

**The Lanczos-Ritz values appearing in  
an orthogonal similarity reduction of a  
matrix into semiseparable form**

*M. Van Barel    R. Vandebril    N. Mastronardi*

*Report TW 360, May 2003*



**Katholieke Universiteit Leuven**  
Department of Computer Science  
Celestijnenlaan 200A – B-3001 Heverlee (Belgium)

# The Lanczos-Ritz values appearing in an orthogonal similarity reduction of a matrix into semiseparable form

*M. Van Barel      R. Vandebril      N. Mastronardi*

*Report TW 360, May 2003*

Department of Computer Science, K.U.Leuven

## Abstract

It is well known how any symmetric matrix can be reduced by an orthogonal similarity transformation into tridiagonal form. Once the tridiagonal matrix has been computed, several algorithms can be used to compute either the whole spectrum or part of it. In this paper, we propose an algorithm to reduce any symmetric matrix into a similar semiseparable one of semiseparability rank 1, by orthogonal similarity transformations.

It turns out that partial execution of this algorithm computes a semiseparable matrix whose eigenvalues are the Ritz-values obtained by the Lanczos' process applied to the original matrix. Moreover, it is shown that at the same time a type of nested subspace iteration is performed. These properties allow to design different algorithms to compute the whole or part of the spectrum.

Numerical experiments illustrate the properties of the new algorithm.

**Keywords :** Similarity transformation, Semiseparable matrix, Lanczos' algorithm, Ritz-values.

**AMS(MOS) Classification :** Primary : 15A18, Secondary : 15A21, 65F15.

# THE LANCZOS-RITZ VALUES APPEARING IN AN ORTHOGONAL SIMILARITY REDUCTION OF A MATRIX INTO SEMISEPARABLE FORM

M. VAN BAREL<sup>†§</sup>, R. VANDEBRIL<sup>†§</sup>, AND N. MASTRONARDI<sup>‡¶</sup>

## Abstract.

It is well known how any symmetric matrix can be reduced by an orthogonal similarity transformation into tridiagonal form. Once the tridiagonal matrix has been computed, several algorithms can be used to compute either the whole spectrum or part of it. In this paper, we propose an algorithm to reduce any symmetric matrix into a similar semiseparable one of semiseparability rank 1, by orthogonal similarity transformations.

It turns out that partial execution of this algorithm computes a semiseparable matrix whose eigenvalues are the Ritz-values obtained by the Lanczos' process applied to the original matrix. Moreover, it is shown that at the same time a type of nested subspace iteration is performed. These properties allow to design different algorithms to compute the whole or part of the spectrum.

Numerical experiments illustrate the properties of the new algorithm.

**AMS subject classifications.** 15A18, 15A21, 65F15

**Key words.** Similarity transformation, Semiseparable matrix, Lanczos' algorithm, Ritz-values.

**1. Introduction.** The standard procedure to compute the complete eigendecomposition of a dense symmetric matrix first reduces it into a similar symmetric tridiagonal one by means of orthogonal transformations. This step is accomplished in  $\frac{4}{3}n^3 + O(n^2)$  flops<sup>1</sup> and it is the most expensive one. Once the similar symmetric tridiagonal matrix has been computed, many fast and stable algorithms can be considered to compute its spectral decomposition (see, for instance, [9, 17, 16] and the references therein).

Recently, several fast and stable algorithms that compute the eigendecomposition of symmetric diagonal plus semiseparable matrices have been developed [2, 3, 6, 15].

In this paper we consider an algorithm that transforms symmetric matrices into similar symmetric semiseparable ones by means of orthogonal transformations with  $\frac{4}{3}n^3 + O(n^2)$  flops. Hence, combining the latter algorithm with one of those for computing the eigendecomposition of semiseparable matrices gives us an alternative way to compute the eigenvalue decomposition of symmetric matrices, at the same computational complexity of the standard approach mentioned before.

Moreover, the proposed algorithm, while running to completion, gives information on the spectrum of the similar initial matrix. In fact the algorithm computes semiseparable matrices of increasing dimension whose eigenvalues are the Ritz values obtained by applying the Lanczos' method to the initial matrix. Hence, the algorithm can be used to approximate

---

<sup>†</sup>Department of Computer Science, Katholieke Universiteit Leuven, Celestijnenlaan 200A, B-3001 Leuven (Heverlee), Belgium. email: {Raf.Vandebriel, Marc.VanBarel}@cs.kuleuven.ac.be

<sup>‡</sup>Istituto per le Applicazioni del Calcolo "M. Picone", sez. Bari, Consiglio Nazionale delle Ricerche, Via G. Amendola, 122/I, I-70126 Bari, Italy. email: irmanm21@area.ba.cnr.it

<sup>§</sup>The research of the first two authors was supported by the Research Council K.U.Leuven, project OT/00/16 (SLAP: Structured Linear Algebra Package), by the Fund for Scientific Research–Flanders (Belgium), projects G.0078.01 (SMA: Structured Matrices and their Applications), G.0176.02 (ANCILA: Asymptotic aNalysis of the Convergence behavior of Iterative methods in numerical Linear Algebra), and G.0184.02 (CORFU: Constructive study of Orthogonal Rational Functions), by the K.U.Leuven (Bijzonder Onderzoeksfonds), and by the Belgian Programme on Interuniversity Poles of Attraction, initiated by the Belgian State, Prime Minister's Office for Science, Technology and Culture, project IUAP V-22 (Dynamical Systems and Control: Computation, Identification & Modelling). The scientific responsibility rests with the authors.

<sup>¶</sup>The research of the third author was partially supported by MIUR, grant number 2002014121.

<sup>1</sup>In this paper the definition given in [9] is adopted: "A flop is a floating point operation." The square root is considered a flop, too.

the “extreme eigenvalues” of the given matrix and the convergence theory of the Lanczos’ algorithm can be applied to the proposed algorithm. Moreover, we will also show that the proposed algorithm can be considered as a kind of subspace–like iteration method, where the size of the subspace increases by one dimension at each step of the algorithm. Hence, when the gaps between the eigenvalues involved are large enough, and when the Ritz values approximate the dominant eigenvalues enough, (diagonal blocks containing) the largest eigenvalues are already computed with high precision. Therefore it turns out that the proposed algorithm is a good candidate for computing an approximation of the subspace associated to the largest eigenvalues of large dense matrices. Such problems arise in many different areas, such as, principal component analysis, data mining, magnetic resonance spectroscopy, microarray data analysis, gene expression data, computing the eigensolutions of the Schrödinger equation on a grid ... (see, for instance, [1, 12, 21, 22, 25, 26] and the references therein).

The paper is organized as follows. In § 2 the definition and possible representations of semiseparable matrices are discussed. The algorithm for reducing symmetric matrices into similar symmetric semiseparable ones is described in § 3. The unsymmetric case is shortly considered in § 4. In § 5 the computational complexity is discussed. It also contains an algorithm for reducing symmetric tridiagonal matrices into similar symmetric semiseparable ones. The strong connection with the Lanczos-Ritz values is investigated in § 6. Convergence properties of the subspace iteration are studied in § 7. The numerical experiments are shown in § 8 followed by the conclusions and future work.

**2. Definition and representation of semiseparable matrices.** In the literature the class of semiseparable matrices is defined in slightly different ways. For an overview of different definitions, representations as well as their properties, we refer the interested reader to [20]. In this paper, semiseparable matrices are defined in the following way.

**DEFINITION 1.** *A matrix  $S$  is called a lower- (upper-)semiseparable matrix of semiseparability rank  $r$  if all submatrices which can be taken out of the lower (upper) triangular part of the matrix  $S$  have rank  $\leq r$  and there exists at least one submatrix having exact rank  $r$ .*

In the remainder of the paper, we will only consider symmetric semiseparable matrices of (lower- and upper-)semiseparability rank 1.

In [20], we have shown that under weak assumptions (see Theorem 2.1), the matrix  $S$  can be represented by two (column-)vectors  $u$  and  $v$ , called generators of the matrix  $S$ :

$$S = \text{triu}(uv^T) + \text{tril}(vu^T)$$

where  $\text{triu}(A)$  and  $\text{tril}(A)$  denote respectively the strictly upper triangular part and the lower triangular part of the matrix  $A$ .

**THEOREM 2.1.** *Suppose  $S$  is a symmetric semiseparable matrix of size  $n$ , then we have the following equivalence:*

$$S \text{ can be represented by two generators } u \text{ and } v \quad \Updownarrow \quad (2.1)$$

*For each  $1 \leq j \leq i \leq n$  such that  $S(i, j) = 0 \Rightarrow S(i, 1 : i) = 0$  or  $S(j : n, j) = 0$ .*

*Proof.* Can be found in [20]. ■

Because the representation with the generators is cheap in terms of memory usage and it is easy to reconstruct arbitrary elements within the matrix, several algorithms for semiseparable matrices are based on this representation. However for several of these algorithms, this representation can lead to a big loss of accuracy when performing the computations in finite precision. This can be seen from the following example.

EXAMPLE 1. Suppose we have a symmetric  $5 \times 5$  matrix with the following eigenvalues:  $(1, 2, 3, 100, 10^5)$ . Constructing a semiseparable matrix from it (the algorithm how to do so, will be explained later on) generates the following matrix:

$$\begin{pmatrix} 1.2738 & -5.7004 \cdot 10^{-1} & 1.2664 \cdot 10^{-1} & -1.6459 \cdot 10^{-4} & 1.5753 \cdot 10^{-12} \\ -5.7004 \cdot 10^{-1} & 2.2236 & -4.9398 \cdot 10^{-1} & 6.4202 \cdot 10^{-4} & -1.5858 \cdot 10^{-13} \\ 1.2664 \cdot 10^{-1} & -4.9398 \cdot 10^{-1} & 2.5026 & -3.2527 \cdot 10^{-3} & 1.5679 \cdot 10^{-12} \\ -1.6459 \cdot 10^{-4} & 6.4202 \cdot 10^{-4} & -3.2527 \cdot 10^{-3} & 1.0000 \cdot 10^2 & 4.8030 \cdot 10^{-8} \\ 1.5753 \cdot 10^{-12} & -1.5858 \cdot 10^{-13} & 1.5679 \cdot 10^{-12} & 4.8030 \cdot 10^{-8} & 1.0000 \cdot 10^5 \end{pmatrix}.$$

Although this matrix is semiseparable it can clearly be seen that the lower right element already converged to the largest eigenvalue. Representing this matrix now with the generators  $u$  and  $v$  gives us the following vectors:

$$u = ( 8.0861 \cdot 10^{11} \quad -3.6187 \cdot 10^{11} \quad 8.0391 \cdot 10^{10} \quad -1.0448 \cdot 10^8 \quad 1.0000 )^T$$

and

$$v = ( 1.5753 \cdot 10^{-12} \quad -1.5858 \cdot 10^{-13} \quad 1.5679 \cdot 10^{-12} \quad 4.8030 \cdot 10^{-8} \quad 1.0000 \cdot 10^5 ).$$

Because the first element of  $v$  is of the order  $10^{-12}$  and is constructed by summations of elements of order 1, we can expect that this element has a precision of only 4 significant decimal digits left. Using this number to reconstruct the elements within the matrix will only give these elements with a limited number of exact digits.

To overcome this problem we consider another representation. For a semiseparable matrix of dimension  $n$ , this representation consists of  $n - 1$  Givens transformations and a vector of length  $n$ . The Givens transformations are denoted as  $G = [G_1, \dots, G_{n-1}]$  and the vector as  $d = [d_1, \dots, d_n]$ .

