SYMMLQ-like procedure for Ax=b where A is a special normal matrix

H. Faßbender* and Kh. D. Ikramov†

June 15, 2005

Abstract: We propose a method of Lanczos type for solving a linear system with a normal matrix whose spectrum is contained in a second degree curve. This is a broader class of matrices than (l, m)-normal matrices introduced in a recent paper by Barth and Manteuffel. Our approach is similar to that by Huhtanen in the sense that both use the condensed form of normal matrices discovered by Elsner and Ikramov. However, there are a number of differences, among which are: (i) our method is modeled after the SYMMLQ algorithm of Paige and Saunders; (ii) it uses only one matrix-vector product per step; (iii) we provide effective means for monitoring the size of the residual during the process. Some numerical experiments are presented.

Keywords: generalized Krylov sequence, Rayleigh-Ritz projection, SYMMLQ algorithm, GMRES algorithm, normal matrix

1 Introduction

Consider the linear system

$$Ax = b. \tag{1}$$

In case A is large and sparse, Krylov subspace based methods should be used for solving system (1). Faber and Manteuffel [3] showed that there does not

*Institut *Computational Mathematics*, TU Braunschweig, D-38023 Braunschweig, email: h.fassbender@tu-bs.de

†corresponding author, Faculty of Computational Mathematics and Cybernetics, Moscow State University, 119899 Moscow, Russia, e-mail: ikramov@cmc.msk.su

exist a conjugate gradient method for solving (1) based on a k-term recursion with a low k and using the Krylov subspace $\mathcal{K}(A, b)$ unless A is normal. Even then, expect for a few anomalies, the eigenvalues of A must be contained in a line. If this is the case, then $k = 3$. Earlier, the same result was proved by Voevodin and Tyrtyshnikov [13, 12]. It is assumed in all of these papers that an orthogonal basis in the Krylov subspace is constructed via a single short recursion.

Gragg [4] demonstrated that if A is unitary, then an orthogonal basis in a Krylov subspace of A can be constructed using a pair of short recurrence formulas. Using Gragg's ideas, Jagels and Reichel [9] derived an efficient minimal residual algorithm for unitary and shifted unitary matrices. In [1], Barth and Manteuffel extend the results of [9] showing that a conjugate gradient method based on a short multiple recursion is possible for linear polynomials of Hermitian and unitary matrices and their low rank modifications. The Jagels–Reichel–Barth–Manteuffel approach is equivalent to using a single short recursion of the form

$$p_{j+1} = \sum_{i=j-t}^{j} \beta_{ij} A p_i - \sum_{i=j-s}^{j} \sigma_{ij} p_i$$

for various small s and t.

In [6], Huhtanen describes a Hermitian Lanczos method for normal matrices. Making use of the Toeplitz decomposition of A, i.e., $A = H + iK$, where

$$H = \frac{1}{2}(A + A^*), \quad K = \frac{1}{2i}(A - A^*),$$

he observes that instead of solving the GMRES minimization problem

$$\min_{p_{j-1} \in \Pi_{j-1}} ||A p_{j-1}(A) b - b||,$$

solving

$$\min_{p_{j-1} \in \Pi_{j-1}} ||A p_{j-1}(H) b - b|| = \min_{p_{j-1} \in \Pi_{j-1}} ||p_{j-1}(H) A b - b|| \tag{2}$$

is useful. Here, Π_j denotes the set of all polynomials $p(x) = \sum_{\ell=0}^{j} a_\ell x^\ell$ of degree at most j. The best approximation to b in (2) can be computed from the Krylov subspace $\mathrm{span}\{Ab, HAb, H^2Ab, \ldots, H^{j-1}Ab\}$ using the Hermitian Lanczos method. The drawback of Huhtanen's method is that each step requires two matrix-vector products, one with H and another one with A.

In [5], Huhtanen gives an optimal iterative method for normal matrices with minimal polyanalytic polynomials of low degree. Polyanalytic polynomials are functions of the form

$$p(z) = \sum_{0 \le j+\ell \le k} c_{j\ell} z^{\ell} \overline{z}^{j}, \qquad c_{jl} \in \mathbb{C}.$$

Polyanalytic polynomials of the form $z^j \overline{z}^\ell$ are called polyanalytic monomials and an order $>$ among them is set as follows. Let $z^{j_1}\overline{z}^{\ell_1}$ and $z^{j_2}\overline{z}^{\ell_2}$ be two polyanalytic monomials. If $j_1 + \ell_1 > j_2 + \ell_2$ or, if $j_1 + \ell_1 = j_2 + \ell_2$ and $j_1 > j_2$, then $z^{j_1}\overline{z}^{\ell_1} > z^{j_2}\overline{z}^{\ell_2}$. The minimal polyanalytic polynomial of a normal matrix $A \in \mathbb{C}^{n\times n}$ is then defined as the monic polyanalytic polynomial of least possible degree annihilating A, see [8] for a more detailed discussion. If the degree of the minimal polyanalytic polynomial is moderate, then linear systems with A can be iteratively solved with a short term recurrence. Huhtanen gives an algorithm which uses a 5-term-recurrence (and scalar products of the form $x^*A^*AA^*y$, the solution vector x and the residual vector r are built as $x = x + \alpha q$ and $r = r - \alpha Aq$, no stopping criteria discussed), and shows that the norm of the residual in his algorithm does not exceed the corresponding norm in GMRES. More work of Huhtanen on normal matrices can be found in [7, 5].

Here, we will explore the solution of linear systems with normal matrices whose eigenvalues are contained in a curve of second degree; that is, for the Toeplitz decomposition of a normal matrix $A = H + \imath K$, we have an additional constraint of the form

$$\alpha H^2 + 2\beta HK + \gamma K^2 + \delta H + \epsilon K + \mu I = 0$$

for $\alpha, \beta, \gamma, \delta, \epsilon, \mu \in \mathbb{R}$. Note that unitary matrices are the simplest example of normal matrices with the spectrum on a second degree curve (the unit circle, in this case).

Our approach is similar to that of Huhtanen in the sense that both use the condensed form for normal matrices introduced by Elsner and Ikramov in [2]. The main differences are as follows:

1. Our method belongs to the orthogonal residual (or Galerkin) class of methods and is modeled after the SYMMLQ algorithm of Paige and Saunders (see [10]).

2. With the exception of the third step, we use only matrix-vector products of type A^*q, one product at a step.

3. We provide effective means for monitoring the size of the residual. The residuals are kept orthogonal to each other.

The paper is organized as follows. In Section 2, we describe our Lanczos-type algorithm. In Section 3, we discuss the SYMMLQ-like approach for solving the projected system. In Section 4, we present the results of numerical experiments.

