THE UNSYMMETRIC LANCZOS ALGORITHMS AND THEIR RELATIONS TO PADÉ APPROXIMATION, CONTINUED FRACTIONS, AND THE QD ALGORITHM

MARTIN H. GUTKNECHT*

Abstract. First, several algorithms based on the unsymmetric Lanczos process are surveyed: the biorthogonalization (BO) algorithm for constructing a tridiagonal matrix T similar to a given matrix A (whose extreme spectrum is sought typically); the "BOBC algorithm", which generates directly the LU factors of T; the BIORES (Lanczos/ORTHORES), BIOMIN (Lanczos/ORTHOMIN or biconjugate gradient (BCG)), and the BIODIR (Lanczos/ORTHODIR) algorithms for solving a nonsymmetric system of linear equations. The possibilities of breakdown in these algorithms are discussed and brought into relation.

Then the connections to formal orthogonal polynomials, Padé approximation, continued fractions, and the qd algorithm are reviewed. They allow us to deapen our understanding of breakdowns. Next, three types of (bi)conjugate gradient squared (CGS) algorithms are presented: BIORES^2, BIOMIN^2 (standard CGS), and BIODIR^2. Finally, fast Hankel solvers related to the Lanczos process are described.[1]

Key Words. Lanczos algorithm, biorthogonalisation algorithm, BO algorithm, biconjugate gradient algorithm, BCG algorithm, conjugate gradient squared, CGS algorithm, formal orthogonal polynomials, Padé approximation, continued fractions, quotient difference algorithm, qd algorithm, moment problem, fast Hankel solver.

AMS(MOS) subject classification. 65F10, 65F15; 30E05, 41A21, 65D05

1. Introduction.

In 1950 and 1952 C. Lanczos [38, 39] published two closely related articles. The first was on finding the eigenvalues and eigenvectors of a not necessarily symmetric matrix or operator and contains several related proposals for approaching such problems. In retrospective, it culminates in what he calls the *method of minimized iterations.* Later this method has been named the *Lanczos biorthogonalization (BO) algorithm* [52], but often it is just referred to as the Lanczos algorithm, in particular when it is applied to a symmetric matrix. However, Lanczos' proposal on how to use this method of minimized iterations differs considerably from the standard procedure that emerged from it. He considered it as an algorithm for the recursive construction of the minimal polynomial and of a finite sequence of vectors, from which then a set of eigenvectors is easily obtained. Today, however, the basic process is just used as an efficient way to produce a tridiagonal matrix, which, except for possibly heavy contamination by roundoff, is similar to the original matrix. The eigenvalues of the tridiagonal matrix are then obtained by other, more efficient and more stable methods. The columns of the similarity transformation matrix are biorthogonalized Krylov vectors, hence the matrix is determined by its first column and a set of polynomials. In practice, this matrix is hardly used, since for large problems, where the Lanczos algorithm is competitive, the memory space for this matrix is not available. It is one of the main features of the Lanczos process that even the given matrix A need not be stored, one just has to provide procedures for computing Az and $A^H z$ for an arbitrary vector z. (A^H denotes the conjugate transposed of A.) The symmetric Lanczos algorithm is a standard tool in many application areas where extreme eigenvalues of very large sparse Hermitian matrices have to be computed. (A number of such applications are described in [5]. For discussions of the nontrivial stability and implementation questions see [9, 42, 43, 44, 45, 47, 49, 48, 63, 64].)

* Interdisciplinary Project Center for Supercomputing, ETH Zurich, ETH-Zentrum, CH-8092 Zurich. mhg@ips.ethz.ch

[1] The sections on (bi)conjugate gradient squared algorithms and on fast Hankel solvers are not contained in this preliminary version.

The purpose of the second Lanczos paper on the subject was "to adopt the general principles of the previous investigation to the specific demands that arise if we are not intersted in the complete analysis of a matrix but only in the more special problem of obtaining the solution of a given set of linear equations (...)". The method of minimized iterations was modified slightly to produce *two* sets of polynomials instead of one. This *"complete algorithm for minimized iterations"* seems not to have found wide acceptance before a variant of it was popularized by Fletcher [15] as the *biconjugate gradient (BCG) method* more than 20 years later. But today it is considered as one of the most efficient iterative methods for nonsymmetric systems, and it is used in a variety of application areas.

