A LANCZOS-TYPE METHOD FOR MULTIPLE STARTING VECTORS*

J. I. ALIAGA, D. L. BOLEY, R. W. FREUND, AND V. HERNÁNDEZ

Abstract. Given a square matrix and single right and left starting vectors, the classical nonsymmetric Lanczos process generates two sequences of biorthogonal basis vectors for the right and left Krylov subspaces induced by the given matrix and vectors. In this paper, we propose a Lanczos-type algorithm that extends the classical Lanczos process for single starting vectors to multiple starting vectors. Given a square matrix and two blocks of right and left starting vectors, the algorithm generates two sequences of biorthogonal basis vectors for the right and left block Krylov subspaces induced by the given data. The algorithm can handle the most general case of right and left starting blocks of arbitrary sizes, while all previously proposed extensions of the Lanczos process are restricted to right and left starting blocks of identical sizes. Other features of our algorithm include a built-in deflation procedure to detect and delete linearly dependent vectors in the block Krylov sequences, and the option to employ look-ahead to remedy the potential breakdowns that may occur in nonsymmetric Lanczos-type methods.

1. Introduction

1.1. The Lanczos process for single starting vectors. Given a square matrix $\mathbf{A} \in \mathbb{C}^{N \times N}$ and two nonzero starting vectors $\mathbf{r}, \mathbf{l} \in \mathbb{C}^N$, the classical nonsymmetric Lanczos process [27] is an algorithm that uses three-term recurrences to generate two sequences of biorthogonal basis vectors for the right Krylov subspace induced by $\mathbf{A}$ and $\mathbf{r}$, and the left Krylov subspace induced by $\mathbf{A}^{\mathrm{T}}$ and $\mathbf{l}$. Furthermore, the coefficients of the three-term recurrences define a sequence of $n \times n$ matrices $\mathbf{T}_n^{(s)}$, $n = 1, 2, \ldots$, that constitute approximations to the given matrix $\mathbf{A}$. More precisely, the n-th Lanczos matrix $\mathbf{T}_n^{(s)}$ represents the oblique projection of $\mathbf{A}$ onto the n-th right Krylov subspace and orthogonal to the n-the left Krylov subspace.

In the Lanczos process in its original form [27], breakdowns or near-breakdowns—triggered by division by zero or a number close to zero—cannot be excluded. Fortunately, the problem of potential breakdowns and near-breakdowns can be remedied by incorporating so-called look-ahead techniques into the Lanczos process. The possibility of such a remedy was first observed by Taylor [38] and Parlett, Taylor, and Liu [33], who also coined the term "look-ahead". Since then, there has been extensive research activities in this area, and as a result, the look-ahead Lanczos

1991 *Mathematics Subject Classification.* Primary 65F10, 65F15; Secondary 65F25, 65F30.

Key words and phrases. Lanczos algorithm, nonsymmetric matrix, block Krylov subspaces, biorthogonalization, oblique projection, deflation, breakdown, look-ahead.

*Numerical Analysis Manuscript No. 98–3–05, Bell Laboratories, Murray Hill, New Jersey, September 1998. Available on WWW at `http://cm.bell-labs.com/cs/doc/98`.

The first and the last author were supported in part by the European ESPRIT III Basic Research Project GEPPCOM #9072. The second author was supported in part by the U.S. NSF Grants #CCR-9405380 and #CCR-9628786.

process is now well understood; see, e.g., [6, 14, 17, 24] and the references given therein. The basic principle of the look-ahead Lanczos process is to continue the algorithm in the event of a breakdown or near-breakdown by relaxing the vector-wise biorthogonality of the Lanczos basis vectors to a cluster-wise biorthogonality and by resorting, for the next few iteration steps, to recurrences that are slightly longer than the three-term recurrences in the classical algorithm.

When applied to large $N \times N$ matrices $\mathbf{A}$, the $n \times n$ Lanczos matrices $\mathbf{T}_n^{(s)}$ are often very good approximations to $\mathbf{A}$ already for $n \ll N$, and this makes the Lanczos process a powerful tool for various computational tasks for large matrices $\mathbf{A}$. We now briefly mention three such applications in large-scale matrix computations.

The first application is the computation of approximate eigenvalues of $\mathbf{A}$. Starting with arbitrary (for example, random) nonzero vectors $\mathbf{r}$ and $\mathbf{l}$, one runs the Lanczos process for n steps to obtain $\mathbf{T}_n^{(s)}$. The eigenvalues of $\mathbf{T}_n^{(s)}$ are then used as approximate eigenvalues of the matrix $\mathbf{A}$; see, e.g., [11].

The second application is the solution of large systems of linear equations,

$$\mathbf{A}\,\mathbf{x} = \mathbf{b}. \tag{1.1}$$

The biconjugate gradient (BCG) algorithm [28] and the quasi-minimal residual (QMR) algorithm [19, 20] are iterative methods that generate approximations $\mathbf{x}_n$ for the solution of (1.1), starting from an arbitrary initial guess $\mathbf{x}_0$ and an arbitrary nonzero left vector $\mathbf{l}$. Both algorithms are intimately connected to the Lanczos process applied to the matrix $\mathbf{A}$, with starting vectors $\mathbf{r} := \mathbf{b} - \mathbf{A}\,\mathbf{x}_0$ and $\mathbf{l}$. For BCG, the n-th iterate $\mathbf{x}_n$ is defined by a Galerkin-type condition that is mathematically equivalent to solving a small $n \times n$ linear system with coefficient matrix $\mathbf{T}_n^{(s)}$, instead of the large $N \times N$ system (1.1). For QMR, $\mathbf{x}_n$ is defined by a quasi-minimization of the residual norm that is mathematically equivalent to solving a small $(n+1) \times n$ least-squares problem whose matrix is $\mathbf{T}_n^{(s)}$, extended by one more row.

A third application is Padé approximation of transfer functions describing large single-input single-output time-invariant linear dynamical systems. Such transfer functions are rational functions $H : \mathbb{C} \mapsto \mathbb{C} \cup \{\infty\}$ of the form

$$H(s) = \mathbf{l}^{\mathrm{T}} \left(\mathbf{I} - s\,\mathbf{A}\right)^{-1} \mathbf{r}, \tag{1.2}$$

where $\mathbf{A} \in \mathbb{C}^{N \times N}$, $\mathbf{r}$, $\mathbf{l} \in \mathbb{C}^N$ are given, and $\mathbf{I}$ denotes the $N \times N$ identity matrix. An n-th Padé approximant, H_n, of the function (1.2) is defined as a rational function with numerator and denominator degree at most $n-1$ and n, respectively, such that the Taylor expansions of H_n and H about $s = 0$ match in as many leading Taylor coefficients as possible. The Padé approximant H_n can be directly obtained from the Lanczos process applied to $\mathbf{A}$, $\mathbf{r}$, and $\mathbf{l}$; see, e.g., [12, 22, 23]. Indeed, assuming for simplicity that no look-ahead steps occur in the Lanczos algorithm, the n-th Padé approximant is simply given by

$$H_n(s) = \mathbf{l}^{\mathrm{T}}\mathbf{r}\; \mathbf{e}_1^{\mathrm{T}} \left(\mathbf{I}_n - s\,\mathbf{T}_n^{(s)}\right)^{-1} \mathbf{e}_1, \tag{1.3}$$

where $\mathbf{I}_n$ is the $n \times n$ identity matrix and $\mathbf{e}_1$ is the first unit vector of length n. A formula similar to (1.3) holds for the case that look-ahead steps do occur.

1.2. **Handling multiple starting vectors.** All three applications described in §1.1 have extensions that involve multiple starting vectors.

For eigenvalue computations of a matrix $\mathbf{A}$ with multiple or clusters of eigenvalues, it is usually preferable to employ a Lanczos-type method that iterates on blocks of, say m, vectors, rather than on single vectors; see, e.g., [10, 21, 35]. Such a procedure then involves m right and m left starting vectors.

Some applications require the repeated solution of linear systems (1.1) with the same matrix $\mathbf{A}$, but different right-hand sides, $\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_m$, that are all available simultaneously. These m systems can be summarized in block form:

$$\mathbf{A}\,\mathbf{X} = \mathbf{B}, \quad \text{where} \quad \mathbf{B} := \begin{bmatrix} \mathbf{b}_1 & \mathbf{b}_2 & \cdots & \mathbf{b}_m \end{bmatrix}. \tag{1.4}$$

Applying a block version of an iterative method to (1.4) is often significantly more efficient than solving the m linear systems individually; see, e.g., [18, 29, 30, 31]. Block versions of BCG and QMR involve a block of m right starting vectors, namely $\mathbf{R} := \mathbf{B} - \mathbf{A}\,\mathbf{X}_0$, where $\mathbf{X}_0 \in \mathbb{C}^{N \times m}$ is an arbitrary initial guess for (1.4).

Time-invariant linear dynamical systems with m inputs and p outputs are characterized by matrix-valued transfer functions $\mathbf{H}\colon \mathbb{C} \mapsto (\mathbb{C} \cup \{\infty\})^{p \times m}$ of the form

$$\mathbf{H}(s) = \mathbf{L}^{\mathrm{T}}\,(\mathbf{I} - s\,\mathbf{A})^{-1}\,\mathbf{R}, \tag{1.5}$$

where $\mathbf{A} \in \mathbb{C}^{N \times N}$, $\mathbf{R} \in \mathbb{C}^{N \times m}$, $\mathbf{L} \in \mathbb{C}^{N \times p}$. Such transfer functions arise in control theory [5, 36, 37] and in circuit simulation [13]. For functions (1.5), $\mathbf{H}$, one can again define n-th Padé approximants, $\mathbf{H}_n$, which are now also matrix-valued functions, i.e., $\mathbf{H}_n : \mathbb{C} \mapsto (\mathbb{C} \cup \{\infty\})^{p \times m}$. Extending the Lanczos-Padé connection (1.3) for the single-input single-output case, $m = p = 1$, to the general m-input p-output case, m, $p \geq 1$, requires a Lanczos-type process that can handle m right and p left starting vectors, namely the columns of $\mathbf{R}$ and $\mathbf{L}$, respectively.

These three applications clearly show the need for a Lanczos-type algorithm for multiple starting vectors. Such an algorithm should be an extension of the classical Lanczos process for single starting vectors, and generate two sequences of basis vectors for the right and left block Krylov subspaces induced by the given matrix and the blocks of right and left starting vectors. In order to obtain a robust algorithm, the following three key difficulties need to be resolved.

(i) The algorithm needs to include a *deflation* procedure in order to detect and delete linearly dependent vectors in the right and left block Krylov subspaces, respectively. This is an issue arising only when using multiple starting vectors. In the Lanczos process for single starting vectors, encountering a linearly dependent right or left vector simply means that the corresponding right or left Krylov subspace is *exhausted*, i.e., it reached its maximal dimension. The algorithm terminates normally in this situation.

(ii) The algorithm needs to be able to handle different block sizes in the right and left block Krylov subspaces. These different block sizes may be due to different sizes of the starting blocks, i.e., $m \neq p$, or due to deflation.

(iii) As in the classical Lanczos algorithm for single starting vectors, it cannot be excluded that breakdowns or near-breakdowns occur. As a result, in general, look-ahead techniques need to be incorporated.

In this paper, we propose a Lanczos-type algorithm that extends the classical Lanczos process for single starting vectors to multiple starting vectors, and that can handle all three difficulties (i)–(iii) listed above. Given a matrix $\mathbf{A} \in \mathbb{C}^{N \times N}$, and right and left starting blocks $\mathbf{R} \in \mathbb{C}^{N \times m}$ and $\mathbf{L} \in \mathbb{C}^{N \times p}$, the algorithm generates two sequences of biorthogonal basis vectors for the right and left block Krylov subspaces induced by the given data. The algorithm includes a simple built-in deflation procedure, and it can handle the most general case of right and left block Krylov subspaces with arbitrary sizes m and p of the starting blocks.

The key property of the algorithm, which allows us to resolve the issues (i)–(iii), is the vector-wise construction of the basis vectors for the block Krylov subspaces. The vector-wise approach appears to originate with Ruhe [35], where it was applied to Hermitian matrices. However, we stress that, for Hermitian matrices, the problem of handling multiple starting vectors is a lot easier for the following two reasons. First, the right and left block Krylov subspaces are (up to complex conjugation if $\mathbf{A}$ is complex) identical, and thus the complication due to different right and left block sizes does not arise. Second, the possibility of breakdowns can be excluded, and hence no look-ahead is needed. We note that, to the best of our knowledge, we seem to be the first to extend Ruhe's vector-wise construction of basis vectors for the Hermitian case to the general case of non-Hermitian matrices.

We remark that early versions of the Lanczos-type algorithm described in this paper had been developed independently by Aliaga, Boley, and Hernández, and by Freund, and were presented by Boley [3] and Freund [16] at the same Oberwolfach meeting in 1994. It was then that we decided to write this joint paper. However, we would like to stress that the algorithm presented in this paper has evolved quite a bit from the versions we had in 1994. We also note that, in his doctoral thesis [1], Aliaga investigated variants of the algorithm that are tailored to parallel computers.

1.3. **Related work on block Lanczos methods.** The problem of extending the Lanczos process for single to multiple starting vectors is, of course, not new, and a number of algorithms have been developed over the years. With the exception of Ruhe's algorithm [35] for the Hermitian case, all previously proposed algorithms use a block-wise construction of block-biorthogonal basis vectors for the underlying block Krylov subspaces. It is easy to see that any such block-wise approach requires all right and left blocks to have the same size. In particular, block Lanczos algorithms are restricted to the special case that $p = m$ and that possible deflation occurs simultaneously in the right and left block Krylov subspaces.

Block Lanczos algorithms for Hermitian matrices were first proposed by Cullum and Donath [9], and Golub and Underwood [21, 39]. Further and more recent work for the Hermitian case is described in [10, 30, 35] and the references given therein. We remark that only the algorithms in [9, 10, 30] and Ruhe's algorithm [35] include a proper deflation procedure.

For non-Hermitian matrices, O'Leary—with her block BCG algorithm [31]—was the first to develop a block Lanczos-type method. A block version of the original three-term Lanczos algorithm [27] was first presented in [25, 26], and a more recent variant was proposed in [4]. As already pointed out above, all these algorithms are restricted to the case $p = m$. Furthermore, none of the existing block Lanczos-type methods for non-Hermitian matrices has a built-in deflation procedure, nor are there any look-ahead variants to remedy possible breakdowns.

1.4. **Outline.** The remainder of this article is organized as follows. In §2, we introduce our notion of block Krylov subspaces associated with multiple starting vectors. In §3, we state some basic properties of the Lanczos vectors. In §4, we describe the construction of the Lanczos vectors. In §5, we present a complete statement of our Lanczos-type algorithm for multiple starting vectors and discuss a few implementation issues. In §6, we establish some properties of the algorithm. In §7, we make some concluding remarks.