2 A Lanczos-type method for projection

The system to be solved is

$$Ax = b, \tag{3}$$

where

$$AA^* = A^*A \tag{4}$$

and for $c \neq 0$

$$cA^2 + \bar{c}A^{*2} + 2dAA^* + 2eA + 2fA^* + gI = 0. \tag{5}$$

In [2], Elsner and Ikramov discuss condensed forms of normal matrices under finite sequences of elementary unitary similarity transformations. The condensed form which can be achieved for a normal matrix A with (5) is given in Figure 1. One of the approaches for reducing A to this form is a geometrical Lanczos-type one where generalized Krylov sequences are used. For the convenience of the reader, this approach is recalled here. The generalized Krylov sequence generated by A and b is

$$\underbrace{b,}_{\text{0th layer}} \quad \underbrace{A^*b, Ab}_{\text{1st layer}}, \underbrace{A^{*2}b, A^*Ab, AA^*b, A^2b}_{\text{2nd layer}}, \underbrace{A^{*3}b, \ldots,}_{\text{3rd layer}} \tag{6}$$

Generally, layer $k+1$ of this sequence is obtained by applying first A^* and then A to all the vectors in layer k. To construct the generalized Lanczos vectors, only linearly independent vectors in sequence (6) are of interest. In view of (4) and (5), this means that the vectors AA^*b and A^2b in the second layer must be skipped. Thus, only the vectors

$$A^{*3}b, A^{*2}Ab, AA^{*2}b, AA^*Ab \tag{7}$$

in the third layer need to be considered. Now, since A and A^* commute, $A^{*2}Ab$ and $AA^{*2}b$ are equal. Multiplying (5) by A^*, we observe that $A^*A^2b =$

AA^*Ab is a linear combination of $A^{*3}b, A^{*2}Ab$ and certain vectors in the first and second layers. Thus, only the first two vectors in (7) should be retained. This is the general rule: for a matrix A satisfying (4) and (5), layer k will consist of only two vectors, $A^{*k}b$ and $(A^*)^{k-1}Ab, k = 2, 3, \ldots$.

Denote by $\mathcal{L}_k$ the subspace spanned by the vectors in layers 0 to k. As usual, the first Lanczos vector is given by

$$q_1 = \frac{b}{||b||}.$$

The vector q_2 is obtained by orthogonalizing A^*q_1 to q_1 and normalizing the resulting vector. This means that

$$q_2 = \alpha_2 A^* q_1 + \beta_2 q_1.$$

In other words,

$$A^*q_1 \in \text{span}\{q_1, q_2\}. \tag{8}$$

The vector q_3 is obtained by orthogonalizing Aq_1 to q_1 and q_2 with subsequent normalization. Thus,

$$q_3 = \alpha_3 A q_1 + \beta_3 q_2 + \gamma_3 q_1$$

or

$$Aq_1 \in \text{span}\{q_1, q_2, q_3\}. \tag{9}$$

To obtain q_4, orthogonalize A^*q_2 to q_1, q_2, and q_3 and then normalize. Similarly, q_5 is obtained by orthogonalizing A^*q_3 to $q_1, \ldots, q_4$ and normalizing. This says that

$$q_4 = \alpha_4 A^* q_2 + \beta_4 q_3 + \gamma_4 q_2 + \delta_4 q_1$$

and

$$q_5 = \alpha_5 A^* q_3 + \beta_5 q_4 + \gamma_5 q_3 + \delta_5 q_2 + \varepsilon_5 q_1,$$

or

$$A^*q_2 \in \text{span}\{q_1, \ldots, q_4\} \tag{10}$$

and

$$A^*q_3 \in \text{span}\{q_1, \ldots, q_5\}. \tag{11}$$

Note that

$$q_4 \perp \mathcal{L}_1, q_4 \in \mathcal{L}_2,$$

and

$$q_5 \perp \mathcal{L}_1, q_5 \in \mathcal{L}_2.$$

It follows that

$$A^* q_4 \perp q_1, A^* q_5 \perp q_1.$$

Indeed,

$$(A^* q_i, q_1) = (q_i, A q_1) = 0, \qquad i = 4, 5,$$

since $Aq_1 \in \mathcal{L}_1$. Thus, q_6 and q_7 are constructed as linear combinations

$$\begin{aligned} q_6 &= \alpha_6 A^* q_4 + \beta_6 q_5 + \gamma_6 q_4 + \delta_6 q_3 + \varepsilon_6 q_2, \\ q_7 &= \alpha_7 A^* q_5 + \beta_7 q_6 + \gamma_7 q_5 + \delta_7 q_4 + \varepsilon_7 q_3 + \zeta_7 q_2. \end{aligned}$$

In other words,

$$A^* q_4 \in \text{span}\{q_2, \ldots, q_6\} \tag{12}$$

and

$$A^* q_5 \in \text{span}\{q_2, \ldots, q_7\}. \tag{13}$$

In (13), the maximum width of a relation in our generalized Lanczos process is attained. Indeed, for $m = 4, 5, \ldots$, we have

$$\begin{aligned} q_{2m} &= \alpha_{2m} A^* q_{2(m-1)} + \beta_{2m} q_{2m-1} + \gamma_{2m} q_{2(m-1)} + \delta_{2m} q_{2m-3} + \varepsilon_{2m} q_{2(m-2)}, \\ q_{2m+1} &= \alpha_{2m+1} A^* q_{2m-1} + \beta_{2m+1} q_{2m} + \gamma_{2m+1} q_{2m-1} + \delta_{2m+1} q_{2(m-1)} \\ &\quad + \varepsilon_{2m+1} q_{2m-3} + \zeta_{2m+1} q_{2(m-2)}. \end{aligned}$$

Relations (8) – (13) and the similar relations for larger m imply that, written in the basis $q_1, q_2, \ldots$, the matrix A has the structure shown in Figure 1. We observe that A has upper bandwidth 2.