The following figures denote how the semiseparable matrix can be reconstructed, using this information. The elements denoted by  $\boxtimes$  make up the semiseparable part of the matrix. The construction of the semiseparable matrix is for a matrix of size 5. Initially one starts on the first 2 rows of the matrix. The element  $d_1$  is placed in the upper left position, then a Givens transformation is applied, and finally to complete the first step element  $d_2$  is added in position  $(2, 1)$ . Only the first two columns and rows are shown here.

$$\begin{pmatrix} d_1 & 0 \\ 0 & 0 \end{pmatrix} \rightarrow G_1 \begin{pmatrix} d_1 & 0 \\ 0 & 0 \end{pmatrix} + \begin{pmatrix} 0 & 0 \\ 0 & d_2 \end{pmatrix} \rightarrow \begin{pmatrix} \boxtimes & 0 \\ \boxtimes & d_2 \end{pmatrix}.$$

The second step consists of applying the Givens transformation  $G_2$  on the second and the third row, furthermore  $d_3$  is added in position  $(3, 3)$ . Here only the first three columns are shown and the second and third row. This leads to:

$$\begin{pmatrix} \boxtimes & d_2 & 0 \\ 0 & 0 & 0 \end{pmatrix} \rightarrow G_2 \begin{pmatrix} \boxtimes & d_2 & 0 \\ 0 & 0 & 0 \end{pmatrix} + \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & d_3 \end{pmatrix} \rightarrow \begin{pmatrix} \boxtimes & \boxtimes & 0 \\ \boxtimes & \boxtimes & d_3 \end{pmatrix}.$$

This process can be repeated by applying the Givens transformation  $G_3$  on the third and the fourth row of the matrix, and afterwards adding the diagonal element  $d_4$ . After applying all the Givens transformations and adding all the diagonal elements, the lower triangular part of a symmetric semiseparable matrix is constructed. Because of the symmetry also the upper triangular part is known.

Suppose the Givens and vector representation of a semiseparable matrix  $S$  is known. When denoting the Givens transformations as:

$$G_l = \begin{pmatrix} c_l & -s_l \\ s_l & c_l \end{pmatrix}. \quad (2.2)$$

The elements  $S(i, j)$  with  $n > i \geq j$  are calculated in the following way:

$S(i, j) = c_i s_{i-1} s_{i-2} \cdots s_j d_j$ . When  $n = i$  we have  $S(i, j) = s_{n-1} s_{n-2} \cdots s_j d_j$ . When  $n \geq j > i$ ,  $S(i, j)$  can be calculated in a similar way, because of the symmetry. The elements of the semiseparable matrix can therefore be calculated in a stable way.

EXAMPLE 2 (Example 1 Continued). *The Givens-vector representation of the matrix in Example 1 is the following: (In the first row of  $G$  the elements  $c_1, \dots, c_4$  are stored and in the second row the elements  $s_1, \dots, s_4$ .)*

$$G = \begin{pmatrix} 9.0903 \cdot 10^{-1} & 9.7620 \cdot 10^{-1} & 9.9999 \cdot 10^{-1} & 1.0000 \\ -4.1672 \cdot 10^{-1} & -2.1686 \cdot 10^{-1} & -1.2997 \cdot 10^{-3} & 4.8030 \cdot 10^{-10} \end{pmatrix}$$

and

$$d = ( 1.4012 \quad 2.2778 \quad 2.5026 \quad 1.0000 \cdot 10^2 \quad 1.0000 \cdot 10^5 ) \quad (2.3)$$

All the elements of the semiseparable matrix can be reconstructed now with high relative precision.

Moreover this representation has another advantage connected to the  $QR$  factorization of this semiseparable matrix. Let  $\hat{G}_j$  represent the Givens transformation  $G_j$  embedded in an identity matrix of size  $n \times n$  such that the Givens transformation  $G_j$  is executed on rows  $j$  and  $j + 1$ . Then it is proven in [18] that the  $QR$  factorization of  $S$  can be written as:

$$\begin{aligned} S &= \hat{G}_{n-1} \cdots \hat{G}_2 \hat{G}_1 R \\ &= QR \end{aligned}$$

with  $R$  upper triangular.

### 3. Orthogonal similarity transformation to reduce a symmetric matrix to a semiseparable one.

**3.1. Theory.** An algorithm to transform a symmetric matrix into a semiseparable one by an orthogonal similarity transformation is presented in this section. The constructive proof of the next theorem, provides the algorithm.

THEOREM 3.1. *Let  $A$  be a symmetric matrix. Then there exists an orthogonal matrix  $U$  such that*

$$U^T A U = S, \quad (3.1)$$

where  $S$  is a semiseparable matrix.

*Proof.* The proof is a constructive one and is made by induction on the rows of the matrix  $A$ . We will construct a similar symmetric semiseparable matrix from a symmetric one. Let  $A_0^{(0)} \equiv A$ . Let  $G_i^{(k)}$  be the Givens transformation, such that the product  $A_{i-1}^{(k)} G_i^{(k)}$  has the entry  $(n-k, i)$  annihilated and modifying the  $i$ th and the  $(i+1)$ th columns of  $A_{i-1}^{(k)}$ .

Step 1 We will start by constructing a similarity transformation which will make the last two rows (columns) linearly dependent in the lower (upper) triangular part. To this end, we multiply  $A_0^{(0)}$  to the left by  $G_1^{(0)T}$  and to the right by  $G_1^{(0)}$  to annihilate the elements in position  $(1, n)$  and  $(n, 1)$  in  $A_0^{(0)}$ , respectively,

$$\begin{array}{c}
 \rightarrow \\
 \rightarrow
 \end{array}
 \begin{pmatrix}
 \downarrow & \downarrow & & & \\
 \times & \times & \cdots & \times & \otimes \\
 \times & \ddots & \vdots & \vdots & \times \\
 \cdots & \cdots & \ddots & \vdots & \vdots \\
 \times & \cdots & \cdots & \ddots & \times \\
 \otimes & \times & \cdots & \times & \times
 \end{pmatrix}
 \xrightarrow{G_1^{(0)T} A_0^{(0)} G_1^{(0)}}
 \begin{pmatrix}
 \times & \times & \cdots & \times & 0 \\
 \times & \ddots & \vdots & \vdots & \times \\
 \cdots & \cdots & \ddots & \vdots & \vdots \\
 \times & \cdots & \cdots & \ddots & \times \\
 0 & \times & \cdots & \times & \times
 \end{pmatrix}$$

$$\begin{array}{c}
 \Downarrow \\
 \Downarrow
 \end{array}
 A_0^{(0)} \xrightarrow{G_1^{(0)T} A_0^{(0)} G_1^{(0)}} A_1^{(0)}.$$

Note that  $\times$  denotes arbitrary elements of the matrix, while  $\otimes$  denotes the elements which are annihilated by the orthogonal similarity transformation. Summarizing, we obtain that  $A_1^{(0)} \equiv G_1^{(0)T} A_0^{(0)} G_1^{(0)}$ .

Continuing this process of annihilating all the elements in the last row (column), except for the element in position  $(n, n-1)$  ( $(n-1, n)$ ), we get

$$A_{n-2}^{(0)} = \begin{pmatrix}
 \times & \times & \cdots & \times & 0 \\
 \times & \ddots & \vdots & \vdots & \vdots \\
 \cdots & \cdots & \ddots & \times & 0 \\
 \times & \cdots & \times & \times & \times \\
 0 & \cdots & 0 & \times & \times
 \end{pmatrix}.$$

Multiplying  $A_{n-2}^{(0)}$  to the left by  $G_{n-1}^{(0)T}$ , we have the following situation,

$$\begin{array}{c}
 \rightarrow \\
 \rightarrow
 \end{array}
 \begin{pmatrix}
 \times & \times & \cdots & \times & 0 \\
 \times & \ddots & \vdots & \vdots & \vdots \\
 \cdots & \cdots & \ddots & \vdots & 0 \\
 \times & \cdots & \cdots & \times & \otimes \\
 0 & \cdots & 0 & \times & \times
 \end{pmatrix}
 \xrightarrow{G_{n-1}^{(0)T} A_{n-2}^{(0)}}
 \begin{pmatrix}
 \times & \times & \cdots & \times & 0 \\
 \times & \ddots & \vdots & \vdots & \vdots \\
 \cdots & \cdots & \ddots & \times & 0 \\
 \boxtimes & \cdots & \boxtimes & \times & 0 \\
 \boxtimes & \cdots & \boxtimes & \times & \times
 \end{pmatrix}$$

$$\begin{array}{c}
 \Downarrow \\
 \Downarrow
 \end{array}
 A_{n-2}^{(0)} \xrightarrow{G_{n-1}^{(0)T} A_{n-2}^{(0)}} \tilde{A}_{n-2}^{(0)},$$

i.e., the last two rows are proportional with the exception of the entries in the last two columns, (to emphasize the semiseparable structure among the rows (columns) we denote by  $\boxtimes$  the elements of the matrix belonging to these rows (columns)).

Let  $\tilde{A}_{n-2}^{(0)} \equiv G_{n-1}^{(0)T} A_{n-2}^{(0)}$ . Moreover, multiplying  $\tilde{A}_{n-2}^{(0)}$  to the right by  $G_{n-1}^{(0)}$ , the last two columns become linearly dependent above and on the main diagonal, i.e., they form an upper-semiseparable part. Because of symmetry, the last two rows become linearly dependent below and on the main diagonal and form a lower-semiseparable

part:

$$\begin{array}{ccc}
 \begin{pmatrix} \times & \times & \cdots & \times & 0 \\ \times & \ddots & \vdots & \vdots & \vdots \\ \cdots & \cdots & \ddots & \vdots & 0 \\ \boxtimes & \cdots & \boxtimes & \times & 0 \\ \boxtimes & \cdots & \boxtimes & \times & \times \end{pmatrix} & \xrightarrow{\tilde{A}_{n-2}^{(0)} G_{n-1}^{(0)}} & \begin{pmatrix} \times & \times & \cdots & \boxtimes & \boxtimes \\ \times & \ddots & \vdots & \vdots & \vdots \\ \cdots & \cdots & \ddots & \boxtimes & \boxtimes \\ \boxtimes & \cdots & \boxtimes & \boxtimes & \boxtimes \\ \boxtimes & \cdots & \boxtimes & \boxtimes & \boxtimes \end{pmatrix} \\
 \updownarrow & & \\
 \tilde{A}_{n-2}^{(0)} & \xrightarrow{\tilde{A}_{n-2}^{(0)} G_{n-1}^{(0)}} & A_{n-1}^{(0)}.
 \end{array}$$

To start the next step,  $A_0^{(1)}$  is defined as  $A_0^{(1)} \equiv A_{n-1}^{(0)}$ .