When adapted to the problem of solving a positive definite symmetric system of linear equations the biorthogonalization algorithm becomes the (three-term) ORTHORES (or ORES) version of the *conjugate gradient (CG) method*, while the biconjugate gradient method reduces to the usual (two-term) ORTHOMIN (or OMIN) version of the CG method due to Hestenes and Stiefel [31]. (It is worth noting that although Stiefel had found the CG method independently in Zurich, the papers [39] and [31] were both worked out at the Los Angeles branch of the National Bureau of Standards, and [39] was completed half a year before [31]. However, the Lanczos paper, although providing all the relevant mathematical relations for the BCG method and, when specialized to the symmetric definite case, for the CG method, differs in the derivation of the algorithm completely and in its specification slightly from Hestenes and Stiefel [31] and from Fletcher [15]. Moreover, the Hestenes-Stiefel paper is far richer in results on the CG method and introduces also the more general conjugate direction (CD) methods, which have become an invaluable approach to defining variants of the CG method for linear *and* nonlinear systems.)

In the symmetric definite case, i.e., when the matrix is Hermitian and positive definite, the first set of vectors generated by the BO algorithm and the CG algorithm is orthogonal and, hence, the induced similarity transformation is orthogonal too; this mirrors the fact that there exists an orthogonal set of eigenvectors in this situation. In the symmetric indefinite case one has still formal orthogonality, but theoretically the CG algorithm may break down before the solution of the system of equations has been found. For the generalization to nonsymmetric systems a number of CG-related orthogonalization methods were investigated in the late seventies and early eighties [4, 11, 12, 14, 34, 57, 59, 70, 73]. On one hand, these algorithms can be classfied into the three types ORTHOMIN, ORTHORES, ORTHODIR [73], names which are now also used for the corresponding CG-variants for the symmetric and the symmetrizable case [36, 3] and are often abbreviated to OMIN, ORES, ODIR. On the other hand, depending on the inner product, the preconditioning, and, possibly, the restart or the truncation, there are many variants of these methods. If orthogonality is based on a definite inner product these methods tend to be stable, but in general they have the drawback to require in principal the orthogonalization of each new vector with *every* previous direction or residual, which is in practice impossible for large matrices (due to the limitations in computer time and memory). In other words, the induced similarity transformation is orthogonal, but brings the given matrix only into Hessenberg and not into tridiagonal form. However, truncated versions of these methods in which orthogonalization is limited to a certain number of previous vectors, are widely used. They have the drawback that theoretical convergence cannot be (or, at least, has not been) established and that the actual convergence behavior, although often good, is hard to predict.

In the seventies it was also shown that in certain situations orthogonalization with respect to two previous iterates (like in standard CG) is still sufficient [7, 71]. The interesting reverse question of finding necessary and sufficient conditions for this to be true was then completely resolved in the 1984 paper of Faber and Manteuffel [13].

In contrast to these orthogonalization methods the BO and the BCG algorithm produce biorthogonal sequences and, hence, in the nonsymmetric case, induce a nonorthogonal similarity transformation of the matrix; this reflects the fact that eigenvectors are nonorthogonal too. Unfortunately, such transformations involve the risk of numerical instability. On the other hand, they still produce a tridiagonal matrix (at least “almost always”, i.e., in the generic case), and thus they only require biorthogonalization with respect to two previous iterates.

In analogy to the three basic CG versions Omin, Ores and Odir [3], there are three iterative linear system solvers based on the nonsymmetric Lanczos process. They have been called Lanczos/Orthomin (= BCG), Lanczos/Orthores and Lanczos/Orthodir by Jea and Young [34], but we suggest the shorter and mathematically more appropriate names *BIORTHOMIN*, *BIORTHORES*, and *BIORTHODIR*, or, even briefer, *BIOMIN*, *BIORES*, *BIODIR*.

We start in Sect. 2 with a modern presentation of the BO algorithm, valid for non-Hermitian complex matrices, and discuss the various possible reasons for its breakdown. Then we describe its adaptation to systems of equations, the *BIORES* algorithm, which is seen to be endangered by an additional type of breakdown. We therefore present a new version of this algorithm, *unnormalized BIORES*, which avoids this breakdown. In Sect. 3 we turn to a slight generalization of the BCG algorithm. It is also first viewed as a method for producing a similarity transform, not as a linear equation solver. We call it in this form, which corresponds to Lanczos' [39] “complete algorithm for minimized iterations” the *BOBC algorithm* since it produces in addition to the biorthogonal pair of vector sequences a second, biconjugate pair of vector sequences. While the BO algorithm generates a tridiagonal matrix T similar to the given matrix A, the BOBC algorithm yields two bidiagonal matrices L and R, which are the LU factors of T, so that $T = LR$. This algorithm is then also adapted to linear equations. One obtains *BIOMIN* (i.e., BCG) and *unnormalized BIOMIN*, the two being theoretically equivalent to each other and to Biores, but in stability inferior to unnormalized Biores. Finally, still in Sect. 3, we discuss also Biodir, which too avoids one type of breakdown occuring in Biomin and Biores. The recurrence coefficients of Biodir are the elements of the tridiagonal matrix $T_1 := RL$ (or, possibly, of a diagonally rescaled matrix $T_1' = D_\Omega RLD_\Omega^{-1}$). More generally, if $T_k := R_{k-1}L_{k-1}$ has an LU decomposition $T_k = L_kR_k$ (and, in the mathematical sense, it almost always does), one can continue the process $T \to (L, R) \to T_1 \to (L_1, R_1) \to \cdots \to (L_k, R_k) \to T_{k+1}$. The transformation $T_{k-1} \mapsto T_k$ is one step of Rutishauser's *LR algorithm* applied to a tridiagonal matrix, and the map $(L_{k-1}, R_{k-1}) \mapsto (L_k, R_k)$ is also one step of Rutishauser's *progressive qd algorithm* [54][2].