1.5. **Notation.** Throughout this article, all vectors and matrices are allowed to have real or complex entries. We use boldface letters to denote vectors and matrices. As usual, $\overline{\mathbf{M}} = \begin{bmatrix}\overline{m_{jk}}\end{bmatrix}$, $\mathbf{M}^{\mathrm{T}} = \begin{bmatrix}m_{kj}\end{bmatrix}$, and $\mathbf{M}^{\mathrm{H}} = \overline{\mathbf{M}}^{\mathrm{T}} = \begin{bmatrix}\overline{m_{kj}}\end{bmatrix}$ denote the complex conjugate, transpose, and the conjugate transpose, respectively, of the matrix $\mathbf{M} = \begin{bmatrix}m_{jk}\end{bmatrix}$. The vector norm $\|\mathbf{x}\| := \sqrt{\mathbf{x}^{\mathrm{H}}\mathbf{x}}$ is always the Euclidean norm, and $\|\mathbf{M}\| := \max_{\|\mathbf{x}\|=1} \|\mathbf{M}\,\mathbf{x}\|$ is the corresponding matrix norm. The sets of real and complex numbers are denoted by $\mathbb{R}$ and $\mathbb{C}$, respectively, and $\mathbb{N}$ is the set of positive integers.

2. Block Krylov subspaces

From now on, it is always assumed that $\mathbf{A} \in \mathbb{C}^{N\times N}$ is a given $N \times N$ matrix,

$$\mathbf{R} = \begin{bmatrix}\mathbf{r}_1 & \mathbf{r}_2 & \cdots & \mathbf{r}_m\end{bmatrix} \in \mathbb{C}^{N\times m}$$

is a given matrix of m *right starting vectors*, $\mathbf{r}_1, \mathbf{r}_2, \ldots, \mathbf{r}_m$, and

$$\mathbf{L} = \begin{bmatrix}\mathbf{l}_1 & \mathbf{l}_2 & \cdots & \mathbf{l}_p\end{bmatrix} \in \mathbb{C}^{N\times p}$$

is a given matrix of p *left starting vectors*, $\mathbf{l}_1, \mathbf{l}_2, \ldots, \mathbf{l}_p$. We stress that $m \geq 1$ and $p \geq 1$ are *arbitrary* integers, and in particular, m and p need not be identical.

In this section, we introduce our notion of block Krylov subspaces induced by the data $\mathbf{A}$, $\mathbf{R}$, and $\mathbf{L}$. We start by defining the *right* and *left block Krylov matrix*

$$\begin{aligned} \mathbf{K}(\mathbf{A},\mathbf{R}) &:= \begin{bmatrix}\mathbf{R} & \mathbf{A}\,\mathbf{R} & \mathbf{A}^2\,\mathbf{R} & \cdots & \mathbf{A}^{N-1}\,\mathbf{R}\end{bmatrix} \\ \text{and}\quad \mathbf{K}(\mathbf{A}^{\mathrm{T}},\mathbf{L}) &:= \begin{bmatrix}\mathbf{L} & \mathbf{A}^{\mathrm{T}}\,\mathbf{L} & (\mathbf{A}^{\mathrm{T}})^2\,\mathbf{L} & \cdots & (\mathbf{A}^{\mathrm{T}})^{N-1}\,\mathbf{L}\end{bmatrix}, \end{aligned} \tag{2.1}$$

respectively. The Lanczos-type algorithm constructs biorthogonal basis vectors for the ascending n-dimensional subspaces, $n = 1, 2, \ldots$, spanned by the first n linearly independent columns of the matrices $\mathbf{K}(\mathbf{A},\mathbf{R})$ and $\mathbf{K}(\mathbf{A}^{\mathrm{T}},\mathbf{L})$, respectively. To properly define these subspaces, we need to delete the linearly dependent columns in (2.1). This is done by scanning the columns of each of the matrices $\mathbf{K}(\mathbf{A},\mathbf{R})$ and $\mathbf{K}(\mathbf{A}^{\mathrm{T}},\mathbf{L})$ from left to right and deleting each column that is linearly dependent on earlier columns within the same matrix. This process of deleting linearly dependent columns is referred to as *exact deflation* in the sequel. Applying exact deflation to (2.1), we obtain the *deflated* right and left block Krylov matrices $\mathbf{K}^{\mathrm{dl}}(\mathbf{A},\mathbf{R})$ and $\mathbf{K}^{\mathrm{dl}}(\mathbf{A}^{\mathrm{T}},\mathbf{L})$, respectively. By the structure (2.1) of $\mathbf{K}(\mathbf{A},\mathbf{R})$, a column $\mathbf{A}^{j-1}\,\mathbf{r}_i$ being linearly dependent on earlier columns implies that all columns $\mathbf{A}^k\,\mathbf{r}_i$, $j \leq k \leq N-1$, are also linearly dependent on earlier columns. An analogous statement holds for the matrix $\mathbf{K}(\mathbf{A}^{\mathrm{T}},\mathbf{L})$ in (2.1). Consequently, the deflated Krylov matrices have the following form:

$$\begin{aligned} \mathbf{K}^{\mathrm{dl}}(\mathbf{A},\mathbf{R}) &= \begin{bmatrix}\mathbf{R}_1 & \mathbf{A}\,\mathbf{R}_2 & \mathbf{A}^2\,\mathbf{R}_3 & \cdots & \mathbf{A}^{j_{\max}-1}\,\mathbf{R}_{j_{\max}}\end{bmatrix}, \\ \mathbf{K}^{\mathrm{dl}}(\mathbf{A}^{\mathrm{T}},\mathbf{L}) &= \begin{bmatrix}\mathbf{L}_1 & \mathbf{A}^{\mathrm{T}}\,\mathbf{L}_2 & (\mathbf{A}^{\mathrm{T}})^2\,\mathbf{L}_3 & \cdots & (\mathbf{A}^{\mathrm{T}})^{k_{\max}-1}\,\mathbf{L}_{k_{\max}}\end{bmatrix}. \end{aligned} \tag{2.2}$$

Here, for each $j = 1, 2, \ldots, j_{\max}$, $\mathbf{R}_j$ is a submatrix of $\mathbf{R}_{j-1}$, with $\mathbf{R}_j \neq \mathbf{R}_{j-1}$ if, and only if, exact deflation occurred within the j-th right Krylov block $\mathbf{A}^{j-1}\,\mathbf{R}$ in (2.1). (For $j = 1$, we set $\mathbf{R}_0 = \mathbf{R}$.) Similarly, for each $k = 1, 2, \ldots, k_{\max}$, $\mathbf{L}_k$ is a submatrix of $\mathbf{L}_{k-1}$, with $\mathbf{L}_k \neq \mathbf{L}_{k-1}$ if, and only if, exact deflation occurred within the k-th left Krylov block $(\mathbf{A}^{\mathrm{T}})^{k-1}\,\mathbf{L}$ in (2.1). (For $k = 1$, we set $\mathbf{L}_0 = \mathbf{L}$.)

We denote by $\mathcal{K}_n(\mathbf{A},\mathbf{R})$ the subspace of $\mathbb{C}^N$ spanned by the first n columns of the deflated right block Krylov matrix $\mathbf{K}^{\mathrm{dl}}(\mathbf{A},\mathbf{R})$ in (2.2). We call $\mathcal{K}_n(\mathbf{A},\mathbf{R})$ the

n-th *right block Krylov subspace* (induced by $\mathbf{A}$ and $\mathbf{R}$). Similarly, the n-th *left block Krylov subspace* (induced by $\mathbf{A}^{\mathrm{T}}$ and $\mathbf{L}$), denoted by $\mathcal{K}_n(\mathbf{A}^{\mathrm{T}}, \mathbf{L})$, is defined as the subspace of $\mathbb{C}^N$ spanned by the first n columns of the matrix $\mathbf{K}^{\mathrm{dl}}(\mathbf{A}^{\mathrm{T}}, \mathbf{L})$ in (2.2). By construction, the columns of each matrix $\mathbf{K}^{\mathrm{dl}}(\mathbf{A}, \mathbf{R})$ and $\mathbf{K}^{\mathrm{dl}}(\mathbf{A}^{\mathrm{T}}, \mathbf{L})$ are linearly independent, and thus both $\mathcal{K}_n(\mathbf{A}, \mathbf{R})$ and $\mathcal{K}_n(\mathbf{A}^{\mathrm{T}}, \mathbf{L})$ are subspaces of dimension n.

Note that in our construction of block Krylov subspaces, we only have used exact deflation. In an actual algorithm for constructing basis vectors for $\mathcal{K}_n(\mathbf{A}, \mathbf{R})$ and $\mathcal{K}_n(\mathbf{A}^{\mathrm{T}}, \mathbf{L})$ in finite-precision arithmetic, one also needs to delete vectors that are in some sense "almost" linearly dependent on earlier vectors. We will refer to the deletion of such almost linearly dependent vectors as *inexact deflation*. Our Lanczos-type algorithm has a simple built-in procedure for both exact and inexact deflation. While inexact deflation is crucial in practice, a concise definition of the corresponding block Krylov subspaces is necessarily quite involved. For the sake of simplicity, in this paper, we use only the notion of block Krylov subspaces based on exact deflation. Finally, in the sequel, "deflation" always means that both exact and inexact deflation are performed.

3. Biorthogonality of the Lanczos vectors

In this section, we formulate some basic properties of the vectors generated by our Lanczos-type algorithm.

3.1. **The Lanczos vectors.** The algorithm generates two sequences of vectors,

$$\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_n, \ldots, \mathbf{v}_{n_{\max}} \quad \text{and} \quad \mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_n, \ldots, \mathbf{w}_{n_{\max}}. \tag{3.1}$$

We will refer to (3.1) as the *right* and *left Lanczos vectors*, respectively.

In the case of exact deflation only, for $n = 1, 2, \ldots, n_{\max}$, the first n vectors in each sequence (3.1) span the n-th right and left block Krylov subspaces:

$$\begin{aligned} \operatorname{span}\{\, \mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_n \,\} &= \mathcal{K}_n(\mathbf{A}, \mathbf{R}), \\ \operatorname{span}\{\, \mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_n \,\} &= \mathcal{K}_n(\mathbf{A}^{\mathrm{T}}, \mathbf{L}). \end{aligned} \tag{3.2}$$

Moreover, the integer $n_{\max}$ in (3.1) is defined as

$$n_{\max} := \min\{\, \operatorname{rank} \mathbf{K}(\mathbf{A}, \mathbf{R}),\ \operatorname{rank} \mathbf{K}(\mathbf{A}^{\mathrm{T}}, \mathbf{L}) \,\}. \tag{3.3}$$

In the presence of inexact deflation, instead of (3.2), we have inclusions of the following form (provable by induction using the equations (4.6) below):

$$\begin{aligned} \operatorname{span}\{\, \mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_n \,\} &\subseteq \mathcal{K}_{n+d_r(n)}(\mathbf{A}, \mathbf{R}), \\ \operatorname{span}\{\, \mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_n \,\} &\subseteq \mathcal{K}_{n+d_l(n)}(\mathbf{A}^{\mathrm{T}}, \mathbf{L}). \end{aligned} \tag{3.4}$$

Here, $d_r(n)$ and $d_l(n)$ depend on the number of inexact deflations that have occurred up to stage n. Moreover, the integer $n_{\max}$ in (3.1) is now given by

$$n_{\max} := \min\{\, \operatorname{rank} \mathbf{K}(\mathbf{A}, \mathbf{R}) - \max_{n \geq 1} d_r(n),\ \operatorname{rank} \mathbf{K}(\mathbf{A}^{\mathrm{T}}, \mathbf{L}) - \max_{n \geq 1} d_l(n) \,\},$$

instead of (3.3). We note that it is also possible to derive a more quantitative version of (3.4). However, for the sake of brevity, such a result is not included in this paper.

As in the classical Lanczos process for single starting vectors, the Lanczos vectors (3.1) are computed in pairs. At pass n of the algorithm, the n-th pair $\mathbf{v}_n$ and

$\mathbf{w}_n$ is built, where $\mathbf{v}_n$ is the vector that advances the right block Krylov subspace, and $\mathbf{w}_n$ advances the left block Krylov subspace. Clearly, this pair-wise construction has to be terminated as soon as one of the two block Krylov subspaces is exhausted. If only one of the subspaces is exhausted, it would be possible to continue the construction of single basis vectors for the non-exhausted block Krylov subspace. However, this is not done in our algorithm, and we simply stop the process as soon as one of the two block Krylov subspaces is exhausted.

3.2. **Vector-wise biorthogonality.** As in the classical Lanczos process, the goal is to construct vectors (3.1) that are *vector-wise biorthogonal*, i.e.,

$$\mathbf{w}_i^{\mathrm{T}}\mathbf{v}_n = \begin{cases} \delta_n & \text{if } i = n, \\ 0 & \text{if } i \neq n, \end{cases} \qquad \text{for all} \quad i, n = 1, 2, \ldots, n_{\max}. \tag{3.5}$$

However, enforcing (3.5) is only possible if $\delta_n \neq 0$ for all $n = 1, 2, \ldots, n_{\max} - 1$. Indeed, constructing biorthogonal vectors (3.1) involves division by δ_n. In the general case, it cannot be excluded that

$$\delta_n = \mathbf{w}_n^{\mathrm{T}}\mathbf{v}_n = 0 \quad \text{for some} \quad n < n_{\max}, \tag{3.6}$$

and thus any algorithm that tries to enforce (3.5) may break down due to division by zero. The event (3.6) will be referred to as an *exact breakdown* of the Lanczos type-algorithm. In finite-precision arithmetic, one also needs to deal with so-called *near-breakdowns* due to division by nonzero numbers

$$\delta_n = \mathbf{w}_n^{\mathrm{T}}\mathbf{v}_n \approx 0, \quad \delta_n \neq 0, \quad \text{for some} \quad n < n_{\max}, \tag{3.7}$$

that are in some sense close to zero.

The key to dealing with possible exact and near-breakdown in the general case is to relax the biorthogonality (3.5) of individual Lanczos vectors to a biorthogonality of clusters of Lanczos vectors.