We combine the relations above in a matrix equality. To this end, define

$$\begin{aligned} Q_k &= (q_1 \ \ldots \ q_k), \\ T_k &= Q_k^* A Q_k, \\ P_k &= (q_{k+1} \quad q_{k+2}). \end{aligned}$$

Then, $Q_k^* Q_k = I_k = (e_1 \ e_2 \ \ldots \ e_k)$ and the relations above can be written as

$$A Q_{2m+1} = Q_{2m+1} T_{2m+1} + P_{2m+1} E_{2m+1}, \qquad m = 1, 2, \ldots \tag{14}$$

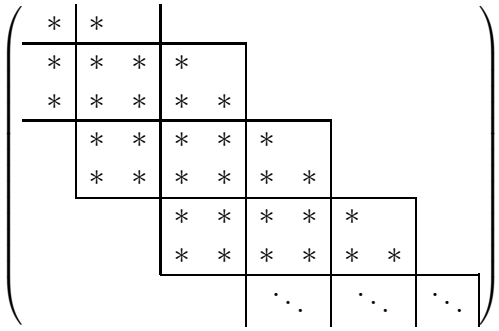

Figure 1: Structure of A written in the basis $q_1, q_2, q_3, \ldots$

where T_{2m+1} is the leading $(2m+1) \times (2m+1)$ principal submatrix in the matrix on Figure 1 and E_{2m+1} is the $2 \times (2m+1)$ matrix of the form $E_{2m+1} = (O \quad \hat{E}_{2m+1})$ with

$$\hat{E}_{2m+1} = \begin{pmatrix} t_{2m+2,2m} & t_{2m+2,2m+1} \\ t_{2m+3,2m} & t_{2m+3,2m+1} \end{pmatrix}.$$

Observe that although A is a normal matrix, T_k is not necessarily one:

$$\begin{aligned} T_k T_k^* &= Q_k^* A Q_k Q_k^* A^* Q_k, \\ T_k^* T_k &= Q_k^* A^* Q_k Q_k^* A Q_k. \end{aligned}$$

In general, these matrices are different since $Q_k Q_k^* \neq I$.

In constructing the matrices Q_{2m+1} and T_{2m+1}, one should keep in mind the conventional recipe for building up the matrix of a linear operator in a given basis; i.e.,

$$t_{ij} = (Aq_j, q_i) = (\overline{A^* q_i, q_j}).$$

Moreover, the matrix Q_{2m+1} is constructed columnwise.

The Rayleigh-Ritz projection of system (3) on the generalized Krylov subspace $\mathcal{L}_m$ is

$$T_{2m+1} y_{2m+1} = \beta e_1, \qquad \beta = ||b||. \tag{15}$$

Assuming that T_{2m+1} is nonsingular, it determines a unique vector $y_{2m+1} = (\eta_1^{(2m+1)}, \ldots, \eta_{2m+1}^{(2m+1)})$. Then, the vector

$$x_{2m+1}^c = Q_{2m+1} y_{2m+1} \tag{16}$$

can be considered as an approximate solution to (3). For the corresponding residual, we have

$$\begin{aligned} r^c_{2m+1} &= b - Ax^c_{2m+1} \\ &= \beta q_1 - AQ_{2m+1}y_{2m+1} \\ &= \beta q_1 - Q_{2m+1}T_{2m+1}y_{2m+1} - P_{2m+1}E_{2m+1}y_{2m+1} \\ &= -(q_{2m+2} \quad q_{2m+3}) \begin{pmatrix} t_{2m+2,2m}\ \eta^{(2m+1)}_{2m} + t_{2m+2,2m+1}\ \eta^{(2m+1)}_{2m+1} \\ t_{2m+3,2m}\ \eta^{(2m+1)}_{2m} + t_{2m+3,2m+1}\ \eta^{(2m+1)}_{2m+1} \end{pmatrix} \\ &\equiv -(q_{2m+2} \quad q_{2m+3}) \begin{pmatrix} \tau^{(2m+1)}_1 \\ \tau^{(2m+1)}_2 \end{pmatrix}. \end{aligned} \tag{17}$$

Thus, the 2-norm of r^c_{2m+1} is equal to

$$[\ |\tau^{(2m+1)}_1|^2 + |\tau^{(2m+1)}_2|^2\ ]^{1/2}. \tag{18}$$

Hence, in order to compute the 2-norm of the residual, one has to solve a system of type (15) provided that T_{2m+1} is a nonsingular matrix. If T_{2m+1} is singular, then the Rayleigh–Ritz approximation x^c_{2m+1} may not exist.

In the next section, we show that the sequence $\{x^c_{2m+1}\}$ can be replaced by a closely related sequence $\{x^L_{2m+1}\}$, whose members always exist. We also indicate an efficient way for calculating the 2-norms of the corresponding residuals r^L_{2m+1}.

3 SYMMLQ-like approach for solving the projected system

To introduce the vectors x^L_{2m+1}, we mimic Paige and Saunder's approach in deriving the algorithm SYMMLQ [10]. Define T as the matrix A written in the generalized Lanczos basis $q_1, \ldots, q_n$. Then all the T_k are the leading principal submatrices in T. Let

$$T = LZ \tag{19}$$

be the triangular-unitary factorization of T with L lower triangular and Z unitary. L is computed by applying a sequence of elementary unitary trans-

formations to T. We first use the complex rotation R_1 that eliminates t_{12},

$$TR_1 = \begin{pmatrix} * & 0 & & & & & \\ * & * & * & * & & & \\ * & * & * & * & * & & \\ * & * & * & * & * & * & \\ * & * & * & * & * & * & * \\ & & & * & * & * & * & * \\ & & & * & * & * & * & * \\ & & & & \ddots & \ddots & \ddots \end{pmatrix}.$$

The first row of TR_1 is already in desired form. Hence, the first row of L can be read off: $e_1^T L = e_1^T TR_1$. Then the Householder transformation H_2 that annihilates the entries (2,3) and (2,4) is used,

$$TR_1H_2 = \begin{pmatrix} * & & & & & & \\ * & * & 0 & 0 & & & \\ * & * & * & * & * & & \\ * & * & * & * & * & * & \\ * & * & * & * & * & * & * \\ & * & * & * & * & * & * \\ & * & * & * & * & * & * \\ & & & & \ddots & \ddots & \ddots \end{pmatrix}.$$

The second row of TR_1H_2 is already in desired form. Hence, $e_2^T L = e_2^T TR_1H_2$. Next the entries (3,4) and (3,5) are annihilated by the Householder transformation H_3:

$$TR_1H_2H_3 = \begin{pmatrix} * & & & & & & \\ * & * & & & & & \\ * & * & * & 0 & 0 & & \\ * & * & * & * & * & * & \\ * & * & * & * & * & * & \\ & * & * & * & * & * & * \\ & * & * & * & * & * & * \\ & & & & \ddots & \ddots & \ddots \end{pmatrix}.$$

The third row of $TR_1H_2H_3$ is in desired form. Hence, $e_3^T L = e_3^T TR_1H_2H_3$. As a result, L takes the form shown in Figure 2.