Step 2 Let  $m = n - j$ ,  $1 \leq j < n$ . Assume by induction that  $A_0^{(m)}$  has the lower (upper) triangular part already semiseparable from row  $n$  up to row  $j$  (column  $j$  to  $n$ ). We will now prove that we can make the lower (upper) triangular part semiseparable up to row  $j - 1$  (column  $j - 1$ ). Let us denote the lower- (and upper-)triangular elements which form a semiseparable part with  $\boxtimes$ . So matrix  $A_0^{(m)}$  looks like:

$$A_0^{(m)} = \begin{pmatrix} \times & \cdots & \times & \boxtimes & \cdots & \boxtimes \\ \vdots & \ddots & \cdots & \vdots & \vdots & \vdots \\ \times & \cdots & \times & \boxtimes & \cdots & \boxtimes \\ \boxtimes & \cdots & \boxtimes & \boxtimes & \cdots & \boxtimes \\ \vdots & \cdots & \cdots & \cdots & \ddots & \vdots \\ \boxtimes & \cdots & \boxtimes & \boxtimes & \cdots & \boxtimes \end{pmatrix} \begin{array}{l} \leftarrow 1 \\ \vdots \\ \leftarrow j-1 \\ \leftarrow j \\ \vdots \\ \leftarrow n \end{array}.$$

Because the submatrix of  $A_0^{(m)}$  consisting of the first  $j - 1$  rows and the last  $n - j + 1$  columns is already semiseparable, it has rank  $\leq 1$ . Hence, we can construct a Givens transformation  $G_1^{(m)}$  such that when we multiply  $A_0^{(m)}$  to the left by  $G_1^{(m)T}$  all elements  $(1, j)$ ,  $(1, j + 1)$  up to  $(1, n)$  of the first row are annihilated. In a similar way, we can construct the Givens transformations  $G_2^{(m)} \cdots G_{j-2}^{(m)}$  which make zero elements in columns  $j, j + 1, \dots, n$  in rows 2 up to  $j - 2$ . Applying these similarity transformations, we obtain the following matrix

$$A_{j-2}^{(m)} \equiv G_{j-2}^{(m)T} \cdots G_1^{(m)T} A_0^{(m)} G_1^{(m)} \cdots G_{j-2}^{(m)} = \begin{pmatrix} \times & \cdots & \times & \times & 0 & \cdots & 0 \\ \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \times & \cdots & \ddots & \times & 0 & \vdots & 0 \\ \times & \cdots & \times & \times & \boxtimes & \cdots & \boxtimes \\ 0 & \cdots & 0 & \boxtimes & \boxtimes & \cdots & \boxtimes \\ \vdots & \cdots & \vdots & \cdots & \cdots & \ddots & \vdots \\ 0 & \cdots & 0 & \boxtimes & \boxtimes & \cdots & \boxtimes \end{pmatrix} \begin{array}{l} \leftarrow 1 \\ \vdots \\ \leftarrow j-1 \\ \leftarrow j \\ \vdots \\ \leftarrow n \end{array}$$

Since the rows  $j - 1$  and  $j$  are proportional for the indexes of columns greater than  $j - 1$ , also the Givens transformation  $G_{j-1}^{(m)T}$  can be constructed annihilating all the entries in row  $j - 1$  with column index greater than  $j - 1$ . Furthermore, the product



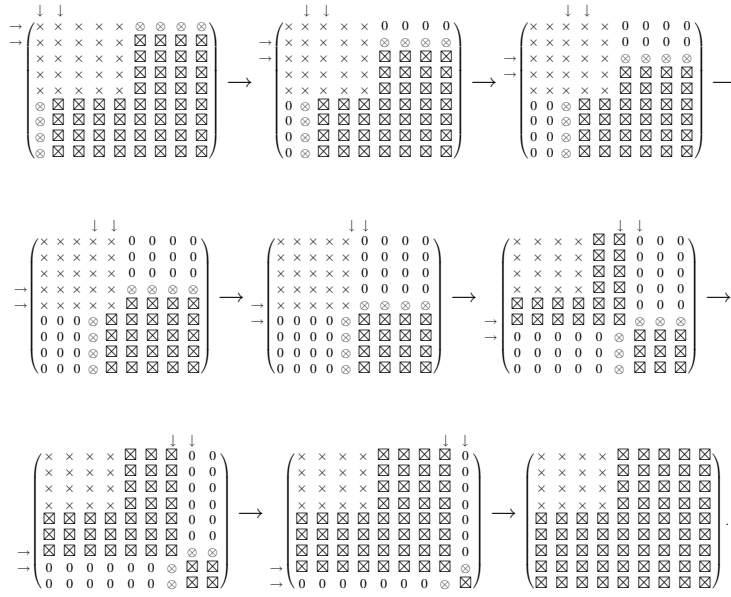


FIGURE 3.2. Description of the similarity transformations used to retrieve the semiseparable structure.

The proposed algorithm itself, however, while running to completion, gives already information on the spectrum of the initial matrix. In Section 6 we will show that the eigenvalues of the subsequent bottom-right semiseparable matrices computed by the proposed algorithm, are nothing else but the Ritz values obtained by the Lanczos' method.

Moreover, the algorithm can be considered as a nested subspace iteration method as shown in Section 7.

First, we look at a more detailed implementation of the algorithm in the next subsection as well as the computational complexity in Section 5.

**3.2. Effective implementation of the reduction.** Now we describe how to implement the method above using the Givens-vector representation of a semiseparable matrix.

This representation gives information which can directly be used in the  $QR$  algorithm applied to the resulting semiseparable matrix [19].

The implementation involves detailed shuffling of indices, therefore only the mathematics behind the implementation is given. The MATLAB-files can be downloaded from <http://www.cs.kuleuven.ac.be/~marc>. Following the mathematical details, one should be able to understand and/or reconstruct the algorithm.

Suppose rows  $j, j+1, \dots, n$  (columns  $j, j+1, \dots, n$ ) are already semiseparable. We can represent this semiseparable part by a row-vector  $R^{(j)} \in \mathbb{R}^{1 \times (j-1)}$ , Givens transformations

$$\begin{pmatrix} c_j & -s_j \\ s_j & c_j \end{pmatrix}, \quad k = j, j+1, \dots, n-1$$

and a vector  $[d_j, d_{j+1}, \dots, d_n]$ . At this point, the matrix similar to the original symmetric

matrix can be divided into four blocks:

$$A_0^{(m)} = \left( \begin{array}{cc|ccc} A^{(j)} & c_j R_j^T & c_{j+1} s_j R_j & \cdots & s_{n-1} \dots s_j R_j^T \\ c_j R_j & c_j d_j & c_{j+1} s_j d_j & \cdots & s_{n-1} \dots s_j d_j \\ \hline c_{j+1} s_j R_j & c_{j+1} s_j d_j & c_{j+1} d_{j+1} & & \\ \vdots & & & \ddots & \vdots \\ s_{n-1} \dots s_j R_j & & & \cdots & d_n \end{array} \right) \quad (3.2)$$

with  $m = n - j$  and  $A^{(j)} \in \mathbb{R}^{(j-1) \times (j-1)}$ .

The first substeps in step  $m$  of the method described in the proof of Theorem 3.1, eliminate the elements  $1, 2, \dots, j-2$  in row  $j$  by multiplying  $A_0^{(m)}$  to the right by the Givens transformations  $G_1^{(m)}, G_2^{(m)}, \dots, G_{j-2}^{(m)}$ , i.e.,

$$R_j G_1^{(m)} G_2^{(m)} \dots G_{j-2}^{(m)} = [0, \dots, 0, \tilde{\alpha}_j] = \tilde{R}_j.$$

It is clear that we can obtain a similar result by applying a Householder transformation  $H^{(m)}$  on the row  $R_j$  such that the following equation is obtained:

$$R_j H^{(m)} = [0, \dots, 0, \alpha_j] = \hat{R}_j,$$

with  $|\alpha_j| = |\tilde{\alpha}_j|$ . Performing the similarity Householder transformation on the matrix (3.2) transforms this matrix into the following one:

$$\left( \begin{array}{cccc|ccc} H^{(m)T} A^{(j)} H^{(m)} & c_j \hat{R}_j^T & c_{j+1} s_j \hat{R}_j^T & \cdots & s_{n-1} \dots s_j \hat{R}_j^T \\ c_j \hat{R}_j & c_j d_j & c_{j+1} s_j d_j & \cdots & s_{n-1} \dots s_j d_j \\ \hline c_{j+1} s_j \hat{R}_j & c_{j+1} s_j d_j & c_{j+1} d_{j+1} & & \vdots \\ \vdots & & & \ddots & \\ s_{n-1} \dots s_j \hat{R}_j & \cdots & & & d_n \end{array} \right). \quad (3.3)$$

From this point the Givens transformation  $G_{j-1}^{(m)}$  can be calculated, such that the following equation is satisfied:

$$[\alpha_j, d_j] G_{j-1}^{(m)} = [0, \alpha_{j+1}] \quad \text{with} \quad G_{j-1}^{(m)} = \begin{pmatrix} c_{j-1} & s_{j-1} \\ -s_{j-1} & c_{j-1} \end{pmatrix}$$

The upper left part  $H^{(m)T} A^{(j)} H^{(m)}$  can be written as:

$$H^{(m)T} A^{(j)} H^{(m)} = \begin{pmatrix} A^{(j-1)} & R_{j-1} \\ R_{j-1} & \tilde{d}_{j-1} \end{pmatrix}.$$

After applying the similarity Givens transformation  $G_{j-1}^{(m)}$  on the matrix (3.3) we get the following matrix:

$$\left( \begin{array}{cccc|ccc} A^{(j-1)} & c_{j-1} R_{j-1}^T & s_{j-1} R_{j-1}^T & 0 & \cdots & 0 \\ c_{j-1} R_{j-1} & c_{j-1} \tilde{d}_{j-1} & s_{j-1} \tilde{d}_{j-1} & 0 & \cdots & 0 \\ s_{j-1} R_{j-1} & s_{j-1} \tilde{d}_{j-1} & c_j \alpha_{j+1} & c_{j+1} s_j \alpha_{j+1} & \cdots & s_{n-1} \dots s_j \alpha_{j+1} \\ 0 & 0 & c_{j+1} s_j \alpha_{j+1} & c_{j+1} d_{j+1} & & \\ \vdots & \vdots & \vdots & & \ddots & \vdots \\ 0 & 0 & s_{n-1} \dots s_j \alpha_{j+1} & & \cdots & d_n \end{array} \right), \quad (3.4)$$

with  $\hat{d}_{j-1} = \tilde{d}_{j-1}c_{j-1} + s_{j-1}c_j\alpha_j$ . The Givens transformation  $G_{j-1}^{(m)T}$  and the element  $\hat{d}_{j-1}$  can already be stored as part of the representation of the new semiseparable part which has to be formed at the bottom of the matrix.

The process explained here can now be repeated to make all rows  $j-1$ ,  $j$  up to  $n$  semiseparable, because the matrix (3.4) has essentially the same structure as the matrix (3.3).

**Note:** The latter step, i.e. the reduction of the lower right part towards the semiseparable structure, can also be applied directly on a semiseparable matrix. This corresponds to performing a  $QR$  step (in fact a  $QL$  step) without shift on this semiseparable matrix [19].

**4. The reduction of an arbitrary matrix.** It is well known how any matrix can be transformed into an upper Hessenberg one by an orthogonal similarity transformation. To find all the eigenvalues of the original matrix, then the  $QR$  algorithm can be applied to this upper Hessenberg matrix. By using an algorithm similar to that one described in the previous section, it is possible to transform every matrix into the sum of a semiseparable and a strictly upper triangular part. This is proved in the next theorem.

**THEOREM 4.1.** *Let  $A$  be an  $n \times n$  matrix.*

*Then there exist a symmetric semiseparable matrix  $S$  of semiseparability rank 1, a strictly upper triangular matrix  $R$  and an orthogonal matrix  $U$  such that the following equation is satisfied:*

$$A = U(S + R)U^T,$$

*i.e.,  $A$  is similar to the sum of a symmetric semiseparable and a strictly upper triangular matrix  $R$ , by means of orthogonal transformations.*

*Proof.* Following the same reduction as in the proof of Theorem 3.1, we can find an orthogonal matrix  $U$  such that the lower triangular part of  $U^T A U$  is semiseparable.

Hence, the lower triangular part of  $A$  can be written as the lower triangular part of a symmetric semiseparable matrix  $S$ , i.e.,

$$\begin{aligned} U^T A U &= \text{tril}(S) + \bar{R} \\ &= S + (\bar{R} - \text{triu}(S)) \\ &= S + R. \end{aligned}$$

■

Note that the algorithm to reduce an arbitrary matrix, by means of orthogonal transformations into a similar matrix whose lower triangular part is semiseparable also requires  $O(n^3)$  operations.