3.3. **Cluster-wise biorthogonality.** In the general case, the right and left Lanczos vectors (3.1) are grouped into *clusters*

$$\begin{aligned} &\mathbf{V}^{(1)}, \mathbf{V}^{(2)}, \ldots, \mathbf{V}^{(l)}, \ldots, \mathbf{V}^{(l_{\max})} \\ \text{and} \quad &\mathbf{W}^{(1)}, \mathbf{W}^{(2)}, \ldots, \mathbf{W}^{(l)}, \ldots, \mathbf{W}^{(l_{\max})}, \end{aligned} \tag{3.8}$$

respectively. Here, for each l,

$$\begin{aligned} \mathbf{V}^{(l)} &:= \begin{bmatrix} \mathbf{v}_{n_l} & \mathbf{v}_{n_l+1} & \cdots & \mathbf{v}_{n_{l+1}-1} \end{bmatrix} \\ \text{and} \quad \mathbf{W}^{(l)} &:= \begin{bmatrix} \mathbf{w}_{n_l} & \mathbf{w}_{n_l+1} & \cdots & \mathbf{w}_{n_{l+1}-1} \end{bmatrix} \end{aligned} \tag{3.9}$$

are $N \times (n_{l+1} - n_l)$ matrices. Furthermore,

$$n_1 := 1 < n_2 < \cdots < n_l < \cdots < n_{l_{\max}} < n_{l_{\max}+1} := n_{\max} + 1 \tag{3.10}$$

are the *cluster indices*. Note that n_l is just the index of the first vector in the l-th pair of clusters (3.9). The size of the l-th pair of clusters is given by $n_{l+1} - n_l$. We call $\mathbf{V}^{(l)}$ and $\mathbf{W}^{(l)}$ *true look-ahead clusters* if they contain more than one Lanczos vector, i.e., if $n_{l+1} - n_l > 1$. The clusters $\mathbf{V}^{(l)}$ and $\mathbf{W}^{(l)}$ are built as true look-ahead clusters if, and only if, an exact breakdown (3.6) or a near-breakdown (3.7) occurs at step $n = n_l$. Furthermore, the next cluster index n_{l+1} is chosen as the smallest integer bigger than n_l such that the matrices $\mathbf{\Delta}^{(l)} := \left(\mathbf{W}^{(l)}\right)^{\mathrm{T}} \mathbf{V}^{(l)}$ are "sufficiently" nonsingular, except the last matrix, $\mathbf{\Delta}^{(l_{\max})}$, which may be singular.

The Lanczos vectors (3.1) are constructed to be *cluster-wise biorthogonal*, i.e.,

$$\left(\mathbf{W}^{(k)}\right)^{\mathrm{T}} \mathbf{V}^{(l)} = \begin{cases} \mathbf{\Delta}^{(l)} & \text{if } k = l, \\ 0 & \text{if } k \neq l, \end{cases} \qquad \text{for all} \quad k, l = 1, 2, \ldots, l_{\max}. \tag{3.11}$$

Note that, in the absence of exact or near-breakdowns, all clusters (3.8) consist of single Lanczos vectors only. In particular, in this case,

$$n_l = l, \quad \mathbf{V}^{(l)} = \mathbf{v}_l, \quad \mathbf{W}^{(l)} = \mathbf{w}_l, \quad \mathbf{\Delta}^{(l)} = \delta_l \quad \text{for all} \quad l = 1, 2, \ldots, l_{\max} = n_{\max},$$

and the cluster-wise biorthogonality (3.11) reduces to (3.5).

In the algorithm, we use the cluster indices (3.10) to keep track of the structure of the clusters (3.8). Occasionally, we will need to determine the index, denoted by $l(n)$, of the clusters $\mathbf{V}^{l(n)}$ and $\mathbf{W}^{l(n)}$ that contain the n-th pair of Lanczos vectors $\mathbf{v}_n$ and $\mathbf{w}_n$. In view of (3.9) and (3.10), for $n = 1, 2, \ldots, n_{\max}$, $l(n)$ is given by

$$l(n) = \max\{\, i \in \mathbb{N} \mid n_i \leq n \,\}. \tag{3.12}$$

3.4. **Matrix formulation.** It turns out to be convenient to use the notation

$$\mathbf{V}_n := \begin{bmatrix} \mathbf{v}_1 & \mathbf{v}_2 & \cdots & \mathbf{v}_n \end{bmatrix} \quad \text{and} \quad \mathbf{W}_n := \begin{bmatrix} \mathbf{w}_1 & \mathbf{w}_2 & \cdots & \mathbf{w}_n \end{bmatrix} \tag{3.13}$$

for the $N \times n$ matrices whose columns are the first n right and n left Lanczos vectors, respectively. Moreover, we define $\mathbf{\Delta}_n$ as the $n \times n$ leading principal submatrix of the block-diagonal matrix, partitioned conformally with the clusters (3.8),

$$\mathbf{\Delta}_{n_{\max}} := \operatorname{diag}\left(\mathbf{\Delta}^{(1)}, \mathbf{\Delta}^{(2)}, \ldots, \mathbf{\Delta}^{(l_{\max})}\right). \tag{3.14}$$

With (3.13) and $\mathbf{\Delta}_n$, we can state the cluster-wise biorthogonality relations (3.11) in compact matrix form as follows:

$$\mathbf{W}_n^{\mathrm{T}} \mathbf{V}_n = \mathbf{\Delta}_n \quad \text{for all} \quad n = 1, 2, \ldots, n_{\max}. \tag{3.15}$$

3.5. **Normalization.** The biorthogonality relations determine the Lanczos vectors only up to scalar normalization factors. In an actual implementation, proper normalization is important in order to avoid possible over- and underflow in finite-precision arithmetic. Furthermore, the usual strategies to decide when to perform a look-ahead step assume that the Lanczos vectors are normalized; see, e.g., [17]. Following [17], we normalize the Lanczos vectors to have unit Euclidean length:

$$\|\mathbf{v}_n\| = \|\mathbf{w}_n\| = 1 \quad \text{for all} \quad n. \tag{3.16}$$

4. The construction of the Lanczos vectors

In this section, we describe the construction of the Lanczos vectors in our algorithm for multiple starting vectors, by analogy with the Lanczos process for single starting vectors.

4.1. **Review of the case of single starting vectors.** The Lanczos process [27] for single starting vectors $\mathbf{r}$ and $\mathbf{l}$ obtains $\mathbf{v}_1$ and $\mathbf{w}_1$ by normalizing $\mathbf{r}$ and $\mathbf{l}$ so that (3.16) (for $n = 1$) is satisfied. For $n > 1$, $\mathbf{v}_n$ is obtained by first computing $\mathbf{v} = \mathbf{A}\,\mathbf{v}_{n-1}$ to advance the right Krylov subspace, then biorthogonalizing $\mathbf{v}$ against the previous left Lanczos vectors $\mathbf{w}_i$, $i < n$, and finally normalizing the biorthogonalized vector $\mathbf{v}$. In exact arithmetic and in the absence of look-ahead, $\mathbf{v}$ is already biorthogonal to $\mathbf{w}_i$, $i < n-2$, and so $\mathbf{v}$ only needs to be biorthogonalized against $\mathbf{w}_{n-2}$ and $\mathbf{w}_{n-1}$. In the case of look-ahead, $\mathbf{v}$ is biorthogonalized against the previous two clusters; see, e.g., [17]. Similarly, $\mathbf{w}_n$ is obtained by first computing $\mathbf{w} = \mathbf{A}^{\mathrm{T}}\,\mathbf{w}_{n-1}$, then biorthogonalizing $\mathbf{w}$ against $\mathbf{v}_{n-2}$ and $\mathbf{v}_{n-1}$, respectively the two previous clusters, and finally normalizing the biorthogonalized vector $\mathbf{w}$. Using the matrix notation (3.13), all these recurrences for the first n right and n left Lanczos vectors can be summarized as follows:

$$
\begin{aligned}
\mathbf{V}_n\,\mathbf{T}_{n-1} &= \begin{cases} \mathbf{r} & \text{if } n = 1, \\ \mathbf{A}\,\mathbf{V}_{n-1} & \text{if } n > 1, \end{cases} \\
\mathbf{W}_n\,\widetilde{\mathbf{T}}_{n-1} &= \begin{cases} \mathbf{l} & \text{if } n = 1, \\ \mathbf{A}^{\mathrm{T}}\,\mathbf{W}_{n-1} & \text{if } n > 1. \end{cases}
\end{aligned}
\tag{4.1}
$$

Here, for $n = 1$, $\mathbf{T}_0$ and $\widetilde{\mathbf{T}}_0$ are scalars that record the normalization of the starting vectors $\mathbf{r}$ and $\mathbf{l}$, respectively. For $n > 1$, $\mathbf{T}_{n-1}$ and $\widetilde{\mathbf{T}}_{n-1}$ are $n \times (n-1)$ matrices that contain the recurrence coefficients. Both matrices are tridiagonal if no look-ahead is necessary, and they are simultaneously upper Hessenberg and block tridiagonal if look-ahead steps are performed; see, e.g., [17].

4.2. **Multiple starting vectors, but no deflation or look-ahead.** We now turn to the Lanczos-type algorithm for multiple starting vectors. In this subsection, we describe the procedure for the simplest case, where neither deflation nor look-ahead occurs. For $n \le m$, the vector $\mathbf{v}_n$ is generated from the n-th right starting vector $\mathbf{v} = \mathbf{r}_n$ by first biorthogonalizing $\mathbf{v}$ against $\mathbf{w}_i$, $1 \le i < n$, and then normalizing the biorthogonalized vector $\mathbf{v}$ to unit length. For $n > m$, the vector $\mathbf{v}_n$ is obtained by first computing $\mathbf{v} = \mathbf{A}\,\mathbf{v}_{n-m}$ to advance the right block Krylov subspace, then biorthogonalizing $\mathbf{v}$ against the previous left Lanczos vectors $\mathbf{w}_i$, $n - m - p \le i < n$, and finally normalizing the biorthogonalized vector $\mathbf{v}$ to unit length. Similarly, for $n \le p$, the vector $\mathbf{w}_n$ is obtained by first biorthogonalizing $\mathbf{w} = \mathbf{l}_n$ against $\mathbf{v}_i$, $1 \le i < n$, and then normalizing the biorthogonalized vector $\mathbf{w}$ to unit length. For $n > p$, the vector $\mathbf{w}_n$ is obtained by first computing $\mathbf{w} = \mathbf{A}^{\mathrm{T}}\,\mathbf{w}_{n-p}$, then biorthogonalizing $\mathbf{w}$ against the previous right Lanczos vectors $\mathbf{v}_i$, $n - m - p \le i < n$. In analogy to (4.1), the resulting recurrences for the first n right and n left Lanczos vectors can be summarized as follows:

$$
\begin{aligned}
\mathbf{V}_n\,\mathbf{T}_{n-m} &= \begin{cases} [\mathbf{r}_1 \quad \mathbf{r}_2 \quad \cdots \quad \mathbf{r}_n] & \text{if } 1 \le n \le m, \\ \mathbf{A}\,\mathbf{V}_{n-m} & \text{if } n > m, \end{cases} \\
\mathbf{W}_n\,\widetilde{\mathbf{T}}_{n-p} &= \begin{cases} [\mathbf{l}_1 \quad \mathbf{l}_2 \quad \cdots \quad \mathbf{l}_n] & \text{if } 1 \le n \le p, \\ \mathbf{A}^{\mathrm{T}}\,\mathbf{W}_{n-p} & \text{if } n > p. \end{cases}
\end{aligned}
\tag{4.2}
$$

0) *Initialization; in particular, set* $\mu = \mu_0 = -m$ *and* $\phi = \phi_0 = -p$.
For $n = 1, 2, \ldots,$ *build* n*-th pair of Lanczos vectors* $\mathbf{v}_n$ *and* $\mathbf{w}_n$:
1) *Build the unnormalized right Lanczos vector* $\mathbf{v}$.
1a) *Set* $\mu = \mu + 1$.
1b) *If the right block Krylov subspace is exhausted, then stop.*
1c) *Advance the right block Krylov subspace.*
1d) *Determine against which clusters* $\mathbf{v}$ *needs to be biorthogonalized.*
1e) *Biorthogonalize* $\mathbf{v}$ *against these clusters.*
1f) *Decide if* $\mathbf{v}$ *should be deflated; if so, deflate it and repeat all of Step* 1).
2) *Build the unnormalized left Lanczos vector* $\mathbf{w}$.
2a) *Set* $\phi = \phi + 1$.
2b) *If the left block Krylov subspace is exhausted, then stop.*
2c) *Advance the left block Krylov subspace.*
2d) *Determine against which clusters* $\mathbf{w}$ *needs to be biorthogonalized.*
2e) *Biorthogonalize* $\mathbf{w}$ *against these clusters.*
2f) *Decide if* $\mathbf{w}$ *should be deflated; if so, deflate it and repeat all of Step* 2).
3) *Normalize* $\mathbf{v}$ *and* $\mathbf{w}$ *to obtain* $\mathbf{v}_n$ *and* $\mathbf{w}_n$*, and add them to current clusters.*
4) *Record the* n*-th history indices* $\mu_n = \mu$ *and* $\phi_n = \phi$.
5) *Check for end of look-ahead clusters.*

FIGURE 1. Structure of Lanczos-type algorithm.

Here, the matrices $\mathbf{T}_{n-m}$ and $\widetilde{\mathbf{T}}_{n-p}$ again contain the recurrence coefficients used in the biorthogonalization of the Lanczos vectors. For $n \leq m$, respectively $n \leq p$, the matrix $\mathbf{T}_{n-m}$, respectively $\widetilde{\mathbf{T}}_{n-p}$, is an upper triangular $n \times n$ matrix, recording the biorthogonalization of the right, respectively left, starting vectors. For $n > m$, $\mathbf{T}_{n-m}$ is a banded $n \times (n-m)$ matrix with lower and upper bandwidth m and p, respectively. For $n > p$, $\widetilde{\mathbf{T}}_{n-p}$ is a banded $n \times (n-p)$ matrix with lower and upper bandwidth p and m, respectively.

4.3. **Deflation.** We now begin our discussion of the algorithm in the general case, where deflation and look-ahead may occur. In Figure 1, we show the basic structure of the algorithm; a complete statement of the algorithm is given in §5.1 below. In the following, we use the step numbers from Figure 1 to describe the algorithm.