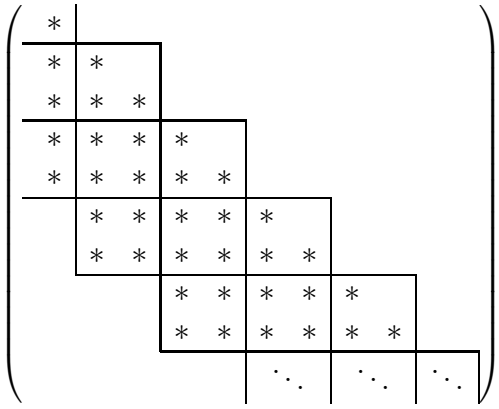

Figure 2: Form of L in $T = LZ$

Now, instead of finding y such that

$$Ty = \beta e_1,$$

we will look for z such that

$$Lz = \beta e_1. \tag{20}$$

If z is found, the solution x to the original system (3) can be computed as

$$x = QZ^* z.$$

The product QZ^* can be obtained by successively postmultiplying Q by the elementary matrices whose product constitutes Z^*.

In contrast to y, the vector z can be computed consecutively, one or two components at a time. Also, the Lanczos vectors q_i can be updated and then discarded, when they are not needed anymore, if x is accumulated.

The overall process proceeds as follows:

1. Find the vectors q_2, q_3, q_4, q_5, q_6, and q_7. This fully determines the first two columns in T.

2. Perform the rotation R_1 that eliminates entry $(1, 2)$ in T and modifies its first two columns.

3. Let χ_1 be entry $(1, 1)$ in the current T. Define

$$\zeta_1 = \beta / \chi_1.$$

   ζ_1 is the first component of z.

4. Apply the rotation R_1 to
$$(q_1 \quad q_2)$$
to obtain
$$(w_1 \quad \hat{w}_2) = (q_1 \quad q_2)R_1.$$

5. Define
$$x_1^L = \zeta_1 w_1.$$
After this step is completed, the vector w_1 can be discarded.

6. Find the vectors q_8 and q_9. This fully determines the first five columns in T.

7. Perform the Householder transformation H_2 that eliminates entries $(2,3)$ and $(2,4)$ in T and modifies its columns 2 to 4.

8. Let χ_2 be entry $(2,2)$ in the current T. In the latter, the first two rows are the rows of L. Define
$$\zeta_2 = -\frac{l_{21}\zeta_1}{\chi_2}.$$
This is the second component of z.

9. Apply H_2 to the matrix
$$(\hat{w}_2 \quad q_3 \quad q_4)$$
to obtain
$$(w_2 \quad \hat{w}_3 \quad \hat{w}_4) = (\hat{w}_2 \quad q_3 \quad q_4)H_2.$$

10. Define
$$x_2^L = x_1^L + \zeta_2 w_2.$$

11. Perform the Householder transformation H_3 that eliminates entries $(3,4)$ and $(3,5)$ and modifies columns 3 to 5 in T.

12. In the current T, the three upper rows are the corresponding rows of L. If χ_3 is entry $(3,3)$, then
$$\zeta_3 = -\frac{l_{31}\zeta_1 + l_{32}\zeta_2}{\chi_3}$$
is the third component of z.

13. Apply H_3 to the matrix

$$(\hat{w}_3 \quad \hat{w}_4 \quad q_5)$$

to obtain

$$(w_3 \quad \tilde{w}_4 \quad \hat{w}_5) = (\hat{w}_3 \quad \hat{w}_4 \quad q_5)H_3.$$

14. Define

$$x_3^L = x_2^L + \zeta_3 w_3.$$

The vectors w_2 and w_3 can now be discarded. In fact, if only the sequence $\{x_i^L\}$ is monitored, then w_2 could be discarded already after Step 10.

15. Find the vectors q_{10} and q_{11}, which fully determines columns 6 and 7 in T, and so on.

This procedure (suitably modified for the last factorization step) will compute the desired factorization $T = LZ$ (19). The resulting vector x^L corresponds in exact arithmetic to the exact solution of $Ax = b$. With

$$z = (\zeta_1, \zeta_2, \zeta_3, \ldots)^T$$

and

$$W = (w_1 \; w_2 \; w_3 \; \ldots) = QR_1H_2H_3 \cdots = QZ^*,$$

we have $W^*W = I$,

$$x = x^L = Wz = \sum_i \zeta_i w_i$$

and with $W_k = (w_1 \; w_2 \; \ldots \; w_k)$ and $z_j = (\zeta_1, \zeta_2, \ldots, \zeta_j)^T$

$$x_j^L = W_j z_j = \sum_{i=1}^{j} \zeta_i w_i.$$

Of course, we would rather compute an approximate solution x_j^c (see (16)) than the vector x_j^L with obscure approximation properties. The problem is that, as opposed to the vectors x_{2m+1}^c, the vectors x_{2m+1}^L exist for all m. However, when x_{2m+1}^c does exist and is a good approximation of x, how can we compute it from the current vector x_{2m+1}^L?

In the following we will answer the more general question: What is the connection between the two sequences of vectors $x_j^L = W_j z_j$ and $x_j^c = Q_j y_j$ (assuming the existence of the vectors)? Let

$$T_k = \widetilde{L}_k \widetilde{Z}_k$$

be the triangular-unitary factorization of the matrix $T_k = Q_k^* A Q_k$. Define

$$\widetilde{z}_k = \widetilde{Z}_k y_k. \tag{21}$$

Then

$$x_k^c = Q_k \widetilde{Z}_k^* \widetilde{z}_k.$$

Observe that all the components in $\widetilde{z}_k$, with the exception of the last two components, are the corresponding numbers ζ_i. This is because $\widetilde{z}_k$ is the solution of the system

$$\widetilde{L}_k \widetilde{z}_k = \beta e_1 \tag{22}$$

and $\widetilde{L}_k$, with the exception of the last two rows, is a leading principal submatrix of L:

$$\widetilde{L}_k(1:k-2, 1:k-2) = L(1:k-2, 1:k-2).$$

Also, $\widetilde{Z}_k^*$ is the product of the same rotation and the same Householder transformations that were used to find $\zeta_1, \ldots, \zeta_{k-2}$ and one additional rotation $\widetilde{R}_k$ designed to eliminate the entry $(k-1, k)$ in T_k. This says that

$$Q_k \widetilde{Z}_k^* = \widetilde{W}_{k-2} \widetilde{R}_k,$$

where

$$\widetilde{W}_{k-2} = (w_1 \ \ldots \ w_{k-2} \ \ \tilde{w}_{k-1} \ \ \hat{w}_k), \qquad \widetilde{W}_{k-2}^* \widetilde{W}_{k-2} = I$$

is the updated version of Q_k corresponding to the step when x_{k-2}^L was computed (as in the algorithm described at the beginning of this section). Thus, to find x_k^c from the known x_{k-2}^L , do the following:

1. Find the rotation $\widetilde{R}_k$ that eliminates entry $(k-1, k)$ in T_k. This determines the two bottom rows in $\widetilde{L}_k$.