## 5. The computational complexity of each of the algorithms.

**5.1. Reduction of a symmetric matrix into a similar semiseparable one.** When reducing a symmetric matrix to a similar semiseparable one, the implementation can be done using only Givens transformations or using Householder and Givens transformations. One can easily see that the complexity can be divided into two parts: first there is the application of a Householder transformation or a number of Givens transformations on the upper left block of the matrix and secondly, there is the updating of the semiseparable part. Executing the first part of all the steps needs the same number of operations as reducing a symmetric matrix into a similar tridiagonal one by Householders or by Givens transformations. Hence, the computational complexity of this part is:

$$\begin{aligned} &2n^3 + O(n^2) \text{ when using Givens transformations} \\ &\frac{4}{3}n^3 + O(n^2) \text{ when using Householder transformations.} \end{aligned}$$

Executing the second part of all the steps needs  $O(n^2)$  operations. More details can be found in the next subsection.

**5.2. Reduction to a tridiagonal matrix.** The third approach to reduce a symmetric matrix into semiseparable form is the following. First we reduce the matrix into a similar tridiagonal one by standard algorithms (This can be achieved with Householder or Givens transformations (see, i.e., [9])). The latter tridiagonal matrix can then be reduced into the semiseparable form. In this case the first part of each step has not to be performed. Note that this algorithm involves essentially the same orthogonal similarity transformations as the algorithms described before. Only the order in which they are applied is different. The results of Section 6 state however that it is worth to intertwine the two reductions, instead of first reducing towards tridiagonal form and thereafter to a semiseparable matrix. Hence, the number of operations to reduce a tridiagonal into a semiseparable matrix is the same as for the second step in the reduction from a symmetric matrix into semiseparable form.

The calculation of the Givens needs 7 flops, as described in [9]. The multiplication on the left hand and right hand side takes 11 operations, when everything is optimized. Therefore, the computational complexity is

$$\sum_{j=n:-1:2} \left( \sum_{i=j:1:n} 18 \right) = 9n^2 + O(n).$$

**5.3. Reduction of a nonsymmetric matrix.** As already mentioned, the algorithm applied to nonsymmetric matrices, reduces the matrix to the sum of a semiseparable and a strictly upper triangular matrix. To implement this, similar algorithms as mentioned above can be used. We now take a look at the complexity of this reduction. Let us assume that we first reduce the matrix to an upper Hessenberg one, and then apply the algorithm to make the lower triangular part semiseparable (This is a similar algorithm as applied to tridiagonal matrices). Then we first have the reduction to Hessenberg form, which costs  $5/3n^3 + O(n^2)$  flops. The reduction from Hessenberg to lower semiseparable form costs an extra  $3n^3 + O(n^2)$  flops.

**5.4. Comparison with other algorithms.** When we compare the algorithm which tridiagonalizes a symmetric matrix, with the algorithm which constructs a similar semiseparable matrix, we find that the latter algorithm only performs  $9n^2 + O(n)$  more than the former algorithm while both algorithms have an overall complexity of  $O(n^3)$ . It is worthwhile to perform these additional operations as is shown by the convergence properties as studied in the next sections.

**6. Connection with the Lanczos-Ritz values.** In this section, we show that at each step of the algorithm the eigenvalues of the semiseparable matrix at the bottom right are the Ritz-values of the matrix  $A$  with respect to the Krylov subspace  $\mathcal{K}_n$ : (whith  $\langle x, y, z, \dots \rangle$  we denote the subspace spanned by the vectors  $x, y, z, \dots$ )

$$\mathcal{K}_n = \langle e_n, Ae_n, \dots, A^m e_n \rangle,$$

with  $e_n$  the  $n$ th vector of the standard basis in  $\mathbb{R}^n$ . Hence the convergence behaviour of these eigenvalues is the same as the convergence behaviour of the Lanczos' algorithm with the initial vector  $e_n$ .

Let us partition the matrix  $A_0^{(m)}$  (from Equation (3.2)) as follows:

$$A_0^{(m)} = \left( \begin{array}{c|c} A_m & u_m v_m^T \\ \hline v_m u_m^T & S_m \end{array} \right), \tag{6.1}$$

with  $A^{(j)} = A_m \in \mathbb{R}^{(n-m-1) \times (n-m-1)}$ ,  $m = n - j$ ,  $u_m \in \mathbb{R}^{(n-m-1) \times 1}$ ,  $v_m \in \mathbb{R}^{(m+1) \times 1}$  and  $S_m \in \mathbb{R}^{(m+1) \times (m+1)}$ . From the semiseparable structure, we know that  $[v_m, S_m e_1]$  has rank one.

The matrix  $A_0^{(m)}$  is similar to the original matrix  $A = A_0^{(0)}$ , by means of orthogonal transformations as described in Theorem 3.1:

$$\begin{aligned} A_0^{(m)} &= Q_m^T \dots Q_1^T A Q_1 \dots Q_m \\ &= Q_{1:m}^T A Q_{1:m}. \end{aligned}$$

We partition the matrix  $Q_{1:m}$  as follows:

$$Q_{1:m} = \left[ \overleftarrow{Q}_{1:m} \mid \overrightarrow{Q}_{1:m} \right] \text{ with } \begin{cases} \overleftarrow{Q}_{1:m} \in \mathbb{R}^{n \times (n-m-1)} \\ \overrightarrow{Q}_{1:m} \in \mathbb{R}^{n \times (m+1)}. \end{cases}$$

This means,

$$A \left[ \overleftarrow{Q}_{1:m} \mid \overrightarrow{Q}_{1:m} \right] = \left[ \overleftarrow{Q}_{1:m} \mid \overrightarrow{Q}_{1:m} \right] \left( \begin{array}{c|c} A_m & u_m^T v_m \\ \hline v_m u_m^T & S_m \end{array} \right).$$

Hence, the eigenvalues of  $S_m$  are the Ritz-values of  $A$  with respect to the subspace spanned by the columns of  $\overrightarrow{Q}_{1:m}$  (see e.g. [5]). We will now prove that these Ritz values are the same as the Lanczos-Ritz values.

**THEOREM 6.1.** *Suppose we have*

$$A \left[ \overleftarrow{Q}_{1:m} \mid \overrightarrow{Q}_{1:m} \right] = \left[ \overleftarrow{Q}_{1:m} \mid \overrightarrow{Q}_{1:m} \right] \left( \begin{array}{c|c} A_m & u_m^T v_m \\ \hline v_m u_m^T & S_m \end{array} \right) \quad (6.2)$$

with  $Q_{1:m} = \left[ \overleftarrow{Q}_{1:m} \mid \overrightarrow{Q}_{1:m} \right]$  constructed according to Theorem 3.1. Then we have that the columns of  $\overrightarrow{Q}_{1:m}$  form an orthogonal basis of  $\mathcal{K}_m$ , i.e.,

$$\overrightarrow{Q}_{1:m} = K_m C_m$$

with

$$K_m = [e_n, A e_n, \dots, A^m e_n]$$

and  $C_m$  a nonsingular  $(m+1) \times (m+1)$  matrix.

This means that the eigenvalues of  $S_m$  are equal to the eigenvalues of  $T_m$ , with

$$T_m = Q_{A,m}^T A Q_{A,m}$$

and  $Q_{A,m} R_m$  is the  $QR$  factorization of the matrix  $K_m$ . This means that the Ritz values calculated here are the same as the one computed by the Lanczos' algorithm.

To prove Theorem 6.1 another theorem is needed:

**THEOREM 6.2.**

$$Q_{1:m} \langle e_n \rangle = A^m \langle e_n \rangle,$$

for  $m = 0, 1, 2, \dots$  where  $Q_{1:0} = I_n$ .

*Proof.* By induction. It is true for  $m = 0$ :  $Q_{1:0} \langle e_n \rangle = \langle e_n \rangle$ .

Suppose the theorem is true for  $m - 1$ , i.e.,

$$Q_{1:m-1} \langle e_n \rangle = A^{m-1} \langle e_n \rangle.$$

From

$$Q_m^T (Q_{1:m-1}^T A Q_{1:m-1}) = \underbrace{\left( \begin{array}{c|ccc} X & 0 & \dots & 0 \\ X & \times & & 0 \\ \vdots & \vdots & \ddots & \\ X & \times & \dots & \times \end{array} \right)}_m \left. \vphantom{\begin{array}{c|ccc} X & 0 & \dots & 0 \\ X & \times & & 0 \\ \vdots & \vdots & \ddots & \\ X & \times & \dots & \times \end{array}} \right\} m.$$

We derive that

$$A Q_{1:m-1} \langle e_n \rangle = Q_{1:m} \langle e_n \rangle \quad \text{or} \quad A^m \langle e_n \rangle = Q_{1:m} \langle e_n \rangle.$$

■

Note that a more general theorem is possible:

THEOREM 6.3.

$$Q_{1:m} \langle e_{n-m+1}, \dots, e_n \rangle = A^m \langle e_{n-m+1}, \dots, e_n \rangle$$

*Proof.* In a similar way as the proof of Theorem 6.2. ■

*Proof.* (proof of Theorem 6.1.) It will be proved by induction. When  $m = 0$  we put  $Q_{1:m} = I_m$ . Hence,  $\vec{Q}_{1:m} = e_n = K_0$ .

Suppose now by induction that

$$\vec{Q}_{1:m} = K_m C_m.$$

We are going to prove now that

$$\vec{Q}_{1:m+1} = K_{m+1} C_{m+1}$$

with  $C_{m+1}$  a nonsingular  $(m+1) \times (m+1)$  matrix. The matrix  $Q_{1:m}$  is transformed into  $Q_{1:m+1}$  as follows:

$$\begin{aligned} Q_{1:m+1} &= Q_{1:m} Q_{m+1} \\ &= Q_{1:m} \left( \begin{array}{c|c} H^{(m)} & 0 \\ \hline 0 & I \end{array} \right) G_{j-1}^{(m)} \cdots G_{n-1}^{(m)}. \end{aligned}$$

$H^{(m)}$  is the Householder transformation of dimension  $(n-m-1) \times (n-m-1)$  such that  $u_m^T H^{(m)}$  (see (6.1)) is a vector with all zeros except in the last position.