We use n as the counter for the main loop of the algorithm. During the n-th pass through the main loop, the Lanczos vectors $\mathbf{v}_n$ and $\mathbf{w}_n$ are being computed. This is done by first constructing, in Steps 1) and 2), candidate vectors $\mathbf{v}$ and $\mathbf{w}$ that satisfy the required biorthogonality conditions for $\mathbf{v}_n$ and $\mathbf{w}_n$. In the proof of Theorem 6.3 in §6, we will show that $\mathbf{v} = \mathbf{0}$ or $\mathbf{w} = \mathbf{0}$ is equivalent to an exact deflation in the right or left block Krylov subspace, respectively. Therefore, in the algorithm, we check for exact deflation by simply testing if $\mathbf{v} = \mathbf{0}$ or $\mathbf{w} = \mathbf{0}$. If inexact deflation is included, we choose a small deflation tolerance `dtol`, and then check for deflation by testing if

$$\|\mathbf{v}\| \leq \texttt{dtol} \quad \text{or} \quad \|\mathbf{w}\| \leq \texttt{dtol}. \tag{4.3}$$

If $\|\mathbf{v}\| \leq$ `dtol`, then $\mathbf{v}$ is deflated and all of Step 1) of the algorithm is repeated. If $\|\mathbf{w}\| \leq$ `dtol`, then $\mathbf{w}$ is deflated and all of Step 2) is repeated. Once Steps 1) and 2) have produced vectors $\mathbf{v}$ and $\mathbf{w}$ with $\|\mathbf{v}\| >$ `dtol` and $\|\mathbf{w}\| >$ `dtol`, the n-th

Lanczos vectors $\mathbf{v}_n$ and $\mathbf{w}_n$ are obtained by normalizing $\mathbf{v}$ and $\mathbf{w}$ in Step 3). Note that deflation reduces to exact deflation only by setting $\mathtt{dtol} = 0$ in (4.3).

Every time a $\mathbf{v}$ vector is deflated, the effective right block size, which was m initially, is reduced by 1. Similarly, each deflation of a $\mathbf{w}$ vector reduces the effective left block size by 1. If only exact deflation is performed, then the effective right and left block sizes could be recorded by keeping track of the sizes of the blocks $\mathbf{A}^{j-1}\,\mathbf{R}_j$ and $(\mathbf{A}^{\mathrm{T}})^{k-1}\,\mathbf{L}_k$ in (2.2), together with pointers for the positions of $\mathbf{v}_n$ and $\mathbf{w}_n$ relative to the current blocks in (2.2). However, this approach cannot easily be extended to include inexact deflation. Instead, we use a different encoding that properly describes the effective block sizes even when both exact and inexact deflation are performed. This encoding is based on history indices for the individual Lanczos vectors. Next, we describe these indices.

4.4. **History indices.** In the algorithm, the indices μ and ϕ are used to record the number of passes through Steps 1) and 2), respectively; see Figure 1. At the beginning of the n-th pass through the main loop of the algorithm, $\mu = \mu_{n-1}$ and $\phi = \phi_{n-1}$, where, initially, we set $\mu_0 = -m$ and $\phi_0 = -p$. The value of μ is incremented by 1 in every pass through Step 1), and ϕ is incremented by 1 in every pass through Step 2). The values of μ and ϕ after Steps 1) and 2) have been performed the last time within the n-th pass through the main loop are recorded as the n-th *history indices*, μ_n and ϕ_n, in Step 4) of the algorithm. Recall from §4.3 that Steps 1) or 2) are repeated within each n-th pass through the main loop if, and only if, a deflation of a $\mathbf{v}$ or $\mathbf{w}$ vector has occurred. If follows that $\mu_n - \mu_{n-1} - 1$ is the number of deflations of consecutive $\mathbf{v}$ vectors in between the construction of $\mathbf{v}_{n-1}$ and $\mathbf{v}_n$. Similarly, $\phi_n - \phi_{n-1} - 1$ is the number of deflations of consecutive $\mathbf{w}$ vectors in between the construction of $\mathbf{w}_{n-1}$ and $\mathbf{w}_n$. In particular,

$$\mu_n = \mu_{n-1} + 1 \quad \text{and} \quad \phi_n = \phi_{n-1} + 1 \tag{4.4}$$

if no deflation of a $\mathbf{v}$ or $\mathbf{w}$ vector occurred during the n-th pass.

It turns out that μ_n and ϕ_n are just the indices of the vectors from which $\mathbf{v}_n$ and $\mathbf{w}_n$, respectively, were generated. More precisely, if $\mu_n \le 0$, then $\mathbf{v}_n$ was obtained by biorthogonalizing and normalizing one of the right starting vectors, namely $\mathbf{r}_{\mu_n+m}$. If $\mu_n > 0$, then $\mathbf{v}_n$ was generated from the $\mathbf{A}$-multiple $\mathbf{A}\,\mathbf{v}_{\mu_n}$ of the previously constructed vector $\mathbf{v}_{\mu_n}$. Similarly, an index $\phi_n \le 0$ means that $\mathbf{w}_n$ was generated from the left starting vector $\mathbf{l}_{\phi_n+p}$, while $\phi_n > 0$ means that $\mathbf{w}_n$ was generated from $\mathbf{A}^{\mathrm{T}}\,\mathbf{w}_{\phi_n}$. It will also be convenient to set $\mu_{n_{\max}+1} := \phi_{n_{\max}+1} := n_{\max} + 1$.

Note that, by construction, the history indices are strictly increasing:

$$\begin{aligned} &\mu_0 < \mu_1 < \cdots < \mu_{n-1} < \mu_n < \cdots < \mu_{n_{\max}+1}, \\ &\phi_0 < \phi_1 < \cdots < \phi_{n-1} < \phi_n < \cdots < \phi_{n_{\max}+1}. \end{aligned} \tag{4.5}$$

Furthermore, we have $n - m \le \mu_n < n$ and $n - p \le \phi_n < n$ for all $n \le n_{\max}$.

The following example illustrates the concept of the history indices.

Example 4.1. Suppose that $\{\,\mu_0, \mu_1, \mu_2, \mu_3, \mu_4, \mu_5, \dots\,\} = \{\,-2, -1, 0, 1, 3, 4, \dots\}$. This means $\mathbf{v}_1$ and $\mathbf{v}_2$ came from the $m = -\mu_0 = 2$ starting vectors, while $\mathbf{v}_3$, $\mathbf{v}_4$, and $\mathbf{v}_5$ came from the $\mathbf{A}$-multiples $\mathbf{A}\,\mathbf{v}_1$, $\mathbf{A}\,\mathbf{v}_3$, and $\mathbf{A}\,\mathbf{v}_4$. Since $\mu_4 - \mu_3 = 2 > 1$, there was one deflation in between the construction of $\mathbf{v}_3$ and $\mathbf{v}_4$. More precisely, the vector that would have resulted from $\mathbf{A}\,\mathbf{v}_2$ was deflated.

Remark 4.2. If no deflation occurred during the construction of the first n pairs of Lanczos vectors, then $\mu_n = n - m$ and $\phi_n = n - p$. This readily follows from (4.4).

Remark 4.3. In the case of only exact deflation, the history indices (4.5) can be used to determine the sizes of the blocks $\mathbf{R}_j$ and $\mathbf{L}_k$ in the deflated block Krylov matrices (2.2). In [2], we show how to do this for the right blocks $\mathbf{R}_j$; the case of the left blocks is analogous.

4.5. **Recurrence relations for the general case.** After having introduced the history indices, we are now in a position to state the recurrence relations for the most general version of our algorithm with both deflation and look-ahead.

We continue to use the matrix notation $\mathbf{V}_n$ and $\mathbf{W}_n$ introduced in (3.13). Moreover, for $n = 0$, we set $\mathbf{V}_0 = \mathbf{W}_0 := \emptyset$. In analogy to (4.2), the recurrences for the construction of the first n right and n left Lanczos vectors can then be summarized as follows. For all $n = 0, 1, \ldots, n_{\max}$, we have

$$
\begin{aligned}
&\mathbf{V}_n \, \mathbf{T}_\mu + \mathbf{V}_\mu^{\mathrm{dl}} = \begin{cases} [\mathbf{r}_1 \quad \mathbf{r}_2 \quad \cdots \quad \mathbf{r}_{\mu+m}] & \text{if } \mu \leq 0, \\ \mathbf{A} \, \mathbf{V}_\mu & \text{if } \mu > 0, \end{cases} \qquad \mu_n \leq \mu < \mu_{n+1}, \\
&\mathbf{W}_n \, \widetilde{\mathbf{T}}_\phi + \mathbf{W}_\phi^{\mathrm{dl}} = \begin{cases} [\mathbf{l}_1 \quad \mathbf{l}_2 \quad \cdots \quad \mathbf{l}_{\phi+p}] & \text{if } \phi \leq 0, \\ \mathbf{A}^{\mathrm{T}} \, \mathbf{W}_\phi & \text{if } \phi > 0, \end{cases} \qquad \phi_n \leq \phi < \phi_{n+1}.
\end{aligned} \tag{4.6}
$$

Here, $\mathbf{V}_\mu^{\mathrm{dl}}$ and $\mathbf{W}_\mu^{\mathrm{dl}}$ are $N \times \mu$ matrices that contain the vectors $\mathbf{v}$ and $\mathbf{w}$ that were not turned into Lanczos vectors due to deflation, together with lots of zero vectors. More precisely, these matrices are built up as follows:

$$
\begin{aligned}
&\mathbf{V}_\mu^{\mathrm{dl}} = \begin{cases} \begin{bmatrix} \mathbf{V}_{\mu-1}^{\mathrm{dl}} & \mathbf{v} \end{bmatrix} & \text{if } \mathbf{v} \neq \mathbf{0} \text{ and } \mathbf{v} \text{ is deflated}, \\ \begin{bmatrix} \mathbf{V}_{\mu-1}^{\mathrm{dl}} & \mathbf{0} \end{bmatrix} & \text{otherwise}, \end{cases} \\
&\mathbf{W}_\phi^{\mathrm{dl}} = \begin{cases} \begin{bmatrix} \mathbf{W}_{\phi-1}^{\mathrm{dl}} & \mathbf{w} \end{bmatrix} & \text{if } \mathbf{w} \neq \mathbf{0} \text{ and } \mathbf{w} \text{ is deflated}, \\ \begin{bmatrix} \mathbf{W}_{\phi-1}^{\mathrm{dl}} & \mathbf{0} \end{bmatrix} & \text{otherwise}, \end{cases}
\end{aligned} \tag{4.7}
$$

where $\mathbf{V}_0^{\mathrm{dl}} = \mathbf{W}_0^{\mathrm{dl}} := \emptyset$. In particular, if no deflation or only exact deflation occurs, then $\mathbf{V}_\mu^{\mathrm{dl}} = \mathbf{W}_\phi^{\mathrm{dl}}$ are zero matrices. If inexact deflation occurs, then $\mathbf{V}_\mu^{\mathrm{dl}}$ and $\mathbf{W}_\phi^{\mathrm{dl}}$ are no longer zero matrices , but they are still small in norm. At any stage of our algorithm, the number of $\mathbf{v}$ and $\mathbf{w}$ vectors that have been deflated so far is given by $m-n+\mu$ and $p-n+\phi$, respectively. Thus $\mathbf{V}_\mu^{\mathrm{dl}}$ and $\mathbf{W}_\phi^{\mathrm{dl}}$ have at most $m-n+\mu$ and $p-n+\phi$, respectively, nonzero columns. In view of the deflation criterion (4.3), these nonzero columns have Euclidean norm at most `dtol`. It follows that

$$
\left\| \mathbf{V}_\mu^{\mathrm{dl}} \right\| \leq \mathtt{dtol} \sqrt{m-n+\mu} \quad \text{and} \quad \left\| \mathbf{W}_\phi^{\mathrm{dl}} \right\| \leq \mathtt{dtol} \sqrt{p-n+\phi}.
$$

As in (4.2), the matrices $\mathbf{T}_\mu$ and $\widetilde{\mathbf{T}}_\phi$ in (4.6) contain the recurrence coefficients used to enforce the biorthogonality conditions for the Lanczos vectors. For $\mu \leq 0$ and $\phi \leq 0$, the columns of $\mathbf{T}_\mu$ and $\widetilde{\mathbf{T}}_\phi$ are obtained by biorthogonalizing the columns of the starting blocks $\mathbf{R}$ and $\mathbf{L}$, respectively. For $\mu > 0$ and $\phi > 0$, the columns of $\mathbf{T}_\mu$ and $\widetilde{\mathbf{T}}_\phi$ contain the recurrences used to advance the right and left block Krylov subspaces by multiplications with $\mathbf{A}$ and $\mathbf{A}^{\mathrm{T}}$, respectively.

If no deflation occurs, then $\mathbf{V}_\mu^{\mathrm{dl}} = \mathbf{0}$, $\mathbf{W}_\phi^{\mathrm{dl}} = \mathbf{0}$, $\mu = n-m$, $\phi = n-p$, and hence the recurrences (4.6) reduce to (4.2). If, in addition, no look-ahead occurs, then the matrices $\mathbf{T}_\mu$ and $\widetilde{\mathbf{T}}_\phi$ in (4.6) are identical to those in (4.2). Recall from §4.2

that $\mathbf{T}_\mu$ and $\widetilde{\mathbf{T}}_\phi$ in (4.2) are banded matrices with lower and upper bandwidth m and p, respectively p and m.

In the general case where deflation and look-ahead may occur, the structure of $\mathbf{T}_\mu$ and $\widetilde{\mathbf{T}}_\phi$ is more complicated. As before, $\mathbf{T}_\mu$ and $\widetilde{\mathbf{T}}_\phi$ still have lower bandwidth m and p, respectively. However, after each deflation of a $\mathbf{v}$ or $\mathbf{w}$ vector, the lower bandwidth for the successive columns of $\mathbf{T}_\mu$ or $\widetilde{\mathbf{T}}_\phi$, respectively, is reduced by one. As before, $\mathbf{T}_\mu$ and $\widetilde{\mathbf{T}}_\phi$ have nonzero elements within a band of upper bandwidth p and m, respectively. After each deflation of a $\mathbf{v}$ or $\mathbf{w}$ vector, the upper bandwidth for the successive columns of $\widetilde{\mathbf{T}}_\phi$ or $\mathbf{T}_\mu$, respectively, is reduced by one. Look-ahead and inexact deflation result in some additional nonzero elements above the upper bands of $\mathbf{T}_\mu$ and $\widetilde{\mathbf{T}}_\phi$. More precisely, look-ahead steps result in "bulges" in $\mathbf{T}_\mu$ and $\widetilde{\mathbf{T}}_\phi$ above their upper bands. The cluster indices $l_\mathbf{v}$ and $l_\mathbf{w}$ describe the beginning of the upper band, including any look-ahead bulges, in the μ-th and ϕ-th column of $\mathbf{T}_\mu$ and $\widetilde{\mathbf{T}}_\phi$, respectively; see Algorithm 5.1 below. Each inexact deflation of a $\mathbf{v}$ vector requires that all successive left Lanczos vectors need to be explicitly biorthogonalized against a certain earlier vector $\mathbf{v}_i$, and in the look-ahead case, against the entire cluster containing $\mathbf{v}_i$. Similarly, inexact deflation of a $\mathbf{w}$ vector requires explicit biorthogonalization of all successive right Lanczos vectors against a certain earlier vector $\mathbf{w}_i$, and in the look-ahead case, against the entire cluster containing $\mathbf{w}_i$. The indices of the look-ahead clusters against which one needs to biorthogonalize due to inexact deflation of $\mathbf{v}$ and $\mathbf{w}$ vectors are stored in the sets $\mathcal{I}_\mathbf{w}$ and $\mathcal{I}_\mathbf{v}$, respectively; see Algorithm 5.1 below. These biorthogonalizations caused by inexact deflation result in nonzeros in rows of $\mathbf{T}_\mu$ and $\widetilde{\mathbf{T}}_\phi$ whose row indices correspond to $\mathcal{I}_\mathbf{v}$ and $\mathcal{I}_\mathbf{w}$, respectively. However, these nonzeros only appear to the right of the bands.