2. Find the last two components $\widetilde{\zeta}_{k-1}$ and $\widetilde{\zeta}_k$ in the vector $\widetilde{z}_k$, using the last two equations in (22).

3. Find the vector x_k^c from

$$x_k^c = x_{k-2}^L + (\tilde{w}_{k-1} \quad \hat{w}_k)\widetilde{R}_k \begin{pmatrix} \widetilde{\zeta}_{k-1} \\ \widetilde{\zeta}_k \end{pmatrix}. \tag{23}$$

The second summand on the right-hand side is, obviously, a linear combination of $\tilde{w}_{k-1}$ and $\hat{w}_k$.

In theory, the proposed algorithm will stop with some $\hat{E}_{2m+1} = 0$, but in practice it is rare to have even a very small $\hat{E}_{2m+1}$. Hence some other stopping criterion must be used. In the algorithm discussed, x_{2m+1}^c and x_{2m+1}^L will be different; x_{2m+1}^c is usually a better approximation to the exact solution than x_{2m+1}^L. But x_{2m+1}^c may not exist (as T_{2m+1} may be singular), while x_{2m+1}^L always exists.

Moreover, x_k^L approximates x in the space spanned by $\{w_1, w_2, \ldots, w_k\}$, as

$$x_k^L = \sum_{i=1}^{k} \zeta_i w_i = W_k z_k$$

where $W_k = [w_1 \; w_2 \; \ldots \; w_k]$ and $z_k = [\zeta_1, \ldots, \zeta_k]^T$ as before. It follows from $W_k^* W_k = I$ that

$$||x_{k+1}^L||_2^2 = \sum_{i=1}^{k+1} |\zeta_i|^2 ||w_i||_2^2 = \sum_{i=1}^{k+1} |\zeta_i|^2 = ||x_k^L||_2^2 + |\zeta_{k+1}|^2.$$

Hence, the x_k^L's increase monotonically in the 2-norm. Furthermore, as $x = Wz$, we have $w_{k+1}^* x = e_{k+1}^T z = \zeta_{k+1}$ and

$$||x - x_{k+1}^L||_2^2 \;\; = \;\; ||x - x_k^L||_2^2 - |\zeta_{k+1}|^2.$$

Thus we have a monotone convergence for $x - x_k^L$:

$$||x - x_{k+1}^L||_2 \leq ||x - x_k^L||_2.$$

The stopping criterion of the proposed algorithm will be based on the residuals

$$\begin{aligned} r_{2m+1}^c &= b - Ax_{2m+1}^c, \\ r_{2m+1}^L &= b - Ax_{2m+1}^L. \end{aligned}$$

To use expressions (17) -(18), giving $||r^c_{2m+1}||$, we need to know the last two components, $\eta^{(2m+1)}_{2m}$ and $\eta^{(2m+1)}_{2m+1}$, of the vector y_{2m+1}. From (21), we know that

$$y_{2m+1} = \widetilde{Z}^*_{2m+1}\widetilde{z}_{2m+1}.$$

Also, we know that $\widetilde{Z}^*_{2m+1}$ is the product of rotations and Householder transformations, of which only the last rotation and the last two Householder transformations affect the last two components in y_{2m+1}. Thus, $\eta^{(2m+1)}_{2m}$ and $\eta^{(2m+1)}_{2m+1}$ can be found without computing the entire y_{2m+1} (but computation of $\widetilde{z}_{2m+1}$ is necessary). This completes the efficient computation of $||r^c_{2m+1}||$.

Next, we describe a way for computing $||r^L_{2m+1}||$. We have

$$\begin{aligned} r^L_{2m+1} &= b - Ax^L_{2m+1} \\ &= \beta q_1 - QTQ^*x^L_{2m+1} \\ &= \beta Qe_1 - QTQ^*x^L_{2m+1} \\ &= Q(\beta e_1 - TQ^*x^L_{2m+1}) \end{aligned} \tag{24}$$

Define the n-dimensional vector

$$z_{2m+1} = (\zeta_1, \zeta_2, \ldots, \zeta_{2m+1}, 0, \ldots, 0)^T,$$

where ζ_i are the components of z computed in the main process. Then, the formulas for updating the vectors x^L_i justify the equality

$$x^L_{2m+1} = QZ^*_{2m+1}z_{2m+1}, \tag{25}$$

where the $n \times n$ matrix Z^*_{2m+1} is the product of the rotation R_1 and the Householder transformations used in computing $\zeta_1, \ldots, \zeta_{2m+1}$. Substituting (25) in the right-hand side of (24) yields

$$r^L_{2m+1} = Q(\beta e_1 - TZ^*_{2m+1}z_{2m+1}). \tag{26}$$

Observe that, in view of (19) and the way by which Z is formed, the upper $2m+1$ rows in TZ^*_{2m+1} are the corresponding rows of L. Now, $\zeta_1, \ldots, \zeta_{2m+1}$ are defined in precisely such a way as to make the first $2m+1$ equations in (20) be fulfilled. Thus, the first $2m+1$ components of the vector inside the parentheses in (26) are zero.

Inspecting now the columns of TZ^*_{2m+1}, we observe that the first $2m+1$ columns contain only zeros in the group of rows beginning from row $2m+6$.

Since only the first $2m+1$ components of z_{2m+1} are nonzero, this says that only components $2m+2, 2m+3, 2m+4, 2m+5$ of the vector inside the parentheses in (26) can be nonzero. These components can be calculated as follows:

$$\begin{aligned}
\varphi_1 &= -(l_{2m+2,\,2m-2}\zeta_{2m-2} + l_{2m+2,\,2m-1}\zeta_{2m-1} \\
&\qquad + l_{2m+2,\,2m}\zeta_{2m} + l_{2m+2,\,2m+1}\zeta_{2m+1}) \\
\varphi_2 &= -(l_{2m+3,\,2m-2}\zeta_{2m-2} + l_{2m+3,\,2m-1}\zeta_{2m-1} \\
&\qquad + l_{2m+3,\,2m}\zeta_{2m} + l_{2m+3,\,2m+1}\zeta_{2m+1}) \\
\varphi_3 &= -(l_{2m+4,\,2m}\zeta_{2m} + l_{2m+4,\,2m+1}\zeta_{2m+1}) \\
\varphi_4 &= -(l_{2m+5,\,2m}\zeta_{2m} + l_{2m+5,\,2m+1}\zeta_{2m+1})
\end{aligned}$$