Hence the matrix  $Q_{m+1}$  has the following structure:

$$Q_{m+1} = \left( \begin{array}{c|c|ccc} \tilde{H} & h & & & \\ \hline 0 & \times & \times & \dots & \times \\ & 0 & \times & & \\ \vdots & \vdots & & \ddots & \vdots \\ 0 & 0 & 0 & & \times \times \end{array} \right).$$

with

$$\begin{cases} \tilde{H} \in \mathbb{R}^{(n-m-1) \times (n-m-2)} \\ h \in \mathbb{R}^{(n-m-1) \times 1} \\ q \in \mathbb{R}^{(m+1) \times 1} \end{cases}.$$

Therefore the matrix  $\vec{Q}_{1:m+1}$  can be written as:

$$\vec{Q}_{1:m+1} = [\overleftarrow{Q}_{1:m} | \overrightarrow{Q}_{1:m}] \left( \begin{array}{c|ccc} h & & & \\ \times & \times & \dots & \times \\ 0 & \times & & \\ \vdots & & \ddots & \vdots \\ 0 & 0 & & \times \times \end{array} \right).$$

This can be rewritten as:

$$\vec{Q}_{1:m+1} = \overleftarrow{Q}_{1:m} h [1, q^T] + \overrightarrow{Q}_{1:m} \left( \begin{array}{c|ccc} \times & \times & \dots & \times \\ 0 & \times & & \\ \vdots & & \ddots & \vdots \\ 0 & 0 & & \times \times \end{array} \right). \quad (6.3)$$

If we assume that the last component of vector  $q$  is  $\alpha \neq 0$ , then the equality of the last column of the left and the righthand side of (6.3) gives us:

$$\vec{Q}_{1:m+1} e_{m+1} = r + \overrightarrow{Q}_{1:m} \left( \begin{array}{c|ccc} \times & \times & \dots & \times \\ 0 & \times & & \\ \vdots & & \ddots & \vdots \\ 0 & 0 & & \times \times \end{array} \right) e_{m+1}$$

with

$$r = \overleftarrow{Q}_{1:m} \alpha h. \quad (6.4)$$

Together with Theorem 6.2 and the fact that the vectors of  $\vec{Q}_{1:m}$  belong to  $\mathcal{K}_m \subset \mathcal{K}_{m+1}$ , we have that  $r \in \mathcal{K}_{m+1}$ . Hence, we get that all the columns of  $\vec{Q}_{1:m+1}$  belong to  $\mathcal{K}_{m+1}$  and because they are independent from each other, they form an orthogonal basis for  $\mathcal{K}_{m+1}$ . If  $\alpha = 0$  this means that  $r = 0$  and we reached an invariant subspace. ■

**7. Subspace Iteration while reducing the matrix.** The previous section shows that it is worth building up the intermediate semiseparable parts, instead of first reducing to the tridiagonal form and then constructing a semiseparable matrix from it. The intermediate matrices do not only contain the Lanczos-Ritz values in the already semiseparable part. During the construction, also a kind of nested subspace iteration is performed on the matrix. Depending on the gaps, and the approximation of the Ritz values the subspace iteration will create blocks in the semiseparable structure.

**7.1. Theoretical proof.** At each step of the algorithm introduced in Theorem 3.1, one more row (column) is added by means of orthogonal transformations to the set of the rows (columns) of the matrix already proportional to each other. In this section, using arguments similar to those considered in [23, 24], we show that this algorithm can be interpreted as a kind of nested subspace iteration method [9], where the size of the vector subspace is increased by one and a change of coordinate system is made at each step of the algorithm. As a consequence, depending on the gap between the eigenvalues, the semiseparable part of the matrix will converge to a block diagonal matrix, and the eigenvalues of these blocks converge to the largest eigenvalues in absolute value of the original symmetric matrix.

Given a matrix  $A$  and an initial subspace  $\mathcal{S}^{(0)}$ , the subspace iteration method [9] can be described as follows

$$\mathcal{S}^{(i)} = A \mathcal{S}^{(i-1)}, \quad i = 1, 2, 3, \dots$$

Under weak assumptions on  $A$  and  $\mathcal{S}^{(0)}$ , the  $\mathcal{S}^{(i)}$  converge to an invariant subspace. We will see that the reduction algorithm from symmetric to semiseparable matrix can be seen as such a kind of subspace iteration, where the size of the subspace grows by one dimension at each step of the algorithm. Let  $A_0^{(0)} = A$ . Suppose we have only performed the first orthogonal similarity transformations such that the rows (columns)  $n$  and  $n - 1$  are already proportional:

$$A_0^{(1)} = Q_1^T A_0^{(0)} Q_1, \quad (7.1)$$

where  $A_0^{(1)}$  has the semiseparable structure in the rows (columns)  $n$  and  $n - 1$  and  $Q_1 = [q_1^{(1)}, \dots, q_n^{(1)}]$ . From (7.1), we can write

$$A_0^{(0)} = Q_1 \begin{pmatrix} \times & \cdots & \times & 0 \\ \vdots & & \vdots & \vdots \\ \times & \cdots & \times & 0 \\ \times & \cdots & \times & \times \end{pmatrix} \equiv Q_1 L_1. \quad (7.2)$$

Let  $e_1, \dots, e_n$  be the standard basis vectors of  $\mathbb{R}^n$ . From (7.2), because of the structure of  $L_1$ , it can clearly be seen that:

$$A_0^{(0)} \langle e_n \rangle = \langle q_n^{(1)} \rangle.$$

This means that the last column of  $A_0^{(0)}$  and  $q_n^{(1)}$  span the same one-dimensional space. In fact one subspace iteration step is performed on the column  $e_n$ . The first step of the algorithm is completed when the following orthogonal transformation is performed,

$$A_0^{(1)} = Q_1^T A_0^{(0)} Q_1.$$

The latter transformation can be interpreted as a change of coordinate system:  $A_0^{(0)}$  and  $A_0^{(1)}$  represent the same linear transformation with respect to different coordinate systems. Let  $y \in \mathbb{R}^n$ . Then  $y$  is represented in the new system by  $Q_1^T y$ . This means that for the vector  $q_n^{(1)}$  we get  $Q_1^T q_n^{(1)} = e_n$ . Summarizing, this means that one step of subspace iteration on the subspace  $\langle e_n \rangle$  is performed, resulting in a new subspace  $\langle q_n^{(1)} \rangle$ , and then, by means of a coordinate transformation, it is transformed back into the subspace  $\langle e_n \rangle$ . So, instead of working with a fixed matrix and changing subspaces, we work with fixed subspaces and changing matrices.

The second step can be interpreted in a completely analogous way. Suppose we have already the semiseparable structure in the last two rows (columns). Then we perform the following similarity transformation on  $A_0^{(1)}$ ,

$$A_0^{(2)} = Q_2^T A_0^{(1)} Q_2, \quad (7.3)$$

in order to make the rows (columns)  $n$  up to  $n - 2$  dependent. Using (7.3),  $A_0^{(1)}$  can be written as follows,

$$A_0^{(1)} = Q_2 \begin{pmatrix} \times & \cdots & \times & 0 & 0 \\ \vdots & & \vdots & \vdots & \vdots \\ \times & \cdots & \times & 0 & 0 \\ \times & \cdots & \times & \times & 0 \\ \times & \cdots & \times & \times & \times \end{pmatrix} = Q_2 L_2.$$

Considering the subspace  $\langle e_{n-1}, e_n \rangle$  and using the same notation as above, we have,

$$A_0^{(1)} \langle e_{n-1}, e_n \rangle = \langle q_{n-1}^{(2)}, q_n^{(2)} \rangle.$$

This means that the second step of the algorithm is a step of subspace iteration performed on a slightly grown subspace. For every new dependency that is created in the symmetric matrix  $A_0^{(i)}$ , the dimension of the subspace is increased by one.

This means that from step  $i$ ,  $i = 0, 1, \dots, n$ , (so there is dependency between the rows  $n$  up to  $n - i$ ), all the consecutive steps perform subspace iterations on the subspace of dimension  $i + 1$ . From [24], we know that these consecutive iterations on subspaces tend to create block upper triangular matrices. Hence, for a symmetric matrix these are block diagonal. Furthermore, the process works for all the nested subspaces at the same time, and so the semiseparable part of the matrices  $A_0^{(i)}$  generated by the proposed algorithm, becomes more and more block diagonal, and the blocks contain the eigenvalues having a comparable absolute value. This explains why the lower-right block already gives a good estimate of the largest eigenvalues, since they are connected to a subspace on which the subspace iteration step is already performed several times. Hence as a special case, denoting by  $z^{(i)}$  the eigenvector corresponding to the largest eigenvalue in absolute value  $\lambda$  of  $A_0^{(i)}$ ,  $i = 0, 1, 2, \dots$ , we can say that, if  $z^{(0)}$  has a nonzero last component and if the largest eigenvalue is unique, the sequence  $\{z^{(i)}\}$  converges to  $e_n$ , and, consequently, the lower-right element of  $A_0^{(i)}$  converges to  $\lambda$ .

This insight also opens several new perspectives. In many problems, only the few largest eigenvalues (in absolute value) need to be computed [1, 12, 21, 22, 25, 26]. In such cases, the proposed algorithm gives the required information after only few steps, without running the algorithm to completion. Moreover, because the sequence of similar matrices generated at each step of the algorithm converges to a block diagonal matrix, the original problem can be divided into smaller independent subproblems.

Furthermore, if the second step of the algorithm is “iterated”, the sequence of semiseparable matrices generated converges to a block diagonal matrix (see the remark at the end of Section 3.2).

We finish this section with a theorem from [24] concerning the speed of convergence of subspace iterations [24, Theorem 5.4].

**DEFINITION 7.1.** *Denote with  $\mathcal{S}$  and  $\mathcal{T}$  two subspaces, then the distance  $d(\mathcal{S}, \mathcal{T})$  between these two subspaces is defined in the following way:*

$$d(\mathcal{S}, \mathcal{T}) = \sup_{s \in \mathcal{S}, \|s\|_2=1} \inf_{t \in \mathcal{T}} \|s - t\|_2.$$

Using this definition, we can state the following convergence theorem:

**THEOREM 7.2.** *Let  $A \in \mathbb{C}^{n \times n}$ , and let  $p$  be a polynomial of degree  $\leq n$ . Let  $\lambda_1, \dots, \lambda_n$  denote the eigenvalues of  $A$ , ordered so that  $|p(\lambda_1)| \geq |p(\lambda_2)| \geq \dots \geq |p(\lambda_n)|$ . Suppose  $k$  is a positive integer less than  $n$  for which  $|p(\lambda_k)| > |p(\lambda_{k+1})|$ . Let  $(p_i)$  be a sequence of polynomials of degree  $\leq n$  such that  $p_i \rightarrow p$  as  $i \rightarrow \infty$  and  $p_i(\lambda_j) \neq 0$  for  $j = 1, \dots, k$  and all  $i$ . Let  $\rho = |p(\lambda_k)|/|p(\lambda_{k+1})|$ . Let  $\mathcal{T}$  and  $\mathcal{U}$  be the invariant subspaces of  $A$  associated with  $\lambda_1, \dots, \lambda_k$  and  $\lambda_{k+1}, \dots, \lambda_n$  respectively. Consider the nonstationary subspace iteration*

$$\mathcal{S}_i = p_i(A)\mathcal{S}_{i-1}$$

where  $\mathcal{S}_0$  is a  $k$ -dimensional subspace of  $\mathbb{C}^n$  satisfying  $\mathcal{S} \cap \mathcal{U} = \{0\}$ . Then for every  $\hat{\rho}$  satisfying  $\rho < \hat{\rho} < 1$  there exists a constant  $\hat{C}$  such that

$$d(\mathcal{S}_i, \mathcal{T}) \leq \hat{C}\hat{\rho}^i, \quad i = 1, 2, 3, \dots$$

In our case Theorem 7.2 can be applied with the polynomials  $p_i(z)$  and  $p(z)$  chosen in the following sense:  $p_i(z) = p(z) = z$ .