4.6. **An example.** In this subsection, we present a specific example to illustrate how deflation and look-ahead affect the zero structure of $\mathbf{T}_\mu$ and $\widetilde{\mathbf{T}}_\phi$.

This is an example with $m = 3$ and $p = 2$. We assume that three deflations of $\mathbf{v}$ vectors and one deflation of a $\mathbf{w}$ vector occur. More precisely, the $\mathbf{v}$ vectors obtained by biorthogonalizing $\mathbf{A}\,\mathbf{v}_8$, $\mathbf{A}\,\mathbf{v}_{19}$, and $\mathbf{A}\,\mathbf{v}_{20}$ and the $\mathbf{w}$ vector obtained by biorthogonalizing $\mathbf{A}^\mathrm{T}\,\mathbf{w}_{12}$ are deflated. The corresponding history indices are shown in the second and third columns of Table 1, where the values of μ and ϕ in parentheses indicate the repeated executions of Steps 1) and 2), respectively, due to deflation. Note that $n = \mu$ at pass $n = 21$. This means that the right block Krylov subspace is exhausted, and thus the algorithm stops in Step 1b) at pass $n = 21$, after having generated 20 pairs of Lanczos vectors, namely

$$\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_{20} \quad \text{and} \quad \mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_{20}. \tag{4.8}$$

If no look-ahead occurred, the vectors (4.8) are constructed to be vector-wise biorthogonal. We also consider a case where true look-ahead clusters occur, with the values of the cluster indices (3.10) given in the ninth column of Table 1. Note that there are two true look-ahead clusters of length three, starting at $n_5 = 5$ and at $n_6 = 8$, respectively. Thus, the vectors (4.8) are grouped into clusters

$$\mathbf{V}^{(1)}, \mathbf{V}^{(2)}, \ldots, \mathbf{V}^{(16)} \quad \text{and} \quad \mathbf{W}^{(1)}, \mathbf{W}^{(2)}, \ldots, \mathbf{W}^{(16)}, \tag{4.9}$$

where $\mathbf{V}^{(5)} = \begin{bmatrix}\mathbf{v}_5 & \mathbf{v}_6 & \mathbf{v}_7\end{bmatrix}$, $\mathbf{V}^{(6)} = \begin{bmatrix}\mathbf{v}_8 & \mathbf{v}_9 & \mathbf{v}_{10}\end{bmatrix}$, $\mathbf{W}^{(5)} = \begin{bmatrix}\mathbf{w}_5 & \mathbf{w}_6 & \mathbf{w}_7\end{bmatrix}$, $\mathbf{W}^{(6)} = \begin{bmatrix}\mathbf{w}_8 & \mathbf{w}_9 & \mathbf{w}_{10}\end{bmatrix}$, and the remaining clusters consist of single vectors only.

	History indices		No look-ahead				With look-ahead					
n	μ_n	ϕ_n	$l_{\mathbf{v}}$	$\mathcal{I}_{\mathbf{v}}$	$l_{\mathbf{w}}$	$\mathcal{I}_{\mathbf{w}}$	l	n_l	$l_{\mathbf{v}}$	$\mathcal{I}_{\mathbf{v}}$	$l_{\mathbf{w}}$	$\mathcal{I}_{\mathbf{w}}$
0	-3	-2										
1	-2	-1	1	$\emptyset$	1	$\emptyset$	1	1	1	$\emptyset$	1	$\emptyset$
2	-1	0	1	$\emptyset$	1	$\emptyset$	2	2	1	$\emptyset$	1	$\emptyset$
3	0	1	1	$\emptyset$	1	$\emptyset$	3	3	1	$\emptyset$	1	$\emptyset$
4	1	2	1	$\emptyset$	1	$\emptyset$	4	4	1	$\emptyset$	1	$\emptyset$
5	2	3	1	$\emptyset$	1	$\emptyset$	5	5	1	$\emptyset$	1	$\emptyset$
6	3	4	1	$\emptyset$	1	$\emptyset$			1	$\emptyset$	1	$\emptyset$
7	4	5	2	$\emptyset$	2	$\emptyset$			2	$\emptyset$	2	$\emptyset$
8	5	6	3	$\emptyset$	3	$\emptyset$	6	8	3	$\emptyset$	2	$\emptyset$
9	6	7	4	$\emptyset$	4	$\emptyset$			3	$\emptyset$	2	$\emptyset$
10	7	8	5	$\emptyset$	5	$\emptyset$			3	$\emptyset$	5	$\emptyset$
11	$(\mu = 8)$		6	$\emptyset$		$\{8\}$			5	$\emptyset$		$\{6\}$
11	9	9	7	$\emptyset$	6	$\{8\}$	7	11	5	$\emptyset$	5	$\{6\}$
12	10	10	8	$\emptyset$	7	$\{8\}$	8	12	5	$\emptyset$	5	$\{6\}$
13	11	11	9	$\emptyset$	9	$\{8\}$	9	13	6	$\emptyset$	6	$\{6\}$
14		$(\phi = 12)$		$\{12\}$	10	$\{8\}$				$\{8\}$	6	$\{6\}$
14	12	13	10	$\{12\}$	11	$\{8\}$	10	14	6	$\{8\}$	7	$\{6\}$
15	13	14	11	$\{12\}$	12	$\{8\}$	11	15	7	$\{8\}$	8	$\{6\}$
16	14	15	13	$\{12\}$	13	$\{8\}$	12	16	9	$\{8\}$	9	$\{6\}$
17	15	16	14	$\{12\}$	14	$\{8\}$	13	17	10	$\{8\}$	10	$\{6\}$
18	16	17	15	$\{12\}$	15	$\{8\}$	14	18	11	$\{8\}$	11	$\{6\}$
19	17	18	16	$\{12\}$	16	$\{8\}$	15	19	12	$\{8\}$	12	$\{6\}$
20	18	19	17	$\{12\}$	17	$\{8\}$	16	20	13	$\{8\}$	13	$\{6\}$
21	$(\mu = 19)$		18			$\{8,19\}$			14			$\{6,15\}$
21	$(\mu = 20)$		19			$\{8,19,20\}$			15			$\{6,15,16\}$
21	$(\mu = 21)$											

TABLE 1. Indices and sets, which are used in Algorithm 5.1, for the example in §4.6.

In the look-ahead case, the vectors (4.8) are constructed such that the clusters (4.9) are biorthogonal.

For both the no-look-ahead and the look-ahead case, the zero structure of the matrices $\mathbf{\Delta}_{20}$, $\mathbf{T}_{20}$, and $\widetilde{\mathbf{T}}_{20}$ are shown in Figures 2 and 3. The following convention is used: guaranteed zeros are marked by "·", while potential nonzeros are marked by "+", "x", "c", "d", or "b". The distinction between the potential nonzeros is as follows: diagonal entries are marked by "+", other elements within the bands of $\mathbf{T}_{20}$ and $\widetilde{\mathbf{T}}_{20}$ are marked by "x", entries due to look-ahead are marked by "c", entries due to inexact deflation are marked by "d", and entries resulting from the combined effects of look-ahead and deflation are marked by "b".

Note that, by (3.14) and (3.15), the biorthogonality properties of the vectors (4.8) translate into a block-diagonal structure of the matrix $\mathbf{\Delta}_{20}$. Indeed, as shown in Figure 2, in the no-look-ahead case, $\mathbf{\Delta}_{20}$ is diagonal, reflecting the vector-wise biorthogonality of the vectors (4.8). In the look-ahead case, $\mathbf{\Delta}_{20}$ has two 3×3 diagonal blocks, reflecting the cluster-wise biorthogonality of the clusters (4.9).

Figure 3 shows the zero structure of $\mathbf{T}_{20}$ and $\widetilde{\mathbf{T}}_{20}$. First, consider the no-look-ahead case. Here the structure of $\mathbf{T}_{20}$ and $\widetilde{\mathbf{T}}_{20}$ is completely described by the data listed in the first seven columns of Table 1. At the n-th pass through the main loop of the algorithm, the μ-th column of $\mathbf{T}_{20}$ for all $\mu_{n-1} < \mu \leq \mu_n$ and the

$$
\mathbf{\Delta}_{20} = \begin{bmatrix}
+ & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
\cdot & + & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & + & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & + & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot & + & c & c & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot & c & + & c & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot & c & c & + & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & + & c & c & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & c & + & c & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & c & c & + & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & + & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & + & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & + & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & + & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & + & \cdot & \cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & + & \cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & + & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & + & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & + & \cdot \\
\cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & +
\end{bmatrix}
$$

FIGURE 2. Zero structure of the matrix $\mathbf{\Delta}_{20}$ for the example in §4.6.

ϕ-th column of $\widetilde{\mathbf{T}}_{20}$ for all $\phi_{n-1} < \phi \le \phi_n$ are constructed. Recall that, by (4.4), we have $\mu = \mu_n$ and $\phi = \phi_n$ if, and only if, no deflation occurs during the n-th pass. From Table 1, the zero structure of the columns of $\mathbf{T}_{20}$ and $\widetilde{\mathbf{T}}_{20}$ that are constructed at the n-th pass can be read off directly from the row(s) of Table 1 starting with the value of n. Note that there are multiple rows starting with the same value of n and corresponding to multiple values of μ or ϕ if deflation occurs during the n-th pass. For example, in Table 1, there are two rows associated with $n = 11$. The first row records the structure of column $\mu = 8$ of $\mathbf{T}_{20}$, as well as the deflation of the $\mathbf{v}$ vector obtained from $\mathbf{A}\,\mathbf{v}_8$. More precisely, the banded part of the 8-th column of $\mathbf{T}_{20}$ starts with the entry in row $l_{\mathbf{v}} = 6$, and since $\mathcal{I}_{\mathbf{v}} = \emptyset$,

$$
\mathbf{T}_{20}: \quad \begin{bmatrix}
+ & x & x & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
x & + & x & x & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
x & x & + & x & x & c & c & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
x & x & x & + & x & x & c & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
\cdot & x & x & x & + & x & x & c & c & c & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & x & x & x & + & x & x & c & c & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & x & x & x & + & x & x & c & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot & x & x & x & + & x & x & c & c & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot & \cdot & x & x & x & + & x & x & c & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot & \cdot & \cdot & x & x & x & + & x & x & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & x & x & + & x & x & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & x & x & + & x & d & d & d & d & d & d & d \\
\cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & x & x & + & x & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & x & x & + & x & \cdot & \cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & x & x & + & x & \cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & x & x & + & x & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & x & x & + & x & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & x & x & + & x & \cdot \\
\cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & x & x & + & x \\
\cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & x & x & +
\end{bmatrix}
$$

$$
\widetilde{\mathbf{T}}_{20}: \quad \begin{bmatrix}
+ & x & x & x & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
x & + & x & x & x & c & c & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
x & x & + & x & x & x & c & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
\cdot & x & x & + & x & x & x & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & x & x & + & x & x & x & c & c & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & x & x & + & x & x & x & c & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot & x & x & + & x & x & x & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot & \cdot & x & x & + & x & x & x & d & d & d & d & d & d & d & d & d \\
\cdot & \cdot & \cdot & \cdot & \cdot & \cdot & x & x & + & x & x & c & b & b & b & b & b & b & b & b \\
\cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & x & x & + & x & x & b & b & b & b & b & b & b & b \\
\cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & x & x & + & x & x & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & x & x & + & x & x & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & x & x & + & x & x & \cdot & \cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & x & + & x & x & \cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & x & + & x & x & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & x & + & x & x & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & x & + & x & x & \cdot \\
\cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & x & + & x & x \\
\cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & x & + & x \\
\cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & x & +
\end{bmatrix}
$$

FIGURE 3. Zero structure of the Lanczos matrices $\mathbf{T}_{20}$ and $\widetilde{\mathbf{T}}_{20}$ for the example in §4.6.

there are no nonzero entries due to earlier deflations outside the banded part. The deflation of the $\mathbf{v}$ vector obtained from $\mathbf{A}\,\mathbf{v}_8$ has the affect that $\mu = 8$ is added to $\mathcal{I}_{\mathbf{w}}$. This means that all later $\mathbf{w}$ vectors will have to be biorthogonalized against $\mathbf{v}_8$, resulting in potentially nonzero entries in row 8 of $\widetilde{\mathbf{T}}_{20}$; see Figure 3. However, due to the deflation of the $\mathbf{v}$ vector, from column $\mu = 8$ on, the lower bandwidth of $\mathbf{T}_{20}$ is reduced by one, which compensates for the "spike" in row 8 of $\widetilde{\mathbf{T}}_{20}$. In Table 1, the second row associated with $n = 11$ records the nonzero structure of column $\mu_{11} = 9$ of $\mathbf{T}_{20}$ and of column $\phi_{11} = 9$ of $\widetilde{\mathbf{T}}_{20}$. The banded parts of these columns start at position $l_{\mathbf{v}} = 7$ and $l_{\mathbf{w}} = 6$, respectively; see Figure 3.

If no deflation occurred at the n-th pass, then, in Table 1, there is only a single row associated with n. For example, consider row $n = 18$. The history indices $\mu_{18} = 16$ and $\phi_{18} = 17$ record that, at pass $n = 18$, column 16 of $\mathbf{T}_{20}$ and column 17 of $\widetilde{\mathbf{T}}_{20}$ were constructed. The banded part of both these columns starts with entries in row $l_{\mathbf{v}} = l_{\mathbf{w}} = 15$. Furthermore, since $\mathcal{I}_{\mathbf{v}} = \{12\}$ and $\mathcal{I}_{\mathbf{w}} = \{8\}$, there are potential nonzeros in row 12 of $\mathbf{T}_{20}$ and row 8 of $\widetilde{\mathbf{T}}_{20}$, due to earlier deflations.

