

iterative method	relative error in iterates
MINRES(0)	$\|x_9^{(0)} - \hat{x}\|/\|\hat{x}\| = 8.88 \cdot 10^{-2}$
MINRES(1)	$\|x_{28}^{(1)} - \hat{x}\|/\|\hat{x}\| = 8.27 \cdot 10^{-2}$
MINRES(2)	$\|x_{40}^{(2)} - \hat{x}\|/\|\hat{x}\| = 8.30 \cdot 10^{-2}$

Table 4.4 Relative errors in iterates that best approximate $\hat{x}$.

The discrepancy principle is not guaranteed to determine the iterate that best approximates $\hat{x}$. Table 4.4 shows the relative errors in the best approximations of $\hat{x}$ determined by MINRES(ℓ) for $\ell \in \{0,1,2\}$. Again, MINRES(0) yields the least accurate approximation and MINRES(1) the best one. □

5 Conclusion

We discussed the structure of the matrices that arise in minimal residual methods for the iterative solution of linear discrete ill-posed problems with a symmetric matrix. The exploitation of the structure made it possible to present a new algorithm that requires less storage and achieves the same accuracy as the best available methods in the literature.

Acknowledgments

We would like to thank the referees for comments. The work of F.M. was supported by Dirección General de Investigación Científica y Técnica, Ministerio de Economía y Competitividad of Spain under grant MTM2012-36732-C03-01. Work of L.R. was supported by Universidad Carlos III de Madrid in the Department of Mathematics during the academic year 2010-2011 within the framework of the Chair of Excellence Program and by NSF grant DMS-1115385.

References

1. C. Brezinski, M. Redivo-Zaglia, G. Rodriguez, and S. Seatzu, Multi-parameter regularization techniques for ill-conditioned linear systems, Numer. Math., 94 (2003), pp. 203–228.
2. C. Brezinski, M. Redivo-Zaglia, and H. Sadok, New look-ahead Lanczos-type algorithms for linear systems, Numer. Math., 83 (1999), pp. 53–85.
3. M. D. Buhmann and A. Iserles, On orthogonal polynomials transformed by the QR algorithm, J. Comput. Appl. Math., 43 (1992), pp. 117–134.

4. D. Calvetti, B. Lewis, and L. Reichel, On the choice of subspace for iterative methods for linear discrete ill-posed problems, Int. J. Appl. Math. Comput. Sci., 11 (2001), pp. 1069–1092.
5. D. Calvetti, L. Reichel, and Q. Zhang, Conjugate gradient algorithms for symmetric inconsistent linear systems, in Proceedings of the Cornelius Lanczos International Centenary Conference, eds. J. D. Brown, M. T. Chu, D. C. Ellison, and R. J. Plemmons, SIAM, Philadelphia, 1994, pp. 267–272.
6. L. Dykes and L. Reichel, A family of range restricted iterative methods for linear discrete ill-posed problems, Dolomites Res. Notes Approx., 6 (2013), pp. 27–36.
7. W. Gautschi, Orthogonal Polynomials: Computation and Approximation, Oxford University Press, Oxford, 2004.
8. G. H. Golub and C. F. Van Loan, Matrix Computations, 4th ed., Johns Hopkins University Press, Baltimore, 2013.
9. M. Hanke, Conjugate Gradient Type Methods for Ill-Posed Problems, Longman, Harlow, 1995.
10. P. C. Hansen, Regularization tools version 4.0 for Matlab 7.3, Numer. Algorithms, 46 (2007), pp. 189–194.
11. P. C. Hansen and T. K. Jensen, Noise propagation in regularizing iterations for image deblurring, Electron. Trans. Numer. Anal., 31 (2008), pp. 204–220.
12. J. Kautsky and G. H. Golub, On the calculation of Jacobi matrices, Linear Algebra Appl., 52/53 (1983), pp. 439–455.
13. S. Kindermann, Convergence analysis of minimization-based noise levelfree parameter choice rules for linear ill-posed problems, Electron. Trans. Numer. Anal., 38 (2011), pp. 233–257.
14. S. Morigi, L. Reichel, F. Sgallari, and F. Zama, Iterative methods for ill-posed problems and semiconvergent sequences, J. Comput. Appl. Math., 193 (2006), pp. 157–167.
15. A. Neuman, L. Reichel, and H. Sadok, Implementations of range restricted iterative methods for linear discrete ill-posed problems, Linear Algebra Appl., 436 (2012), pp. 3974–3990.
16. A. Neuman, L. Reichel, and H. Sadok, Algorithms for range restricted iterative methods for linear discrete ill-posed problems, Numer. Algorithms, 59 (2012), pp. 325–331.
17. C. C. Paige and M. A. Saunders, Solution of sparse indefinite systems of linear equations, SIAM J. Numer. Anal., 12 (1975), pp. 617–629.
18. D. L. Phillips, A technique for the numerical solution of certain integral equations of the first kind, J. ACM, 9 (1962), pp. 84–97.
19. L. Reichel and G. Rodriguez, Old and new parameter choice rules for discrete ill-posed problems, Numer. Algorithms, 63 (2013), pp. 65–87.
20. Y. Saad, Iterative Methods for Sparse Linear Systems, 2nd ed., SIAM, Philadelphia, 2003.
21. C. B. Shaw, Jr., Improvements of the resolution of an instrument by numerical solution of an integral equation, J. Math. Anal. Appl., 37 (1972), pp. 83–112.