**7.2. Practical interpretation.** Taking a look at the numerical examples, sometimes it seems that initially there is no sign of subspace iteration. The proofs stated above are correct, but there is an interaction with the Lanczos' behaviour of the algorithm, which will sometimes lead to a delay in the convergence behaviour of the subspace iteration. This can only happen when, all the vectors  $\langle e_{n-i}, e_{n-i+1}, \dots, e_n \rangle$  have a small component in one or more directions of the eigenspace connected to the dominant eigenvalues. Before we show by an example, that this happens in practice, we first give a condition under which we are sure that the subspace iteration lets the matrix converge to one having a diagonal block containing the dominant eigenvalues. As soon as some of the Ritz values approximate all of the dominant eigenvalues, this convergence behaviour appears. To show this we assume that we can split up the eigenvalues of the initial matrix  $A$  into two subsets,  $\Lambda_1 = \{\lambda_{1,j} | j \in J\}$  and  $\Lambda_2 = \{\lambda_{2,i} | i \in I\}$ , with a gap between cluster 2 and cluster 1,  $\min_{i \in I} |\lambda_{2,i}| \gg \max_{j \in J} |\lambda_{1,j}|$ . Using the following notations  $\Delta_1 = \text{diag}(\Lambda_1)$  and  $\Delta_2 = \text{diag}(\Lambda_2)$ , we can write the matrix  $A_0^{(m)}$  in the following way:

$$\begin{pmatrix} \tilde{A} & R \\ R & S \end{pmatrix} = \begin{pmatrix} V_1 & W_1 \\ V_2 & W_2 \end{pmatrix} \begin{pmatrix} \Delta_1 & 0 \\ 0 & \Delta_2 \end{pmatrix} \begin{pmatrix} V_1^T & V_2^T \\ W_1^T & W_2^T \end{pmatrix} \quad (7.4)$$

where  $\tilde{A} = A^{(j)} = A_m$ ,  $R = u_m v_m^T$  and  $S = S_m$  in correspondence with equation (3.2) and (6.2). From equation (7.4), we have the following equation for  $S$ :

$$S = V_2 \Delta_1 V_2^T + W_2 \Delta_2 W_2^T.$$

Let the eigenvalue decomposition of  $S$  be denoted as:

$$S = \begin{pmatrix} V_S & W_S \end{pmatrix} \begin{pmatrix} \Delta_{S,1} & 0 \\ 0 & \Delta_{S,2} \end{pmatrix} \begin{pmatrix} V_S^T \\ W_S^T \end{pmatrix} \quad (7.5)$$

where we assume that some of the eigenvalues of  $S$  (in fact the Ritz values) approximate already the real dominant eigenvalues, this means assume that  $|\lambda_{S,2,i}| \approx |\lambda_{2,i}|, i \in I$ .

If  $W_2$  has full rank and is far from being not of full rank, there is no delay in the convergence behaviour of the subspace iteration. This corresponds to the condition that the last vectors  $\{e_{n-m}, e_{n-m+1}, \dots, e_n\}$  projected on the invariant subspace connected to the dominant eigenvalues  $\lambda_{2,i}$  have a large component in every direction of this subspace.

Via an Householder similarity transformation we can transform equation (7.4) into

$$\begin{pmatrix} \tilde{A} & \begin{pmatrix} 0 \\ v^T \end{pmatrix} \\ \begin{pmatrix} 0 & v \end{pmatrix} & S \end{pmatrix} = \begin{pmatrix} \bar{V}_1 & \bar{W}_1 \\ V_2 & W_2 \end{pmatrix} \begin{pmatrix} \Delta_1 & 0 \\ 0 & \Delta_2 \end{pmatrix} \begin{pmatrix} \bar{V}_1^T & V_2^T \\ \bar{W}_1^T & W_2^T \end{pmatrix}.$$

Then we can substitute  $S$  by its eigenvalue decomposition (7.5).

$$\begin{pmatrix} \tilde{A} & \begin{pmatrix} 0 \\ -\bar{v}^T \\ -\bar{v} \end{pmatrix} \\ \begin{pmatrix} 0 & -\bar{v} \\ -\bar{v} \end{pmatrix} & \begin{pmatrix} \Delta_{S,1} & 0 \\ 0 & \Delta_{S,2} \end{pmatrix} \end{pmatrix} = \begin{pmatrix} \bar{V}_1 & \bar{W}_1 \\ \bar{V}_2 & \bar{W}_2 \end{pmatrix} \begin{pmatrix} \Delta_1 & 0 \\ 0 & \Delta_2 \end{pmatrix} \begin{pmatrix} \bar{V}_1^T & \bar{V}_2^T \\ \bar{W}_1^T & \bar{W}_2^T \end{pmatrix}.$$

Note that  $W_2$  is of full rank, if and only if  $\bar{W}_2$  is of full rank. We get

$$\Delta_{2,S} = \begin{pmatrix} P \bar{V}_2 \Delta_1 & P \bar{W}_2 \Delta_2 \end{pmatrix} \begin{pmatrix} \bar{V}_2^T P^T \\ \bar{W}_2^T P^T \end{pmatrix},$$

where  $P$  is the projection matrix  $[0, I]$  of size  $n_{S,2} \times n$  where  $I$  is the identity matrix of size  $n_{S,2} \times n_{S,2}$ , with  $n_{S,2}$  the dimension of  $\Delta_{S,2}$ . This means that:

$$\Delta_{2,S} - P\bar{V}_2\Delta_1\bar{V}_2^T P^T = P\bar{W}_2\Delta_2\bar{W}_2^T P^T. \quad (7.6)$$

Note that  $P\bar{W}_2$  is a square matrix.

Because there is a gap between the eigenvalues  $\lambda_{1,j}, j \in J$  and  $\lambda_{2,i}, i \in I$  there will be a comparable gap between  $\lambda_{S,2,i} \in I$  and  $\lambda_{1,j}, j \in J$ . Hence the matrix at the lefthandside of equation (7.6) is far from singular. Therefore this is also true for the righthandside, and  $\bar{W}_2$  is of full rank.

In the remaining part of this paragraph, a brief explanation is given, why in certain situations the subspace iteration behaviour does not show up immediately from the start. Suppose the size of the block  $\Delta_2$  is 2. We can write the matrix  $A$  in decomposed form:

$$A_0^{(0)} = A = \begin{pmatrix} \bar{A} & a_n \\ a_n^T & \bar{\alpha} \end{pmatrix} = \begin{pmatrix} V & W \\ v^T & w^T \end{pmatrix} \begin{pmatrix} \Delta_1 & 0 \\ 0 & \Delta_2 \end{pmatrix} \begin{pmatrix} V^T & v \\ W^T & w \end{pmatrix}. \quad (7.7)$$

Applying already the Householder similarity transformation on the matrix  $A$  gives us the following decomposition:

$$\begin{aligned} & \begin{pmatrix} \tilde{H} & 0 \\ q^T & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} V & W \\ v^T & w^T \end{pmatrix} \begin{pmatrix} \Delta_1 & 0 \\ 0 & \Delta_2 \end{pmatrix} \begin{pmatrix} V^T & v \\ W^T & w \end{pmatrix} \begin{pmatrix} \tilde{H}^T & q & 0 \\ 0 & 0 & 1 \end{pmatrix} \\ &= \begin{pmatrix} \tilde{A} & a^T & 0 \\ a & \gamma & \beta \\ 0 & \beta & \alpha \end{pmatrix} = \tilde{A}^{(1)}. \end{aligned}$$

The Ritz-values are the eigenvalues of the two by two matrix:

$$\begin{pmatrix} \gamma & \beta \\ \beta & \alpha \end{pmatrix}.$$

In fact, to complete the previous step a Givens transformation should still be performed on the matrix  $\tilde{A}^{(1)}$ , to make the last two rows and columns dependent, and thereby transforming the matrix into  $A_0^{(1)}$ . Remark that this last Givens transformation does not change the eigenvalues of the bottomright submatrix.

We show now that the assumptions for an immediate convergence behaviour of the subspace iteration, are not satisfied. We prove that

$$[e_{n-1}, e_n]$$

does not have two large linearly independent components in the span of

$$\begin{pmatrix} \tilde{H} & 0 \\ q^T & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} W \\ w^T \end{pmatrix}.$$

We can write  $[e_{n-1}, e_n]$  as a linear combination of the eigenvectors:

$$[e_{n-1}, e_n] = \begin{pmatrix} \tilde{H} & 0 \\ q^T & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} V & W \\ v^T & w^T \end{pmatrix} \begin{pmatrix} C_1 \\ C_2 \end{pmatrix}.$$

The coordinates  $C_2$  of  $[e_{n-1}, e_n]$  with respect to the dominant eigenvectors, are the following:

$$\begin{aligned} C_2 &= (W^T \quad w) \begin{pmatrix} \tilde{H}^T & q & 0 \\ 0 & 0 & 1 \end{pmatrix} [e_{n-1}, e_n] \\ &= (W^T \quad w) \begin{pmatrix} q & 0 \\ 0 & 1 \end{pmatrix} = [W^T q, w]. \end{aligned}$$

Using equation (7.7), we get that

$$\begin{pmatrix} V^T & v \\ W^T & w \end{pmatrix} \begin{pmatrix} a_n \\ \bar{\alpha} \end{pmatrix} = \begin{pmatrix} \Delta_1 & 0 \\ 0 & \Delta_2 \end{pmatrix} \begin{pmatrix} v \\ w \end{pmatrix}.$$

Hence,

$$W^T a_n + w \bar{\alpha} = \Delta_2 w. \quad (7.8)$$

Because of the Householder transformation we know that  $q = a_n / \|a_n\|$ , hence

$$W^T q = (\Delta_2 - \bar{\alpha} I) \frac{w}{\|a_n\|}. \quad (7.9)$$

Therefore,

$$\begin{aligned} C_2 &= [W^T q, w] \\ &= \left[ (\Delta_2 - \bar{\alpha} I) \frac{w}{\|a_n\|}, w \right]. \end{aligned}$$

This means that if  $\Delta_2 w$  lies almost in the same direction as  $w$  that  $C_2$  is almost singular. If the two largest eigenvalues  $\lambda_{2,1} \approx \lambda_{2,2}$  this is the case. Note however that when  $\lambda_{2,1} \approx -\lambda_{2,2}$ , the subspace iteration converges immediately, because they are both extreme and immediately located by the Lanczos' procedure.

**8. Numerical experiments.** In the first experiment we want to illustrate the Lanczos' convergence behavior of the new method. Experiments two up to four illustrate the Lanczos' behaviour and the subspace iteration convergence. In Experiment 2 an example is created such that the subspace iteration has a large delay, and is not visible. After a clear delay in Experiment 3 the subspace iteration starts its convergence behaviour. In Experiment 4 the example is created in such a way, that there is no delay in the convergence behaviour of the subspace iteration. In Experiment 5 a non graded matrix with a large condition number ( $\approx 10^{40}$ ) is transformed accurately into a semiseparable matrix. Experiment 6 shows the rank revealing properties of the reduction algorithm.

In each experiment, we obtain a symmetric matrix  $A$  by transforming the diagonal matrix  $\Delta$  containing the prescribed eigenvalues by an orthogonal similarity transformation  $A = Q^T \Delta Q$ . The orthogonal matrix used is taken as the  $Q$ -factor in the  $QR$  factorization of a random matrix, built by the Matlab<sup>2</sup> command  $rand(n)$  where  $n$  is the dimension of the matrix.

**8.1. Experiment 1.** We choose the eigenvalues as  $\lambda_i = i$  for  $i = 1, 2, \dots, 200$ . (These are equidistant eigenvalues.) In Figure 8.1 on the  $y$ -axis the eigenvalues are located. In each step of the algorithm ( $x$ -axis), a cross is placed if a Ritz value approximates a real eigenvalue up to 8 correct digits. This behaviour is the same as the one described in [13, Section 4.2, fig. 4.1], for equal spaced eigenvalues.

<sup>2</sup>Matlab is a registered trademark of the Mathworks Inc.