In the general case, where look-ahead may occur, the structure of $\mathbf{T}_{20}$ and $\widetilde{\mathbf{T}}_{20}$ is completely described by the data listed in the first three and the last six columns of Table 1. Recall from §4.5 that, in the look-ahead case, $l_{\mathbf{v}}$, $l_{\mathbf{w}}$, $\mathcal{I}_{\mathbf{v}}$, and $\mathcal{I}_{\mathbf{w}}$ record the indices of look-ahead clusters. For example, consider again row $n = 18$ in Table 1. In the look-ahead case, we have $l_{\mathbf{v}} = l_{\mathbf{w}} = 11$. This means that the banded part of both column $\mu_{18} = 16$ of $\mathbf{T}_{20}$ and column $\phi_{18} = 17$ starts with entries in row $n_{11} = 15$. Furthermore, since $\mathcal{I}_{\mathbf{v}} = \{8\}$ and $\mathcal{I}_{\mathbf{w}} = \{6\}$, there are additional potential nonzero entries in row $n_8 = 11$ of $\mathbf{T}_{20}$ and rows $n_6 = 8, 9, 10$ of $\widetilde{\mathbf{T}}_{20}$; see Figure 3. Note that since the 6-th cluster corresponds to a true look-ahead cluster of length 3, the "spike" in row 8 of $\widetilde{\mathbf{T}}_{20}$ in the no-look-ahead case has spread to "spikes" in rows 8, 9, and 10 in the look-ahead case.

5. The Lanczos-type algorithm with deflation and look-ahead

In this section, we present a statement of the algorithm in its most general form, with deflation and look-ahead, and discuss a few implementation issues.

5.1. A complete statement of the algorithm. Recall that, in Figure 1, we showed the basic structure of the algorithm. The step numbers from Figure 1 match those used in the following statement of the algorithm.

Algorithm 5.1. (Lanczos-type method with deflation and look-ahead.)
INPUT: Matrix $\mathbf{A} \in \mathbb{C}^{N\times N}$;
  m right starting vectors $\mathbf{r}_1, \mathbf{r}_2, \ldots, \mathbf{r}_m \in \mathbb{C}^N$;
  p left starting vectors $\mathbf{l}_1, \mathbf{l}_2, \ldots, \mathbf{l}_p \in \mathbb{C}^N$;
  Deflation tolerance $\mathtt{dtol} \geq 0$. (Set $\mathtt{dtol} = 0$ for exact deflation only.)

INDICES AND INDEX SETS:

- μ is the index of the $\mathbf{v}$ vector currently being expanded; ϕ is the index of the $\mathbf{w}$ vector currently being expanded. (A non-positive μ or ϕ means we are still fetching starting vectors.)
- $\mathcal{I}_{\mathbf{v}}$ and $\mathcal{I}_{\mathbf{w}}$ record indices of clusters that must be preserved due to inexact deflation. If $k \in \mathcal{I}_{\mathbf{v}}$ or $k \in \mathcal{I}_{\mathbf{w}}$ then $\mathbf{v}$ or $\mathbf{w}$ needs to be biorthogonalized against the entire earlier clusters $\mathbf{W}^{(k)}$ or $\mathbf{V}^{(k)}$, respectively.
- l is the index of the currently constructed clusters $\mathbf{V}^{(l)}$ and $\mathbf{W}^{(l)}$. n_l is the index of the first pair of Lanczos vectors in $\mathbf{V}^{(l)}$ and $\mathbf{W}^{(l)}$.

0) (Initialization)
Set $\mu = \mu_0 = -m$ *and* $\phi = \phi_0 = -p$.
Set $\mathcal{I}_{\mathbf{v}} = \emptyset$ *and* $\mathcal{I}_{\mathbf{w}} = \emptyset$.
Set $l = 1$, $n_l = 1$, *and* $\mathbf{V}^{(l)} = \mathbf{W}^{(l)} = \emptyset$.

For $n = 1, 2, \ldots$, *build* n*-th pair of Lanczos vectors* $\mathbf{v}_n$ *and* $\mathbf{w}_n$:

1) (Build the unnormalized right Lanczos vector $\mathbf{v}$.)
1a) *Set* $\mu = \mu + 1$.
1b) *If* $\mu = n$, *then stop.* (The right block Krylov subspace is exhausted.)
1c) (Advance the right block Krylov subspace.)
Set

$$\mathbf{v} = \begin{cases} \mathbf{r}_{\mu+m} & \text{if } \mu \le 0, \\ \mathbf{A}\,\mathbf{v}_\mu & \text{if } \mu > 0. \end{cases} \tag{5.1}$$

If $\mathbf{V}^{(l)} \neq \emptyset$, *optionally set*

$$\mathbf{v} = \mathbf{v} - \mathbf{V}^{(l)} \left[t_{i,\mu}\right]_{n_l \le i < n}, \quad \text{with arbitrary} \quad t_{i,\mu} \in \mathbb{C}. \tag{5.2}$$

(We may add arbitrary combinations of vectors in the current cluster $\mathbf{V}^{(l)}$, if nonempty.)

1d) (Determine against which clusters $\mathbf{v}$ needs to be biorthogonalized.)
If $\mu > 0$, *set* $l(\mu) = \max\{\, i \in \mathbb{N} \mid n_i \le \mu \,\}$.
Set

$$l_{\mathbf{v}} = \begin{cases} \max\{\, i \in \mathbb{N} \mid n_i \le \phi_{n_{l(\mu)}} \,\} & \text{if } \mu > 0 \text{ and } \phi_{n_{l(\mu)}} > 0, \\ 1 & \text{otherwise}, \end{cases} \tag{5.3}$$

and define the temporary index set

$$\mathcal{I} = \{\, l_{\mathbf{v}}, l_{\mathbf{v}} + 1, \ldots, l - 1 \,\} \cup \bigcup_{\substack{k \in \mathcal{I}_{\mathbf{v}} \\ k < l_{\mathbf{v}}}} \{k\}. \tag{5.4}$$

1e) (Biorthogonalize $\mathbf{v}$ against these clusters.)
Compute the coefficient vectors

$$\left[t_{i,\mu}\right]_{n_k \le i < n_{k+1}} = \left(\mathbf{\Delta}^{(k)}\right)^{-1} \left(\mathbf{W}^{(k)}\right)^{\mathrm{T}} \mathbf{v} \quad \text{for all} \quad k \in \mathcal{I}. \tag{5.5}$$

Set

$$\mathbf{v} = \mathbf{v} - \sum_{k \in \mathcal{I}} \mathbf{V}^{(k)} \left[t_{i,\mu}\right]_{n_k \le i < n_{k+1}}. \tag{5.6}$$

1f) (Decide if $\mathbf{v}$ should be deflated.)
(i) (If $n_l \le n - 1$, the current cluster $\mathbf{V}^{(l)}$ is nonempty, and we first orthogonalize, in the ordinary one-sided sense, $\mathbf{v}$ against $\mathbf{V}^{(l)}$.)
If $n_l \le n - 1$, *then for* $i = n_l, n_l + 1, \ldots, n - 1$, *set*

$$\tau_{i,\mu} = \frac{\mathbf{v}_i^{\mathrm{H}}\,\mathbf{v}}{\|\mathbf{v}_i\|^2}, \quad \mathbf{v} = \mathbf{v} - \mathbf{v}_i\,\tau_{i,\mu}, \quad \text{and} \quad t_{i,\mu} = t_{i,\mu} + \tau_{i,\mu}. \tag{5.7}$$

(ii) *Check if* $\|\mathbf{v}\| \le$ `dtol`.
If no, continue with Step 2).
If yes, deflate $\mathbf{v}$ *by doing the following* :

(iii) *If* $\mu > 0$ *and* $\mathbf{v} \neq \mathbf{0}$, *then set* $\mathcal{I}_{\mathbf{w}} = \mathcal{I}_{\mathbf{w}} \cup \{\, l(\mu) \,\}$, *and save the clusters* $\mathbf{V}^{(l(\mu))}$ *and* $\mathbf{W}^{(l(\mu))}$.
(The vector $\mathbf{v}$ is the μ-th column of the matrix $\mathbf{V}_{\mu}^{\mathrm{dl}}$ in (4.6).)

(iv) *Repeat all of Step* 1).

2) (Build the unnormalized left Lanczos vector $\mathbf{w}$.)

2a) *Set* $\phi = \phi + 1$.

2b) *If* $\phi = n$, *then stop.* (The left block Krylov subspace is exhausted.)

2c) (Advance the left block Krylov subspace.)
Set

$$\mathbf{w} = \begin{cases} \mathbf{l}_{\phi+p} & \text{if } \phi \leq 0, \\ \mathbf{A}^{\mathrm{T}}\, \mathbf{w}_{\phi} & \text{if } \phi > 0. \end{cases} \tag{5.8}$$

If $\mathbf{W}^{(l)} \neq \emptyset$, *optionally set*

$$\mathbf{w} = \mathbf{w} - \mathbf{W}^{(l)} \left[\tilde{t}_{i,\phi}\right]_{n_l \leq i < n}, \quad \text{with arbitrary} \quad \tilde{t}_{i,\phi} \in \mathbb{C}. \tag{5.9}$$

(We may add arbitrary combinations of vectors in the current cluster $\mathbf{W}^{(l)}$, if nonempty.)

2d) (Determine against which clusters $\mathbf{w}$ needs to be biorthogonalized.)
If $\phi > 0$, *set* $l(\phi) = \max\{\, i \in \mathbb{N} \mid n_i \leq \phi \,\}$.
Set

$$l_{\mathbf{w}} = \begin{cases} \max\{\, i \in \mathbb{N} \mid n_i \leq \mu_{n_{l(\phi)}} \,\} & \text{if } \phi > 0 \text{ and } \mu_{n_{l(\phi)}} > 0, \\ 1 & \text{otherwise}, \end{cases}$$

and define the temporary index set

$$\mathcal{I} = \{\, l_{\mathbf{w}}, l_{\mathbf{w}} + 1, \ldots, l-1 \,\} \cup \bigcup_{\substack{k \in \mathcal{I}_{\mathbf{w}} \\ k < l_{\mathbf{w}}}} \{k\}. \tag{5.10}$$

2e) (Biorthogonalize $\mathbf{w}$ against these clusters.)
Compute the coefficient vectors

$$\left[\tilde{t}_{i,\phi}\right]_{n_k \leq i < n_{k+1}} = \left(\mathbf{\Delta}^{(k)}\right)^{-\mathrm{T}} \left(\mathbf{V}^{(k)}\right)^{\mathrm{T}} \mathbf{w} \quad \text{for all} \quad k \in \mathcal{I}. \tag{5.11}$$

Set

$$\mathbf{w} = \mathbf{w} - \sum_{k \in \mathcal{I}} \mathbf{W}^{(k)} \left[\tilde{t}_{i,\phi}\right]_{n_k \leq i < n_{k+1}}.$$

2f) (Decide if $\mathbf{w}$ should be deflated.)

(i) (If $n_l \leq n-1$, the current cluster $\mathbf{W}^{(l)}$ is nonempty, and we first orthogonalize, in the ordinary one-sided sense, $\mathbf{v}$ against $\mathbf{W}^{(l)}$.)
If $n_l \leq n-1$, *then for* $i = n_l, n_l+1, \ldots, n-1$, *set*

$$\tilde{\tau}_{i,\phi} = \frac{\mathbf{w}_i^{\mathrm{H}}\, \mathbf{w}}{\|\mathbf{w}_i\|^2}, \quad \mathbf{w} = \mathbf{w} - \mathbf{w}_i\, \tilde{\tau}_{i,\phi}, \quad \text{and} \quad \tilde{t}_{i,\phi} = \tilde{t}_{i,\phi} + \tilde{\tau}_{i,\phi}. \tag{5.12}$$

(ii) *Check if* $\|\mathbf{w}\| \leq$ `dtol`.
If no, continue with Step 3).
If yes, deflate $\mathbf{w}$ *by doing the following* :

(iii) *If* $\phi > 0$ *and* $\mathbf{w} \neq \mathbf{0}$, *then set* $\mathcal{I}_{\mathbf{v}} = \mathcal{I}_{\mathbf{v}} \cup \{\, l(\phi) \,\}$ *and save the clusters* $\mathbf{V}^{(l(\phi))}$ *and* $\mathbf{W}^{(l(\phi))}$.
(The vector $\mathbf{w}$ is the ϕ-th column of the matrix $\mathbf{W}_{\phi}^{\mathrm{dl}}$ in (4.6).)

(iv) *Repeat all of Step* 2).

3) (Normalize $\mathbf{v}$ and $\mathbf{w}$ to obtain the n-th pair of Lanczos vectors $\mathbf{v}_n$ and $\mathbf{w}_n$, and add them to the current clusters.)
Set

$$t_{n,\mu} = \|\mathbf{v}\|, \quad \tilde{t}_{n,\phi} = \|\mathbf{w}\|, \quad \mathbf{v}_n = \frac{\mathbf{v}}{t_{n,\mu}}, \quad \textit{and} \quad \mathbf{w}_n = \frac{\mathbf{w}}{\tilde{t}_{n,\phi}}. \tag{5.13}$$

Set $\mathbf{V}^{(l)} = \mathbf{V}^{(l)} \cup \{\mathbf{v}_n\}$ *and* $\mathbf{W}^{(l)} = \mathbf{W}^{(l)} \cup \{\mathbf{w}_n\}$.

4) (Record the n-th history indices.)
Set $\mu_n = \mu$ *and* $\phi_n = \phi$.

5) (Compute $\mathbf{\Delta}^{(l)}$ and check for end of look-ahead clusters.)
Form $\mathbf{\Delta}^{(l)} = \left(\mathbf{W}^{(l)}\right)^{\mathrm{T}} \mathbf{V}^{(l)}$.
If $\mathbf{\Delta}^{(l)}$ *is "sufficiently" nonsingular, then set*

$$l = l+1, \quad n_l = n+1, \quad \textit{and} \quad \mathbf{V}^{(l)} = \mathbf{W}^{(l)} = \emptyset.$$

(The current clusters are complete, so start new clusters in the next pass.)

Remark 5.2. Recall from (4.6) that the entries of the matrices $\mathbf{T}_\mu$ and $\widetilde{\mathbf{T}}_\phi$ contain the recurrence coefficients used to generate the Lanczos vectors. The nonzero entries $t_{i,j}$ and $\tilde{t}_{i,j}$ of these matrices are given by equations (5.2), (5.5)–(5.7), (5.9), and (5.11)–(5.13) in Algorithm 5.1. The matrices $\mathbf{T}_\mu$ and $\widetilde{\mathbf{T}}_\phi$ are given by

$$\begin{aligned} \mathbf{T}_\mu &= \begin{cases} [t_{i,j}]_{1\le i\le n,\, 1-m\le j\le \mu} & \text{if } \mu \le 0, \\ [t_{i,j}]_{1\le i\le n,\, 1\le j\le \mu} & \text{if } \mu > 0, \end{cases} \\ \widetilde{\mathbf{T}}_\phi &= \begin{cases} \left[\tilde{t}_{i,j}\right]_{1\le i\le n,\, 1-p\le j\le \phi} & \text{if } \phi \le 0, \\ \left[\tilde{t}_{i,j}\right]_{1\le i\le n,\, 1\le j\le \phi} & \text{if } \phi > 0, \end{cases} \end{aligned} \tag{5.14}$$

where all elements $t_{i,j}$ and $\tilde{t}_{i,j}$ that are not explicitly defined in Algorithm 5.1 are set to be zero.