Equality (26) implies that

$$||r^L_{2m+1}||^2 = |\varphi_1|^2 + |\varphi_2|^2 + |\varphi_3|^2 + |\varphi_4|^2,$$

which is the required formula. Note another implication of (26): residual (26) is a linear combination of the vectors

$$q_{2m+2}, \quad q_{2m+3}, \quad q_{2m+4}, \quad q_{2m+5}.$$

4 Numerical experiments

The proposed algorithm has been programmed in MATLAB and tested on various systems of equations in order to obtain an impression of its numerical behavior. The code was run on an Intel Pentium 4 PC using MATLAB 7.0.4. Our implementation uses a modified Gram–Schmidt version of the Lanczos algorithm described in Section 2 without any further reorthogonalization. The projected system is solved as explained in Section 3. The examples are constructed to illustrate the convergence behavior for two matrices with eigenvalues on two different types of second degree curves. Therefore the chosen problems are somewhat artificial. In both examples the matrix $A \in \mathbb{C}^{n\times n}$ is normal with eigenvalues on a second degree curve. The right hand side b of the equation $Ax = b$ is a random complex vector. The computations were stopped as soon as the norm of the residual $||Ax^L_j - b||$ was less than 10^{-8}.

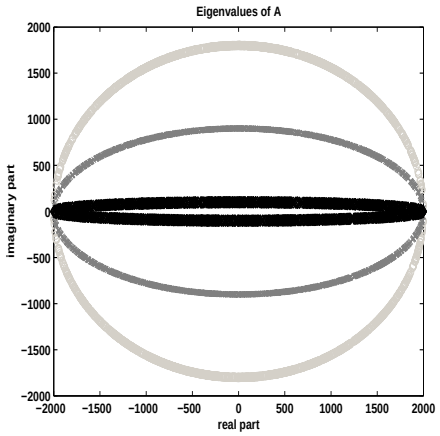

Figure 3: Eigenvalues of A_1, A_2, A_3 of Example 1, denoted by light-grey 'o', grey 'x', black '$\diamond$', respectively

Example 1: For the examples presented here, diagonal normal matrices with eigenvalues on the ellipse

$$\frac{x^2}{\alpha^2} + \frac{y^2}{\beta^2} = 1 \tag{27}$$

for real $x, y \in \mathbb{R}$, that is,

$$(\alpha^{-2} - \beta^{-2})(z^2 + \overline{z}^2) + 2(\alpha^{-2} + \beta^{-2})z\overline{z} - 4 = 0$$

for complex $z = x + \imath y \in \mathbb{C}$ were constructed for $\alpha = 2000$ and different real values of β. Depending on the choice of β the ellipse described by (27) is varying between a circle ($\beta = \alpha$) and a straight line ($\beta = 0$).

The tests reported were done with matrices A_1, A_2 and A_3 of size 2000, where for A_1 β was chosen to be 1800, for A_2 $\beta = 900$ and for A_3 $\beta = 100$. The eigenvalues of these matrices are displayed in Figure 3. As all matrices have the same size, the smaller β is chosen, the closer the eigenvalues move together. The condition number $\kappa_2(A) = \frac{\lambda_{max}}{\lambda_{min}}$ (where $\lambda_{max}/\lambda_{min}$ is the eigenvalue of largest/smallest absolute value) of the test matrices were $\kappa(A_1) \approx 1.1$, $\kappa(A_2) \approx 2.2$, and $\kappa(A_3) \approx 20$.

In Figure 4, we compare the convergence behavior of our method for the different test matrices in terms of the relative error $\frac{||x - x_j^L||}{||x||}$ between the exact

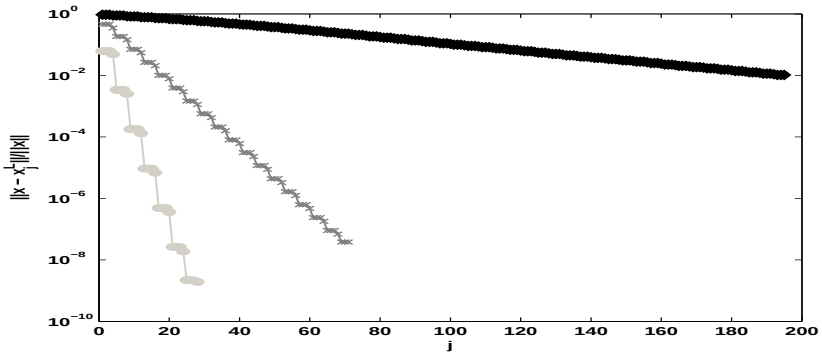

Figure 4: Norm of the relative error $\frac{||x-x_j^L||}{||x||}$ for A_1, A_2, A_3 of Example 1, denoted by light-grey 'o', grey 'x', black '$\diamond$', respectively

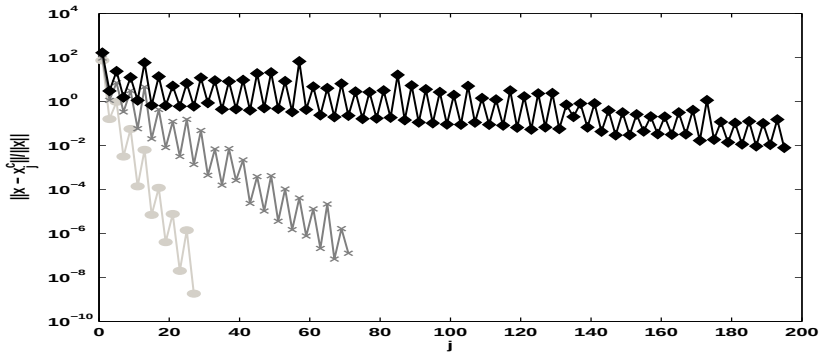

Figure 5: Norm of the relative error $\frac{||x-x_j^c||}{||x||}$ for A_1, A_2, A_3 of Example 1, denoted by light-grey 'o', grey 'x', black '$\diamond$', respectively

solution x and the approximate solution x_j^L for our test matrices. As shown in Section 3, $||x - x_j^L||$ decreases monotonically. Clearly, the convergence of our algorithm slows down when β is chosen smaller. In order to keep the convergence plot readable, we cut off the convergence curve of the third example. Experiments with other normal matrices whose eigenvalues lie on an ellipse showed similar convergence behavior. For test purposes the vectors x_{2j+1}^c have been computed from x_{2j+1}^L. As can be seen in Figure 5, the error for the x^c's converge, but they do not converge monotonically.