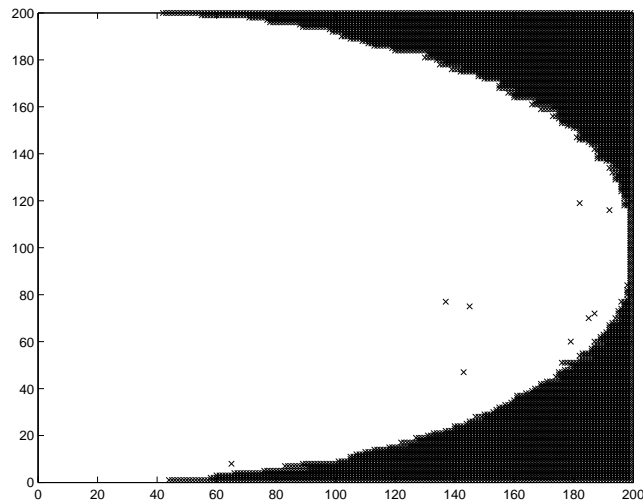


FIGURE 8.1. Equal spaced eigenvalues 1 : 200

**8.2. Experiment 2.** We choose two clusters of equidistant eigenvalues, namely  $[1 : 100, 1000 : 1099]$ , each cluster has the same number of eigenvalues. The following convergence behaviour of the Ritz values is computed (see Figure 8.2 left). Note however that the gap between the two intervals, does not appear in the following figure. For each  $i = 1 : 199$  the norm of the block  $S(i + 1 : n, 1 : i)$  of the resulting semiseparable matrix is plotted. One would expect a small value for  $i = 100$ , because of the subspace iteration, but as explained in Section 7 this is not the case (see Figure 8.2 right).

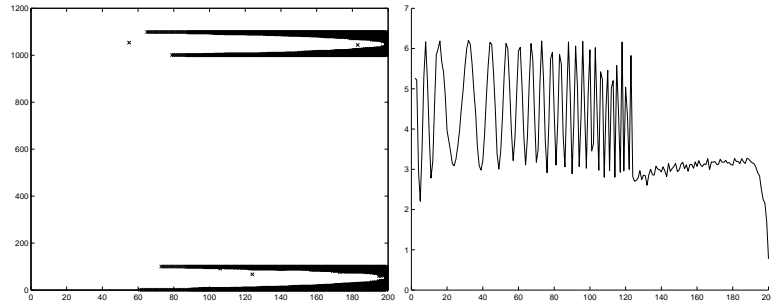


FIGURE 8.2. Equal spaced eigenvalues in two clusters 1 : 100 and 1000 : 1099

**8.3. Experiment 3.** In the previous experiment there was no sign of the influence of subspace iteration. In this experiment however we will clearly see the effect of the subspace iteration. For this example again two clusters of eigenvalues were chosen  $[1 : 100]$  and  $[1000 : 1099]$ , one expects a clear view of the convergence of the subspace iteration in this case. Because the Lanczos-Ritz values approximate the extreme eigenvalues, it will at least take 20 steps before the 10 dominant eigenvalues are approximated. After these steps one can expect to see the convergence of the subspace iteration. The first figure (left of Figure 8.3) shows for each step  $j = 1, 2, \dots, n - 1$  in the algorithm the norms of the blocks  $S(i : n, 1 : i - 1)$  for  $i = n - j : n$ . The second figure (right of Figure 8.3) is constructed in an analogous way as in Experiment 8.2. In Figure 8.4 it can be seen in which step the Ritz values approximate

the most extreme eigenvalues well enough, this is also the point from which the convergence behaviour starts in Figure 8.3.

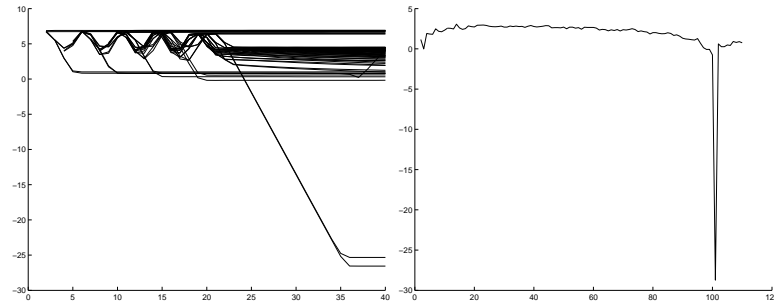


FIGURE 8.3. Equal spaced eigenvalues in two clusters 1 : 100 and 1000 : 1009

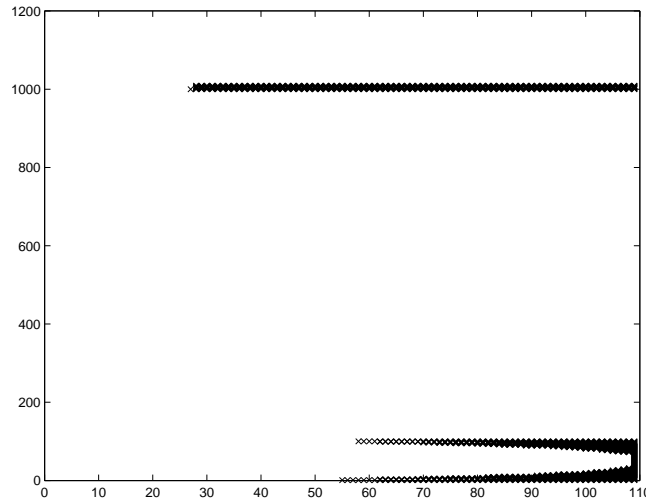
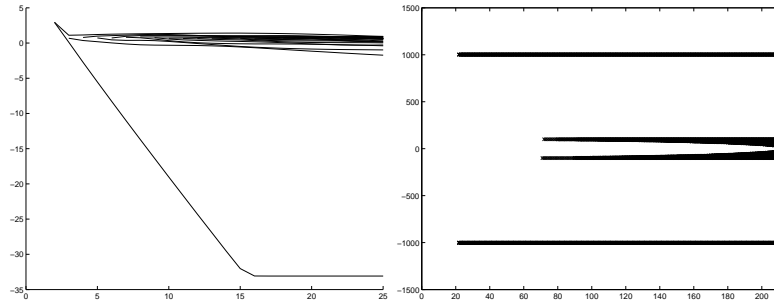


FIGURE 8.4. Lanczos behaviour of equal spaced eigenvalues in two clusters 1 : 100 and 1000 : 109

It is clearly seen in Figure 8.3 that the subspace iteration starts with a small delay (as soon as the Lanczos-Ritz values approximate the eigenvalues well enough the convergence behaviour starts).

**8.4. Experiment 4.** The previous experiment showed that the subspace iteration started working after a delay. In this experiment the largest eigenvalues in absolute value, have opposite signs, such that they will be located fast by the Lanczos' algorithm and therefore the subspace iteration convergence will show up without a delay. The eigenvalues are located in three clusters  $[-1004 : -1000, -100 : 100, 1000 : 1004]$ . The Lanczos-Ritz values will converge fast to the extreme eigenvalues, and therefore the subspace iteration convergence will start fast. Figure 8.5 shows the fast convergence after few steps of the iteration and also the Lanczos convergence behaviour.

**8.5. Experiment 5.** The following problems are taken from [4]. We consider some of the eigenvalue problems which the traditional  $QR$  algorithm cannot solve. We consider the

FIGURE 8.5. Equal spaced eigenvalues in three clusters  $[-1004 : 1000, -100 : 100, 1000 : 1004]$ 

following matrix:  $A = DPD$ , where  $D = \text{diag}(10^{20}, 10^{10}, 1)$ ,

$$A = \begin{pmatrix} 10^{40} & 10^{29} & 10^{19} \\ 10^{29} & 10^{20} & 10^9 \\ 10^{19} & 10^9 & 1 \end{pmatrix} \text{ and } P = \begin{pmatrix} 1 & 0.1 & 0.1 \\ 0.1 & 1 & 0.1 \\ 0.1 & 0.1 & 1 \end{pmatrix}.$$

The eigenvalues of the matrix  $A$  are the following:

$$\Lambda = [1.0000000000000000 \cdot 10^{40}, 9.9000000000000000 \cdot 10^{19}, 9.818181818181818 \cdot 10^{-1}].$$

The eigenvalues computed by the routine  $\text{eig}(\cdot)$  in Matlab gives the following results:

$$[-3.85544 \cdot 10^{23}, 9.90002 \cdot 10^{-01}, 1.0000000000000000 \cdot 10^{40}].$$

One can see that the eigenvalue solver of Matlab, was only able to calculate one eigenvalue correctly. When we reduce the matrix  $A$  to semiseparable form we get the following matrix:

$$\begin{pmatrix} 1.1880000000000000 \cdot 10^2 & 1.0800000000000000 \cdot 10^{11} & 9.7200000000000001 \\ 1.0800000000000000 \cdot 10^{11} & 9.9000000000000000 \cdot 10^{19} & 8.9100000000000000 \cdot 10^9 \\ 9.7200000000000001 & 8.9100000000000000 \cdot 10^9 & 1.0000000000000000 \cdot 10^{40} \end{pmatrix}.$$

One eigenvalue already converged, and computing the eigenvalues of the upper left 2 by 2 block, we get the following eigenvalues:

$$[9.818181818181789 \cdot 10^{-1}, 9.9000000000000000 \cdot 10^{19}].$$

So all the eigenvalues can be computed when first using the reduction to semiseparable form, and this up to at least 14 correct digits.

Calculating the eigenvalues of the matrix  $JAJ$  with

$$J = \begin{pmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{pmatrix},$$

both algorithms give accurate results. The eigenvalue solver of Matlab gives the following result:

$$[9.818181818181818 \cdot 10^{-1}, 9.9000000000000000 \cdot 10^{19}, 1.0000000000000000 \cdot 10^{40}].$$

and the results based on the construction of the semiseparable matrix are exactly the same up to sixteen digits.

**9. Experiment 6: Symmetric Rank Revealing.** This example shows that, in many cases, the factorization described in § 3 can be used as a symmetric rank revealing factorization. A detailed analysis of the proposed factorization as rank revealing factorization can be found in [14].

Let  $A \in \mathbb{R}^{n \times n}$  and let

$$A = V\Lambda V^T = \sum_{i=1}^n \lambda_i v_i v_i^T,$$

$|\lambda_1| = \sigma_1 \geq |\lambda_2| = \sigma_2 \geq \dots \geq |\lambda_n| = \sigma_n \geq 0$ , be its eigenvalue decomposition. The following decomposition

$$A = V_S S V_S^T \text{ with } S = \begin{pmatrix} S_{11} & S_{12} \\ S_{12}^T & S_{22} \end{pmatrix},$$

where  $S_{11} \in \mathbb{R}^{k \times k}$ ,  $S_{12} \in \mathbb{R}^{k \times (n-k)}$ , and  $S_{22} \in \mathbb{R}^{(n-k) \times (n-k)}$ , is said to be a *symmetric rank revealing decomposition* [7, 11] if

$$\text{cond}(S_{11}) \simeq \sigma_1 / \sigma_k, \quad \|S_{12}\|_F + \|S_{22}\|_F \simeq \sigma_{k+1}^2 + \dots + \sigma_n^2.$$

We compare the behaviour of the algorithm described in § 3 (denoted by SSRR<sup>3</sup>: SemiSeparable Rank Revealing) with those of the algorithms ULV and URV, considered among the best symmetric rank revealing algorithms available in the literature for semidefinite and indefinite matrices, respectively, (see [8, 10, 7, 11] for the details).

For the semidefinite case, random test matrices of size  $n = 64, 128, 256$  (100 matrices of each size), each with  $n/2$  eigenvalues geometrically distributed between 1 and  $10^{-4}$ , and the remaining eigenvalues geometrically distributed between  $10^{-7}$  and  $10^{-10}$ , such that the numerical rank with respect to the threshold  $\tau = 10^{-5}$  is  $k = n/2$ , have been computed. In Fig. 9.1 the distribution of the singular values in real scale (left) and logarithmic scale (right) is shown. For the indefinite case, the same matrices have been generated, choosing the signs of the eigenvalues to alternate.