Remark 5.3. For $\mu > 0$, $\mathbf{T}_\mu$ is an $n \times \mu$ matrix, where $n > \mu$. In some applications, such as the BL-QMR method for the solution of linear systems with multiple right-hand sides, the rectangular matrix $\mathbf{T}_\mu$ is used; see [18]. For eigenvalue computations and Padé approximation of matrix-valued transfer functions, only the $\mu \times \mu$ *square part* $\mathbf{T}_\mu^{(s)} := [t_{i,j}]_{1\le i,j\le \mu}$ of $\mathbf{T}_\mu$ is used.

Remark 5.4. Algorithm 5.1 simplifies considerably if no deflation and/or no look-ahead occurs. In the absence of look-ahead, all clusters consist of single vectors only, and thus $n_l = l$ for all l. If, in addition, no deflation occurs, then the index sets (5.4) and (5.10) are identical and given by

$$\mathcal{I} = \{\, l_{\mathbf{v}}, l_{\mathbf{v}}+1, \ldots, n-1 \,\} \quad \text{where} \quad l_{\mathbf{v}} = \max\{\, 1,\ n-m-p \,\}.$$

This means that in the special case of no deflation and no look-ahead, each Lanczos vector has to be explicitly biorthogonalized only against the last $m+p$ vectors; see the discussion in §4.2. A detailed statement of a version of Algorithm 5.1 with deflation, but without look-ahead, is given in [2].

5.2. **Implementation details.** In this section, we discuss a few implementation details for Algorithm 5.1.

Steps 1e) and 2e) implement a classical two-sided Gram-Schmidt biorthogonalization, but in practice a "modified" two-sided Gram-Schmidt process would be preferred; see, e.g., [32]. For example, for the "modified" version of Step 1e), one simply replaces (5.5) and (5.6) by the following block-by-block update.

For all $k \in \mathcal{I}$ (in increasing order), set

$$\left[t_{i,\mu}\right]_{n_k \le i < n_{k+1}} = \left(\mathbf{\Delta}^{(k)}\right)^{-1} \left(\mathbf{W}^{(k)}\right)^{\mathrm{T}} \mathbf{v} \quad \textit{and} \quad \mathbf{v} = \mathbf{v} - \mathbf{V}^{(k)} \left[t_{i,\mu}\right]_{n_k \le i < n_{k+1}}.$$

The "modified" version of Step 2e) is analogous.

The computational work for obtaining the recurrence coefficients $t_{i,\mu}$ and $\tilde{t}_{i,\phi}$ in Steps 1e) and 2e) is dominated by the vector products $\left(\mathbf{W}^{(k)}\right)^{\mathrm{T}} \mathbf{v}$ and $\left(\mathbf{V}^{(k)}\right)^{\mathrm{T}} \mathbf{w}$. Roughly half of these vector products can be eliminated by exploiting relations that connect the coefficient matrices $\mathbf{T}_\mu$ and $\widetilde{\mathbf{T}}_\phi$. For example, in the simplest case of no deflation and no look-ahead, the square parts $\mathbf{T}_\mu^{(s)}$ and $\widetilde{\mathbf{T}}_\mu^{(s)}$ of these matrices are connected by $\left(\widetilde{\mathbf{T}}_\mu^{(s)}\right)^{\mathrm{T}} = \mathbf{\Delta}_\mu \, \mathbf{T}_\mu \, \mathbf{\Delta}_\mu^{-1}$ where $\mathbf{\Delta}_\mu$ is a diagonal matrix. A similar relation holds in the general case. Exploiting the connection between $\mathbf{T}_\mu$ and $\widetilde{\mathbf{T}}_\phi$ not only halves the number of vector products required in Steps 1e) and 2e), but, as in the case of the classical Lanczos algorithm, it usually also enhances the overall numerical stability of Algorithm 5.1.

In Step 5), one needs to decide if the current look-ahead cluster can be closed. In view of the nonsingularity of the $\mathbf{\Delta}^{(k)}$ in (3.11), a necessary condition for closing the look-ahead cluster is that the matrix $\mathbf{\Delta}^{(l)}$ is nonsingular. It is thus tempting to base the look-ahead strategy solely on a measure of singularity of $\mathbf{\Delta}^{(l)}$, such as the smallest singular value. However, as was illustrated in [17] for the Lanczos algorithm with single starting vectors, such a look-ahead strategy is not appropriate and does not lead to a robust algorithm. Instead, a reliable look-ahead strategy needs to check the singularity of $\mathbf{\Delta}^{(l)}$, as well as the sizes of the recurrence coefficients $t_{i,\mu}$ and $\tilde{t}_{i,\phi}$ in (5.5) and (5.11) relative to some estimate, $\mathtt{nest}(\mathbf{A})$, for $\|\mathbf{A}\|$. More precisely, the current look-ahead cluster should only be closed if

$$|t_{i,\mu}|, \ \left|\tilde{t}_{i,\phi}\right| \le \mathtt{fac} \times \mathtt{nest}(\mathbf{A}) \quad \text{for all} \quad i. \tag{5.15}$$

Here, `fac` is an appropriate constant, typically $\mathtt{fac} = 10$, and if not available a priori, the norm estimate $\mathtt{nest}(\mathbf{A})$ can be obtained easily during the first few passes through the algorithm. The check (5.15) guarantees that the component $\mathbf{A}\,\mathbf{v}_\mu$ or $\mathbf{A}^{\mathrm{T}}\,\mathbf{w}_\phi$ (respectively) of the new Lanczos vector is not dominated by the previous Lanczos vectors. Note that $\mathbf{A}\,\mathbf{v}_\mu$ and $\mathbf{A}^{\mathrm{T}}\,\mathbf{w}_\phi$ are the only parts of $\mathbf{v}$ and $\mathbf{w}$, respectively, that advance the block Krylov subspaces.

6. Properties of the algorithm

In this section, we establish some theoretical properties of the Lanczos vectors. Throughout this section, we assume that the Lanczos vectors are generated by running Algorithm 5.1 in exact arithmetic. A round-off error analysis of the algorithm is beyond the scope of this paper.

First we show that the Lanczos vectors are indeed cluster-wise biorthogonal, as stated in §3.3, and satisfy the recurrences stated in §4.5.

Theorem 6.1. *The Lanczos-type Algorithm* 5.1 *generates two sequences* (3.1) *of right and left Lanczos vectors that satisfy the cluster-wise biorthogonality condition* (3.11). *The algorithm also generates matrices of coefficients* $\mathbf{T}_\mu$ *and* $\widetilde{\mathbf{T}}_\phi$ *such that the recurrences* (4.6) *are satisfied.*

Proof. The recurrences (4.6) follow directly by summarizing equations (5.1), (5.2), (5.5)–(5.9), and (5.11)–(5.13), which are used to generate the Lanczos vectors, in compact matrix form, using the notation introduced in (3.13), (4.7), and (5.14).

Next, we prove the cluster-wise biorthogonality condition (3.11) by showing that each n-th pair of Lanczos vectors $\mathbf{v}_n$ and $\mathbf{w}_n$ satisfies

$$\left(\mathbf{W}^{(j)}\right)^{\mathrm{T}} \mathbf{v}_n = \mathbf{0} \quad \text{and} \quad \left(\mathbf{V}^{(j)}\right)^{\mathrm{T}} \mathbf{w}_n = \mathbf{0} \quad \text{for all} \quad j = 1, 2, \ldots, l(n) - 1. \tag{6.1}$$

Recall that, by (3.12), $l(n)$ is defined as the index of the clusters that contain $\mathbf{v}_n$ and $\mathbf{w}_n$. We prove (6.1) by induction on n. Since $l(1) = 1$, there is nothing to show for $n = 1$. Now let $n \geq 2$, and assume, as induction hypothesis, that (6.1) holds true for all i-th Lanczos vectors with $i \leq n - 1$. We need to show that the n-th Lanczos vectors satisfy (6.1). We will only establish the conditions for $\mathbf{v}_n$ in (6.1); the proof of the conditions for $\mathbf{w}_n$ is completely analogous and thus omitted. Note that, in view of (5.2), (5.6), (5.7), and (5.13), we have

$$\mathbf{v}_n = \frac{1}{t_{n,\mu}} \left(\mathbf{v} - \sum_{i=n_l}^{n-1} \mathbf{v}_i \, t_{i,\mu} - \sum_{k \in \mathcal{I}} \mathbf{V}^{(k)} \left[t_{i,\mu} \right]_{n_k \leq i < n_{k+1}} \right), \tag{6.2}$$

where $\mathbf{v}$ is given by (5.1) and $\left[t_{i,\mu} \right]_{n_k \leq i < n_{k+1}}$ by (5.5). Here, for simplicity, we have set $\mu := \mu_n$. Using the induction hypothesis and (5.5), it follows from (6.2) that

$$\left(\mathbf{W}^{(j)}\right)^{\mathrm{T}} \mathbf{v}_n = \frac{1}{t_{n,\mu}} \left(\left(\mathbf{W}^{(j)}\right)^{\mathrm{T}} \mathbf{v} - \left(\mathbf{W}^{(j)}\right)^{\mathrm{T}} \mathbf{V}^{(j)} \left(\boldsymbol{\Delta}^{(j)}\right)^{-1} \left(\mathbf{W}^{(j)}\right)^{\mathrm{T}} \mathbf{v} \right) = \mathbf{0} \tag{6.3}$$

for all $j \in \mathcal{I}$. Here, $\mathcal{I}$ is the set of cluster indices given by (5.4) (with $l = l(n)$). By (6.3), we have established (6.1) for all $j \in \mathcal{I}$. If $\mathcal{I}$ contains all the cluster indices $1 \leq j < l(n)$, the proof is complete. Otherwise, it remains to show that

$$\left(\mathbf{W}^{(j)}\right)^{\mathrm{T}} \mathbf{v}_n = \mathbf{0} \tag{6.4}$$

for all indices $1 \leq j < l(n)$ with $j \notin \mathcal{I}$. Let j be any such index. By (5.4), $j \notin \mathcal{I}$ implies that $l_{\mathbf{v}} > 1$. By (5.3) and (5.1), it follows that $\mu > 0$ and $\mathbf{v} = \mathbf{A}\,\mathbf{v}_\mu$. Inserting $\mathbf{v} = \mathbf{A}\,\mathbf{v}_\mu$ into (6.2) and using the induction hypothesis, we get

$$t_{n,\mu} \left(\mathbf{W}^{(j)}\right)^{\mathrm{T}} \mathbf{v}_n = \left(\mathbf{W}^{(j)}\right)^{\mathrm{T}} \mathbf{A}\,\mathbf{v}_\mu = \left(\mathbf{A}^{\mathrm{T}} \mathbf{W}^{(j)}\right)^{\mathrm{T}} \mathbf{v}_\mu.$$

It follows that (6.4) is satisfied if, and only if,

$$\mathbf{v}_\mu^{\mathrm{T}} \left(\mathbf{A}^{\mathrm{T}} \mathbf{w}_i\right) = 0 \quad \text{for all} \quad n_j \leq i < n_{j+1}. \tag{6.5}$$

In view of (5.8), each of the vectors $\mathbf{A}^{\mathrm{T}} \mathbf{w}_i$ in (6.5) is used as a candidate $\mathbf{w}$ for a left Lanczos vector in Algorithm 5.1. After $\mathbf{w}$ has been biorthogonalized, the resulting vector $\mathbf{w}$ is either deflated in Step 2f), or normalized to become a left Lanczos vector, say $\mathbf{w}_q$, where q is the unique index such that $i = \phi_q$. However, since $j \notin \mathcal{I}$, the first case can be excluded. Indeed, if $\mathbf{w}$ was deflated, then the cluster index j would have been added to $\mathcal{I}_{\mathbf{v}}$ in Step 2f), and by (5.4), it would follow that $j \in \mathcal{I}$. In the second case, in view of the recurrences (4.6), we have

$$\mathbf{A}^{\mathrm{T}}\,\mathbf{w}_i = \mathbf{A}^{\mathrm{T}}\,\mathbf{w}_{\phi_q} = \sum_{k=1}^{q} \mathbf{w}_k\,\tilde{t}_{k,\phi_q}. \tag{6.6}$$

By (5.4), $j \notin \mathcal{I}$ implies $j+1 \le l_{\mathbf{v}}$. Using the definition of $l_{\mathbf{v}}$ in (5.3), we obtain $\phi_q = i < n_{j+1} \le \phi_{n_{l(\mu)}}$. It follows that $q < n_{l(\mu)}$ and, further, $l(q) \le l(\mu)-1$, where $\mu \le n-1$. We can thus employ the induction hypothesis to deduce that $\mathbf{w}_k^{\mathrm{T}}\,\mathbf{v}_\mu = 0$ for all $k \le q$. Finally, inserting (6.6) into (6.5), we get

$$\mathbf{v}_\mu^{\mathrm{T}}\left(\mathbf{A}^{\mathrm{T}}\,\mathbf{w}_i\right) = \sum_{k=1}^{q} \mathbf{v}_\mu^{\mathrm{T}}\,\mathbf{w}_k\,\tilde{t}_{k,\phi_q} = \sum_{k=1}^{q}\left(\mathbf{w}_k^{\mathrm{T}}\,\mathbf{v}_\mu\right)\,\tilde{t}_{k,\phi_q} = 0,$$

which shows that (6.5) holds. Therefore, the proof of the theorem is complete. □

Remark 6.2. The result of Theorem 6.1 is correct independent of the choice of the deflation tolerance `dtol` and the look-ahead strategy. In fact, Algorithm 5.1 will work even if a large `dtol` is chosen and if look-head is carried out when unnecessary.

Next, we show that for exact deflation, the Lanczos vectors indeed span the right and left block Krylov subspaces.

Theorem 6.3. *If the Lanczos-type Algorithm* 5.1 *is run with exact deflation only, i.e., with deflation tolerance* `dtol` $= 0$*, then the generated Lanczos vectors* (3.1) *span the right and left block Krylov subspaces, i.e., they satisfy* (3.2).