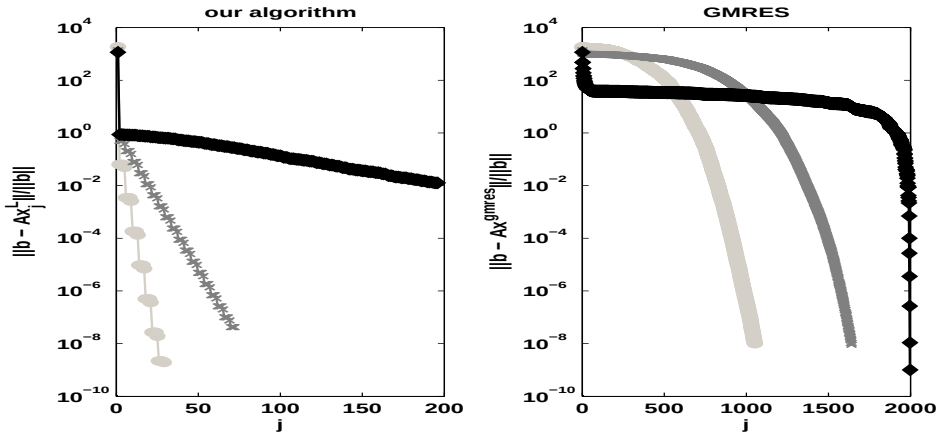

Figure 6: our algorithm versus GMRES for A_1, A_2, A_3 of Example 1, denoted by light-grey 'o', grey 'x', black '$\diamond$', respectively

Next, we compared the behavior of our algorithm to the famous GMRES algorithm [11]. In order to do so, the corresponding MATLAB routine 'gmres' was called. The problems under consideration are difficult problems for GMRES. Figure 6 shows the residuals $||b - Ax||$ for $x = x_j^L$ on the left hand side and for x_j^{gmres} on the right hand side. GMRES needed around 1050 (1600) iteration steps to reduce the residual to $\approx 10^{-8}$ for our test matrix A_1 (A_2), while our algorithm needed only about 30 (70) steps. For the example A_3 GMRES needs almost 1970 steps to reduce the norm of the residual to 10^{-2}, while our algorithm achieves this after about 200 steps. Note that, for both algorithms, the iterate x_j solves a system of size $j \times j$. Hence a comparison of the iterates x_j^L and x_j^{gmres} is appropriate. Clearly, in general, one step of our algorithm is cheaper than one step of GMRES.

Placing eigenvalues only on parts of the ellipse influences the convergence behavior. In the following test, we generated 2000×2000 matrices $\widetilde{A}_1, \widetilde{A}_2, \widetilde{A}_3$ as above, but such that all eigenvalues have positive real parts (see Figure 7). Figure 8 display the convergence behavior of our method as well as that of GMRES. Both methods are significantly faster due to the more favorable eigenvalue distribution.

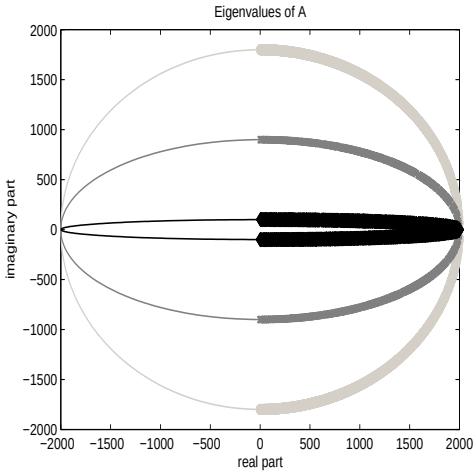

Figure 7: Eigenvalues of $\widetilde{A}_1, \widetilde{A}_2, \widetilde{A}_3$ of Example 1, denoted by light-grey 'o', grey 'x', black '$\diamond$', respectively

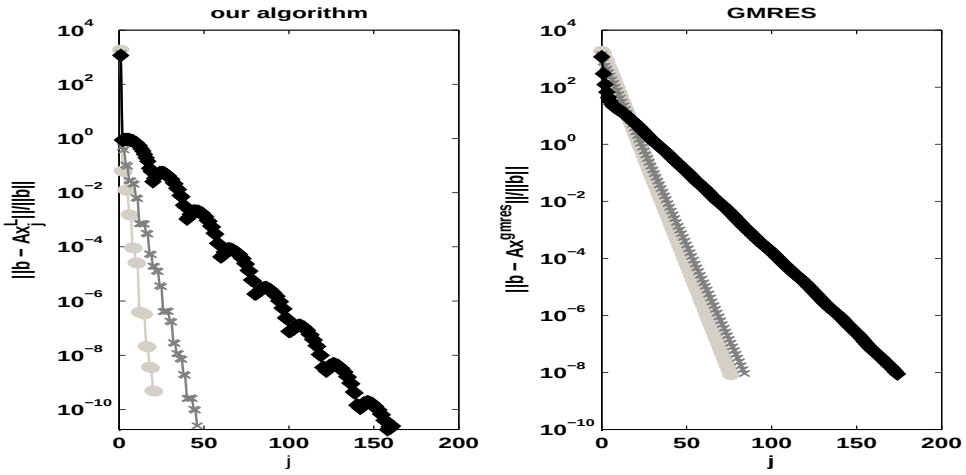

Figure 8: our algorithm versus GMRES for $\widetilde{A}_1, \widetilde{A}_2, \widetilde{A}_3$ of Example 1, denoted by light-grey 'o', grey 'x', black '$\diamond$', respectively

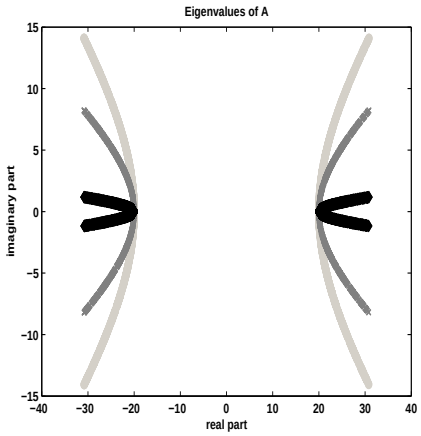

Figure 9: Eigenvalues of A_1, A_2, A_3 of Example 2, denoted by light-grey 'o', grey 'x', black '$\diamond$', respectively

Example 2: For this example diagonal normal matrices with eigenvalues on the hyperbola

$$\frac{x^2}{\alpha^2} - \frac{y^2}{\beta^2} = 1 \tag{28}$$

for real $x, y \in \mathbb{R}$, that is,

$$(\alpha^{-2} + \beta^{-2})(z^2 + \overline{z}^2) + 2(\alpha^{-2} - \beta^{-2})z\overline{z} - 4 = 0$$

for complex $z = x + iy \in \mathbb{C}$ were generated for $\alpha = 20$ and different real values of β. Depending on the choice of β the hyperbola described by (28) is varying.