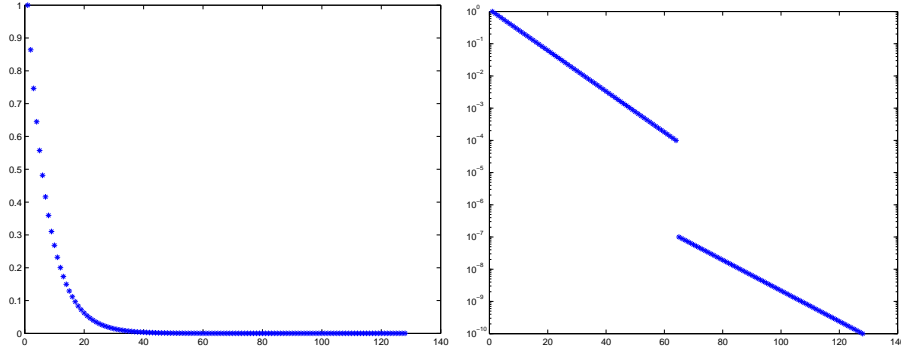


FIGURE 9.1. Distribution of the singular values in real scale (left) and logarithmic scale (right) of the matrices considered for the rank revealing example

For each sample of 100 matrices, the means and the maxima of the spectral norms of the submatrices  $S_{12}$  and  $S_{22}$  have been computed. The results are shown in Table 9.1. It can be

<sup>3</sup>The semiseparable matrix computed by the reduction described in § 3 is transformed into a symmetric similar one by the multiplication of the “exchange” matrix.

noticed that the results obtained by the SSRR algorithm are similar to those obtained by the best available algorithms in the literature for these kind of problems.

A more elaborated implementation of the SSRR algorithm for the symmetric rank revealing factorization is considered in [14].

**10. Conclusions and future research.** In this paper, we have designed a new algorithm to transform any symmetric matrix into a similar semiseparable one by orthogonal similarity transformations. We have shown that during execution of the algorithm semiseparable matrices are generated whose eigenvalues are nothing else but the Ritz values obtained by the Lanczos' process. Moreover, the algorithm can be interpreted as a nested subspace iteration method. We have illustrated the properties of the algorithm by several numerical examples.

When a Ritz value has converged to an eigenvalue of the original matrix, we can apply deflation. Also when the nested subspace iteration has created a diagonal block whose corresponding rank 1 non-diagonal block is small in norm, this part of the semiseparable matrix can be cut off and its eigenvalues can be computed separately.

We are preparing a paper [19] that describes an implicit  $QR$  algorithm for semiseparable matrices. This algorithm can be used to compute the eigenvalues of the resulting matrix as well as of the subblocks which are cut off by the nested subspace iteration.

When we drop the link to subspace iteration, it is clear how to obtain other sequences of structured matrices whose eigenvalues are the Lanczos-Ritz values. These details will be explained elsewhere. This could lead to a Lanczos-type algorithm which does not need (partial) reorthogonalization to improve its stability.

The algorithm provides an efficient method to perform rank-revealing as will be described in [14].

#### REFERENCES

- [1] M. Braun, S.A. Sofianos, D.G. Papageorgiou, and I.E. Lagaris. An efficient Chebyshev Lanczos method for obtaining eigensolutions of the Schrödinger equation on a grid. *Journal of Computational Physics*, 126:315–327, 1996.
- [2] S. Chandrasekaran and M. Gu. Fast and stable eigendecomposition of symmetric banded plus semi-separable matrices. *Linear Algebra and Its Applications*, 313:107–114, 2000.
- [3] S. Chandrasekaran and M. Gu. A divide and conquer algorithm for the eigendecomposition of symmetric block-diagonal plus semi-separable matrices. *submitted for publication*, 2001.
- [4] J. Demmel and K. Veselić. Jacobi's method is more accurate than  $QR$ . *SIAM Journal on Matrix Analysis and its Applications*, 13(4):1204–1245, 1992.
- [5] J. W. Demmel. *Applied numerical linear algebra*. SIAM, 1997.
- [6] D. Fasino, N. Mastronardi, and M. Van Barel. Fast and stable algorithms for reducing diagonal plus semiseparable matrices to tridiagonal and bidiagonal form. In V. Olshevsky, editor, *Fast Algorithms for Structured Matrices: Theory and Applications*, volume 323 of *Contemporary Mathematics Series*. AMS, SIAM, 2003. In press, 14 pages.
- [7] R.D. Fierro and P.C. Hansen. Truncated VSV solutions to symmetric rank-deficient problems. *BIT*, 42(3):531–540.
- [8] R.D. Fierro, P.C. Hansen, and P.S.K. Hansen. UTV tools: Matlab templates for rank-revealing UTV decompositions. *Numer. Algo.*, 20:165–194, 1999.
- [9] G. H. Golub and C. F. Van Loan. *Matrix Computations, third edition*. The Johns Hopkins University Press, 1996.
- [10] P.C. Hansen. SVS tools. <http://www.imm.dtu.dk/~pch/UTV/utv.html>.
- [11] P.C. Hansen and P.Y. Yalamov. Computing symmetric rank-revealing decompositions via triangular factorization. *SIAM J. Matrix Anal. Appl.*, 23(2):443–458, 2001.
- [12] I.T. Jolliffe. *Principal component analysis*. Springer verlag, 1986.
- [13] A. B. J. Kuijlaars. Which eigenvalues are found by the Lanczos method. *SIAM Journal on Matrix Analysis and its Applications*, 22(1):306–321, 2000.
- [14] N. Mastronardi, M. Van Barel, R. Vandebril, and L. Elden. A lanczos subspace iteration method for the symmetric rank revealing factorization. In preparation, 2003.

- [15] N. Mastronardi, E. Van Camp, and M. Van Barel. Divide and conquer type algorithms for computing the eigendecomposition of diagonal plus semiseparable matrices. Technical Report 7 (5/2003), Istituto per le Applicazioni del Calcolo "M. Picone", Consiglio Nazionale delle Ricerche, Rome, Italy, 2003.
- [16] G. W. Stewart. *Matrix Algorithms, Vol II Eigensystems*. SIAM, 1999.
- [17] L. N. Trefethen and D. Bau. *Numerical Linear Algebra*. SIAM, 1997.
- [18] E. Van Camp, N. Mastronardi, and M. Van Barel. A fast QR solver for diagonal plus semi separable linear systems. Technical Report 7 (5/2003), Istituto per le Applicazioni del Calcolo "M. Picone", Consiglio Nazionale delle Ricerche, Bari, Italy, 2002.
- [19] R. Vandebril, M. Van Barel, and N. Mastronardi. An implicit  $QR$  algorithm for semiseparable matrices. In preparation, 2003.
- [20] R. Vandebril, M. Van Barel, and N. Mastronardi. A note on the representation of semiseparable matrices. In preparation, 2003.
- [21] M.E. Wall, P.A. Dyck, and T.S. Brettin. Svdman - singular value decomposition analysis of microarray data. *Bioinformatics*, 17(6):566–568, 2001.
- [22] M.E. Wall, A. Rechtsteiner, and L.M. Rocha. Singular value decomposition and principal component analysis. *A practical approach to microarray data analysis*, 2003. D.P. Berrar, W. Dubitzky, M. Granzow, eds, kluwer.
- [23] D. S. Watkins. QR like algorithms an overview of convergence theory and practice. *The Mathematics of Numerical Analysis, Lectures in Applied Math.*, 32:879–893, 1996.
- [24] D. S. Watkins and L. Elsner. Convergence of algorithms of decomposition type for the eigenvalue problem. *Linear Algebra and Its Applications*, 143:19–47, 1991.
- [25] M.K.S. Yeung and W.L. Ruzzo. Principal component analysis for clustering gene expression data. *Bioinformatics*, 17(9):763–774, 2001.
- [26] M.K.S. Yeung, J. Tegnér, and J.J. Collins, editors. *Reverse engineering gene networks using singular value decomposition and robust regression*, volume 99 of *Proceedings National Academy of Sciences*, 2002.

<b>Semidefinite case</b>			
$n = 64$			
		$\ S_{12}\ _2$	$\ S_{22}\ _2$
ULV	mean	$4.791639932718456 \cdot 10^{-011}$	$1.000000000124228 \cdot 10^{-007}$
	max	$8.934020135671359 \cdot 10^{-011}$	$1.000000000666147 \cdot 10^{-007}$
SSRR	mean	$1.837697901913820 \cdot 10^{-098}$	$9.99999999853075 \cdot 10^{-008}$
	max	$7.498628451264783 \cdot 10^{-097}$	$1.000000000399622 \cdot 10^{-007}$
$n = 128$			
		$\ S_{12}\ _2$	$\ S_{22}\ _2$
ULV	mean	$7.161049583643726 \cdot 10^{-011}$	$1.000000000234679 \cdot 10^{-007}$
	max	$1.131352774730907 \cdot 10^{-010}$	$1.000000000979856 \cdot 10^{-007}$
SSRR	mean	$1.423201103576507 \cdot 10^{-192}$	$9.99999999966758 \cdot 10^{-008}$
	max	$2.804793551904090 \cdot 10^{-190}$	$1.000000000423946 \cdot 10^{-007}$
$n = 256$			
		$\ S_{12}\ _2$	$\ S_{22}\ _2$
ULV	mean	$1.098171024990544 \cdot 10^{-010}$	$1.000000000384130 \cdot 10^{-007}$
	max	$1.708052274908808 \cdot 10^{-010}$	$1.000000001100753 \cdot 10^{-007}$
SSRR	mean	0	$9.99999999776952 \cdot 10^{-008}$
	max	0	$1.000000000333670 \cdot 10^{-007}$
<b>Indefinite case</b>			
$n = 64$			
		$\ S_{12}\ _2$	$\ S_{22}\ _2$
URV	mean	$1.872923469640583 \cdot 10^{-012}$	$1.00000000022273 \cdot 10^{-007}$
	max	$1.092839179147104 \cdot 10^{-011}$	$1.000000000414043 \cdot 10^{-007}$
SSRR	mean	$2.875906467257399 \cdot 10^{-100}$	$1.00000000043458 \cdot 10^{-007}$
	max	$2.437723364247734 \cdot 10^{-099}$	$1.000000000375846 \cdot 10^{-007}$
$n = 128$			
		$\ S_{12}\ _2$	$\ S_{22}\ _2$
URV	mean	$3.141978051167186 \cdot 10^{-012}$	$9.99999999802293 \cdot 10^{-008}$
	max	$4.529790334779005 \cdot 10^{-011}$	$1.000000000222440 \cdot 10^{-007}$
SSRR	mean	$1.234226386891272 \cdot 10^{-195}$	$9.99999999931803 \cdot 10^{-008}$
	max	$9.741325960957623 \cdot 10^{-195}$	$1.000000000279758 \cdot 10^{-007}$
$n = 256$			
		$\ S_{12}\ _2$	$\ S_{22}\ _2$
URV	mean	$7.075128158664781 \cdot 10^{-012}$	$1.00000000016904 \cdot 10^{-007}$
	max	$1.961629685655226 \cdot 10^{-010}$	$1.000000001181266 \cdot 10^{-007}$
SSRR	mean	0	$1.00000000007996 \cdot 10^{-007}$
	max	0	$1.000000000413693 \cdot 10^{-007}$

TABLE 9.1

Comparison between the spectral norms of submatrices  $S_{12}$  and  $S_{22}$  computed by the ULV and SSRR algorithms (semidefinite case) and by the URV and SSRR algorithms (indefinite case)