Proof. We use induction on n to show that

$$\operatorname{span}\{\,\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_n\,\} = \mathcal{K}_n(\mathbf{A}, \mathbf{R}) \quad \text{for all} \quad n \ge 0. \tag{6.7}$$

The proof of the corresponding property of the left Lanczos vectors is analogous, and is therefore omitted. For $n = 0$, there is nothing to show. Now let $n \ge 1$, and assume, as induction hypothesis, that (6.7) holds true for $n-1$. We denote by $\widehat{\mathbf{v}}$ the vector (5.1) in Algorithm 5.1, and by $\mathbf{v}$ the vector that has resulted from $\widehat{\mathbf{v}}$ after the biorthogonalization in Step 1e) and the possible orthogonalization in Step 1f)(i) has been performed. Note that, by (5.6) and (5.7), $\mathbf{v}$ is of the form

$$\mathbf{v} = \widehat{\mathbf{v}} + \sum_{i=1}^{n-1} \mathbf{v}_i\,\alpha_i, \quad \text{where} \quad \alpha_i \in \mathbb{C}. \tag{6.8}$$

In view of (2.1) and (2.2), the next candidate vector to be added to $\mathcal{K}_{n-1}(\mathbf{A}, \mathbf{R})$ is of the form $\mathbf{A}^j\,\mathbf{r}_i$. Using (5.1) and the induction hypothesis, it follows that

$$\widehat{\mathbf{v}} = \gamma\,\mathbf{A}^j\,\mathbf{r}_i + \sum_{i=1}^{n-1} \mathbf{v}_i\,\beta_i, \quad \text{where} \quad \beta_i, \gamma \in \mathbb{C} \quad \text{and} \quad \gamma \ne 0. \tag{6.9}$$

Next, we show that

$$\mathbf{v} = \mathbf{0} \qquad \Longleftrightarrow \qquad \mathbf{A}^j\,\mathbf{r}_i \in \mathcal{K}_{n-1}(\mathbf{A}, \mathbf{R}). \tag{6.10}$$

If $\mathbf{v} = \mathbf{0}$, then, by (6.8) and (6.9), the vector $\mathbf{A}^j\,\mathbf{r}_i$ is a linear combination of $\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_{n-1}$, and, by the induction hypothesis, it is contained in $\mathcal{K}_{n-1}(\mathbf{A}, \mathbf{R})$. Conversely, if $\mathbf{A}^j\,\mathbf{r}_i \in \mathcal{K}_{n-1}(\mathbf{A}, \mathbf{R})$, then, in view of (6.8) and (6.9), we have

$$\mathbf{v} = \sum_{i=1}^{n-1} \mathbf{v}_i\,\gamma_i, \quad \text{where} \quad \gamma_i \in \mathbb{C}. \tag{6.11}$$

Recall that $\mathbf{v}$ is constructed to be biorthogonal to the clusters of left Lanczos vectors $\mathbf{W}^{(k)}$ for all $1 \leq k < l(n)$. Multiplying (6.11) from the left by $\left(\mathbf{W}^{(k)}\right)^{\mathrm{T}}$, it follows that $\gamma_i = 0$ for all $1 \leq i < n_{l(n)}$. Moreover, if $n_{l(n)} \leq n-1$, then in Step 1f)(i), $\mathbf{v}$ is also orthogonalized against $\mathbf{v}_i$ for all $n_{l(n)} \leq i \leq n-1$, and by multiplying (6.11) from the left by $\mathbf{v}_i^{\mathrm{H}}$, it follows that also $\gamma_i = 0$ for all $n_{l(n)} \leq i \leq n-1$. Thus all $\gamma_i = 0$ in (6.11) and so $\mathbf{v} = \mathbf{0}$. Note that, since we are assuming exact deflation only, $\mathbf{v}$ is turned into the n-th Lanczos vector $\mathbf{v}_n$ if, and only if, $\mathbf{v} \neq \mathbf{0}$. In view of (6.8)–(6.10), it follows that $\mathbf{v}_n \in \mathcal{K}_n(\mathbf{A}, \mathbf{R})$, but $\mathbf{v}_n \notin \mathcal{K}_{n-1}(\mathbf{A}, \mathbf{R})$. Together with the induction hypothesis, this shows that (6.7) holds true. $\square$

7. Concluding remarks

We presented an extension of the classical Lanczos process for single starting vectors to multiple starting vectors.

The objective of this paper is to describe the algorithm and to prove some of its key properties. In order to keep the length of the paper reasonable, we decided not to include numerical examples. Applications of the algorithm to the problems mentioned in §1.2 and numerical results will be and, in part, already have been reported elsewhere. Freund and Malhotra [18] developed a block version of QMR, called BL-QMR[1], for the solution of multiple linear systems (1.1) that uses a version of Algorithm 5.1 with deflation, but without look-ahead. Numerical experiments with the BL-QMR algorithm are reported in [18, 29]. The results in [18, 29] clearly illustrate the importance of deflation. More precisely, basis vectors do become almost linearly dependent in several of the numerical examples in [18, 29], yet BL-QMR converges as long as these vectors are deflated properly. However, as soon as the deflation procedure is turned off, BL-QMR fails to converge. In [13], Feldmann and Freund use an early version (without deflation and without look-ahead) of the algorithm to compute Padé approximants to matrix-valued transfer functions (1.5), and report numerical results for problems arising in circuit simulation.

Finally, we stress that we are well aware of the connections between the problem of constructing suitable basis vectors for block Krylov subspaces, and the related problems of solving block Hankel systems and constructing matrix-Padé approximations; see, e.g., [7, 8, 34, 40, 41] and the references given therein. The connections between the proposed Lanczos-type algorithm and these related problems should be explored further. A first such result on the connection to matrix-Padé approximation was given by Freund [15].

Acknowledgments

The authors are grateful to the referee and editor for their constructive comments that greatly improved the presentation of the paper. The third author would like to thank Jane Cullum for pointing out the deflation procedure in [9].

References

[1] J. I. Aliaga, *Algoritmos paralelos basados en el método de Lanczos. Aplicaciones en problemas de control*, Doctoral Thesis, Departamento de Sistemas Informáticos y Computación, Universidad Politécnica de Valencia, Valencia, Spain, 1995.

[1] FORTRAN-77 and MATLAB implementations of BL-QMR are available online under a limited-use license agreement from `http://www.bell-labs.com/topic/limited-dist/`.

[2] J. I. Aliaga, D. L. Boley, R. W. Freund, and V. Hernández, *A Lanczos-type algorithm for multiple starting vectors*, Numerical Analysis Manuscript No. 96–18, Bell Laboratories, Murray Hill, NJ, 1996. Also available online from `http://cm.bell-labs.com/cs/doc/96`.

[3] J. I. Aliaga, D. L. Boley, and V. Hernández, *A block clustered Lanczos algorithm*, Presentation at the workshop on "Numerical Linear Algebra with Applications", Oberwolfach, Germany, April 1994.

[4] Z. Bai, D. Day, and Q. Ye, *ABLE: an adaptive block Lanczos method for non-Hermitian eigenvalue problems*, Research Report 95-04, Department of Mathematics, University of Kentucky, Lexington, KY, 1995.

[5] D. L. Boley, *Krylov space methods on state-space control models*, Circuits Systems Signal Process. **13** (1994), 733–758.

[6] D. L. Boley and G. H. Golub, *The nonsymmetric Lanczos algorithm and controllability*, Systems Control Lett. **16** (1991), 97–105.

[7] O. H. Bosgra and A. J. J. Van der Weiden, *Input-output invariants for linear multivariable systems*, IEEE Trans. Automat. Control **AC-25** (1980), 20–36.

[8] A. Bultheel, *Recursive algorithms for the matrix Padé problem*, Math. Comp. **35** (1980), 875–892.

[9] J. K. Cullum and W. E. Donath, *A block Lanczos algorithm for computing the q algebraically largest eigenvalues and a corresponding eigenspace for large, sparse symmetric matrices*, Proc. 1974 IEEE Conference on Decision and Control, IEEE Press, New York, 1974, pp. 505–509.

[10] J. K. Cullum and R. A. Willoughby, *Lanczos algorithms for large symmetric eigenvalue computations, Volume 1, Theory*, Birkhäuser, Basel, 1985.

[11] ———, *A practical procedure for computing eigenvalues of large sparse nonsymmetric matrices*, Large Scale Eigenvalue Problems (J. Cullum and R. A. Willoughby, eds.), North-Holland, Amsterdam, The Netherlands, 1986, pp. 193–240.

[12] P. Feldmann and R. W. Freund, *Efficient linear circuit analysis by Padé approximation via the Lanczos process*, IEEE Trans. Computer-Aided Design **14** (1995), 639–649.

[13] ———, *Reduced-order modeling of large linear subcircuits via a block Lanczos algorithm*, Proc. 32nd Design Automation Conference, ACM, New York, 1995, pp. 474–479.

[14] R. W. Freund, *The look-ahead Lanczos process for nonsymmetric matrices and its applications*, Proceedings of the Cornelius Lanczos International Centenary Conference (J. D. Brown, M. T. Chu, D. C. Ellison, and R. J. Plemmons, eds.), SIAM, Philadelphia, 1994, pp. 33–47.

[15] ———, *Computation of matrix Padé approximations of transfer functions via a Lanczos-type process*, Approximation Theory VIII, Vol. 1: Approximation and Interpolation (C. K. Chui and L. L. Schumaker, eds.), World Scientific Publishing Co., Inc., Singapore, 1995, pp. 215–222.

[16] R. W. Freund and P. Feldmann, *Efficient circuit analysis by Padé approximation via the Lanczos process*, Presentation at the workshop on "Numerical Linear Algebra with Applications", Oberwolfach, Germany, April 1994.

[17] R. W. Freund, M. H. Gutknecht, and N. M. Nachtigal, *An implementation of the look-ahead Lanczos algorithm for non-Hermitian matrices*, SIAM J. Sci. Comput. **14** (1993), 137–158.

[18] R. W. Freund and M. Malhotra, *A block QMR algorithm for non-Hermitian linear systems with multiple right-hand sides*, Linear Algebra Appl. **254** (1997), 119–157.

[19] R. W. Freund and N. M. Nachtigal, *QMR: a quasi-minimal residual method for non-Hermitian linear systems*, Numer. Math. **60** (1991), 315–339.

[20] ———, *An implementation of the QMR method based on coupled two-term recurrences*, SIAM J. Sci. Comput. **15** (1994), 313–337.

[21] G. H. Golub and R. Underwood, *The block Lanczos method for computing eigenvalues*, Mathematical Software III (J. R. Rice, ed.), Academic Press, New York, 1977, pp. 361–377.

[22] W. B. Gragg, *Matrix interpretations and applications of the continued fraction algorithm*, Rocky Mountain J. Math. **4** (1974), 213–225.

[23] W. B. Gragg and A. Lindquist, *On the partial realization problem*, Linear Algebra Appl. **50** (1983), 277–319.

[24] M. H. Gutknecht, *A completed theory of the unsymmetric Lanczos process and related algorithms, part I*, SIAM J. Matrix Anal. Appl. **13** (1992), 594–639.

[25] H. M. Kim and R. R. Craig, Jr., *Structural dynamics analysis using an unsymmetric block Lanczos algorithm*, Internat. J. Numer. Methods Engrg. **26** (1988), 2305–2318.

[26] ———, *Computational enhancement of an unsymmetric block Lanczos algorithm*, Internat. J. Numer. Methods Engrg. **30** (1990), 1083–1089.

[27] C. Lanczos, *An iteration method for the solution of the eigenvalue problem of linear differential and integral operators*, J. Res. Nat. Bur. Standards **45** (1950), 255–282.

[28] ———, *Solution of systems of linear equations by minimized iterations*, J. Res. Nat. Bur. Standards **49** (1952), 33–53.

[29] M. Malhotra, R. W. Freund, and P. M. Pinsky, *Iterative solution of multiple radiation and scattering problems in structural acoustics using a block quasi-minimal residual algorithm*, Comput. Methods Appl. Mech. Engrg. **146** (1997), 173–196.

[30] A. A. Nikishin and A. Yu. Yeremin, *Variable block CG algorithms for solving large sparse symmetric positive definite linear systems on parallel computers, I: general iterative scheme*, SIAM J. Matrix Anal. Appl. **16** (1995), 1135–1153.

[31] D. P. O'Leary, *The block conjugate gradient algorithm and related methods*, Linear Algebra Appl. **29** (1980), 293–322.

[32] B. N. Parlett, *Reduction to tridiagonal form and minimal realizations*, SIAM J. Matrix Anal. Appl. **13** (1992), 567–593.

[33] B. N. Parlett, D. R. Taylor, and Z. A. Liu, *A look-ahead Lanczos algorithm for unsymmetric matrices*, Math. Comp. **44** (1985), 105–124.

[34] J. Rissanen, *Algorithms for triangular decomposition of block Hankel and Toeplitz matrices with application to factoring positive matrix polynomials*, Math. Comp. **27** (1973), 147–154.

[35] A. Ruhe, *Implementation aspects of band Lanczos algorithms for computation of eigenvalues of large sparse symmetric matrices*, Math. Comp. **33** (1979), 680–687.

[36] T.-J. Su, *A decentralized linear quadratic control design method for flexible structures*, Ph.D. Thesis, Department of Aerospace and Engineering Mechanics, The University of Texas at Austin, Austin, TX, 1989.

[37] T.-J. Su and R. R. Craig, Jr., *Model reduction and control of flexible structures using Krylov vectors*, J. Guidance Control Dynamics **14** (1991), 260–267.

[38] D. R. Taylor, *Analysis of the look ahead Lanczos algorithm*, Ph.D. Thesis, Department of Mathematics, University of California, Berkeley, CA, 1982.

[39] R. Underwood, *An iterative block Lanczos method for the solution of large sparse symmetric eigenproblems*, Ph.D. Thesis, Computer Science Department, Stanford University, Stanford, CA, 1975.

[40] A. J. J. Van der Weiden and O. H. Bosgra, *The determination of structural properties of a linear multivariable system by operations of system similarity*, Internat. J. Control **32** (1980), 489–537.

[41] G.-L. Xu and A. Bultheel, *Matrix Padé approximation: definitions and properties*, Linear Algebra Appl. **137/138** (1990), 67–136.

(J. I. Aliaga) Departamento de Informática, Universidad Jaume I, Campus de Riu Sec, 12071 Castellón, Spain

E-mail address: aliaga@inf.uji.es

(D. L. Boley) Department of Computer Science and Engineering, University of Minnesota, 4-192 EE/CSci Building, 200 Union Street S.E., Minneapolis, Minnesota 55455–0159

E-mail address: boley@cs.umn.edu

(R. W. Freund) Bell Laboratories, Lucent Technologies, Room 2C–420, 700 Mountain Avenue, Murray Hill, New Jersey 07974–0636

E-mail address: freund@research.bell-labs.com

(V. Hernández) Departamento de Sistemas Informáticos y Computación, Universidad Politécnica de Valencia, Apartado de Correos 22012, 46071 Valencia, Spain

E-mail address: vhernand@dsic.upv.es