The tests reported were done with matrices A_1, A_2 and A_3 of size 2000 where for A_1 β was chosen to be 12, for A_2 $\beta = 7$ and for A_3 $\beta = 1$. The eigenvalues of these matrices are displayed in Figure 9. As all matrices have the same size, the smaller β is chosen, the closer the eigenvalues move together. We choose the eigenvalues on the given hyperbola such that their real parts lie in the intervals $[-31, -20]$ and $[20, 31]$. The condition number $\kappa_2(A) = \frac{\lambda_{max}}{\lambda_{min}}$ of the test matrices were $\kappa(A_1) \approx 1.7$, $\kappa(A_2) \approx 1.6$, and $\kappa(A_3) \approx 1.5$. Since a hyperbola is an unbounded curve, we could, in principle, obtain any prescribed condition number for the matrix A by moving the eigenvalue farther away from the foci. This is not the case for a single ellipse.

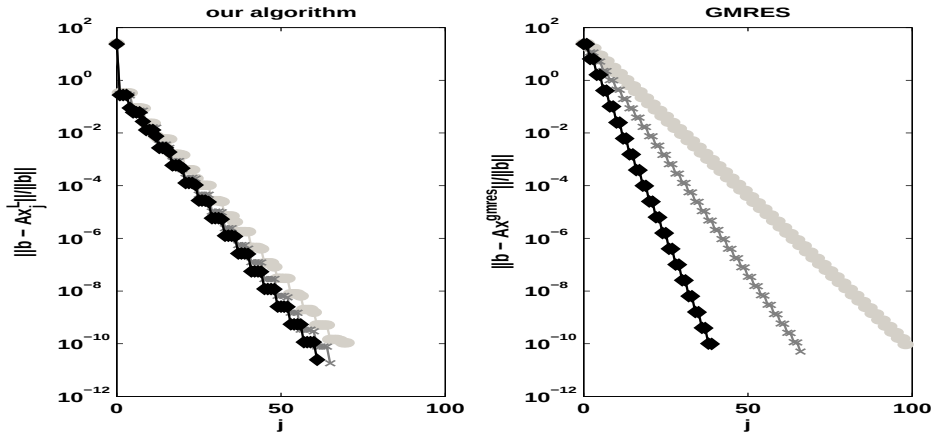

Figure 10: our algorithm versus GMRES for $\widetilde{A}_1, \widetilde{A}_2, \widetilde{A}_3$ of Example 2, denoted by light-grey 'o', grey 'x', black '$\diamond$', respectively

For all three test matrices, our algorithm converged quite fast, the amount of iteration steps needed did not vary much. It took 70 iteration steps to reduce the relative error $\frac{||x-x_j^L||}{||x|}$ to 10^{-10} for the test matrix A_1, 65 steps for A_2, and 60 steps for A_3. In Figure 10 the convergence of our method is compared to that of GMRES. For the matrix A_3 GMRES is slightly faster than our method, while for the other test matrices GMRES needed more iteration steps.

¿From our numerical tests we can conclude that our algorithm converges extremely fast for examples whose eigenvalues lie on a hyperbola or circle-like curves and that the convergence rate deteriorates if the eigenvalues lie on flatter ellipses. GMRES usually needs more iteration steps as the eigenvalue distribution in most of these examples is not well suited for fast GMRES convergence.

5 Acknowledgments

This work was done when the second author has visited the first one at the Technical Universities of Munich and Braunschweig and in-between these two visits. Both visits were supported by the Deutsche Forschungsgemeinschaft, to which organization Kh. D. Ikramov wishes to express his deep gratitude.

We are indebted to Valeria Simoncini and Diana O'Leary for their penetrating observation that our Krylov sequence with a slightly different order of vectors and another partition, namely,

$$b, Ab \mid A^*b, A^*Ab \mid A^{*2}b, A^{*2}Ab \mid \ \ldots,$$

would correspond to the block Lanczos method for the matrix A^*. What is especially interesting in this observation is that, usually, Lanczos methods are used only for Hermitian (and real symmetric) matrices, while our matrices are not Hermitian.

Finally, we want to thank Anatolii Zykov, who assisted us in programming the algorithm.

References

[1] T. Barth and T.A. Manteuffel. Multiple recursion conjugate gradient algorithms Part I: sufficient conditions. *SIAM J. Matrix Anal. Appl.*, 21:768–796, 2000.

[2] L. Elsner and Kh.D. Ikramov. On a condensed form for normal matrices under finite sequences of elementary unitary similarities. *Linear Algebra Appl.*, 254:79–98, 1997.

[3] V. Faber and T.A. Manteuffel. Necessary and sufficient conditions for the existence of a conjugate gradient method. *SIAM J. Numer. Anal.*, 21:352–362, 1984.

[4] W.B. Gragg. Positive definite Toeplitz matrices, the Arnoldi process for isometric operators, and Gaussian quadrature on the unit circle. *J. Comput. Appl. Math.*, 46:183–198, 1993.

[5] M. Huhtanen. Combining normality with the FFT techniques. *Preprint*, 2002.

[6] M. Huhtanen. A Hermitian Lanczos method for normal matrices. *SIAM J. Matrix Anal. Appl.*, 23:1092–1108, 2002.

[7] M. Huhtanen. Orthogonal polyanalytic polynomials and normal matrices. *Math. Comp.*, 72:355–373, 2002.

[8] M. Huhtanen and R.M. Larsen. Exclusion and inclusion regions for the eigenvalues of a normal matrix. *SIAM J. Matrix Anal. Appl.*, 23:1070–1091, 2002.

[9] C.F. Jagels and L. Reichel. A fast minimal residual algorithm for shifted unitary matrices. *Numer. Linear Algebra Appl.*, 1:555–570, 1994.

[10] C.C. Paige and M.A. Saunders. Solution of sparse indefinite systems of linear equations. *SIAM J. Numer. Anal.*, 12:617–629, 1975.

[11] Y. Saad and M. H. Schultz. GMRES: a generalized minimal residual algorithm for solving nonsymmetric linear systems. *SIAM J. Sci. Stat. Comput.*, 7:856–869, 1986.

[12] V.V. Voevodin. The question of non-self-adjoint extension of the conjugate gradient method is closed. *U.S.S.R. Comput. Maths. Phys.*, 23:143–144, 1983.

[13] V.V. Voevodin and E.E. Tyrtyshnikov. On generalization of conjugate direction methods. *Numerical Methods of Algebra (Chislennye Metody Algebry), (a collection of papers)*, pages 3–9, 1981.