The orthogonality properties of the (finite) sequences of vectors generated by the BO and BCG algorithms are mirrored by orthogonality properties of sequences of polynomials. If the matrix is not symmetric positive definite the underlying “inner product” may be indefinite though, but in the generic case we can assume that none of the constructed polynomials is orthogonal to itself. It is well-known that such sequences of *formal orthogonal polynomials* are the denominators of *Padé approximants* lying on a diagonal or a staircase of a Padé table and that these Padé approximants are also the convergents of *continued fractions*. In fact, Rutishauser [54] mentioned most of these connection already. He presented his progressive qd algorithm as a process for generating a sequence of continued fractions belonging to the various descending staircases in a normal Padé table. (The name Padé seems not to be mentioned in [54] however.) In Sects. 4–6 we review these connections and discuss the conclusions that can be drawn concerning the breakdown behavior of the algorithms of Sects. 2 and 3. Although

[2] [54] is a collection of three partly revised papers that were previously published in ZAMP in 1954 and 1955 and an appendix consisting of three new sections on related subjects. We always refer to [54] here, but to see the historical perspective the reader has to keep in mind that the material has been published earlier.

these connections are in principle well-known, they are mostly described in articles and books about Padé approximation, such as [6, 22], except for Rutishauser's original work [54], which requires the reading of several older sources. Here, we try to give a complete treatment of the most important facts; we hope that it appeals also to readers with little interest in Padé theory.

Our account of the qd algorithm is rather short, since there are Henrici's exellent descriptions of the classical theory [28, 29, 27, 30] and since the algorithm is of limited applicability in practice. Our aim is just to recall the well-known connections to the other sections.

In Section 7 we return to the problem of solving linear systems and bring the *conjugate gradient squared (CGS) algorithm*, which was recently proposed by Sonneveld [65], into the framework of Sects. 2–4. This algorithm has the features that it does not require A^H and that it generates approximants whose residual polynomials ρ_n^2 are the squares of the residual polynomials ρ_n of BIORES, BIOMIN, and BIODIR. Moreover, it makes use of the same recurrence coefficients as BIOMIN. We therefore call it here *BIOMIN*^{2}. It turns out that the same trick that allowed Sonneveld to derive BIOMIN^2 from BIOMIN can be applied to find two other, new methods: *BIORES* and *BIODIR*. The first comes again in two non-equivalent versions, normalized and unnormalized. The breakdown conditions are in each case the same as for the corresponding non-squared algorithm.

This paper is aimed at being a survey of the basic theory of the Lanczos process and the various algorithms that emerged from it, and at being a pointer to several subjects that are closely related to the Lanczos process. The intention is not at all to survey all the relevant work that has been done in this area. In particular, we do not discuss the important questions of numerical effects and actual implementation. We have also left out the connections to several subjects that belong to the same circle of ideas, such as, Gauss quadrature [21], modified moments [17, 18, 19], the moment problem [2], Vandermonde matrices [23], extrapolation methods [6, 62], the realization problem of linear system theory [24, 46], and the interpretation of conjugate direction methods as Gauss elimination [31, 67].

2. The Lanczos biorthogonalization algorithm. Let $A \in \mathbf{C}^{N\times N}$ be any complex $N \times N$ matrix, and let B be a nonsingular such matrix that commutes with A and is used to define the formal (i.e., not necessarily definite) inner product $< y, x >_B := y^H Bx$ of x and $y \in \mathbf{C}$. Orthogonality will usually be refered to with respect to this formal inner product. (For simplicity, we do *not* call it B-orthogonality.) However, we are most interested in the case where B is the identity I, and thus $< y, x >_B$ is the ordinary Euclidean inner product of $\mathbf{C}^N$. The case $B = A$ will also play a role.

Next, let x_0, $y_0 \in \mathbf{C}^N$ be two non-orthogonal initial vectors: $< y_0, x_0 >_B := y_0^H B x_0 \neq 0$. The *Lanczos biorthogonalization (BO) algorithm* [38, 39, 52] is a process for computing two finite sequences $\{x_n\}_{n=0}^{\nu-1}$ and $\{y_n\}_{n=0}^{\nu-1}$, whose length ν depends on A, x_0, and y_0, such that for $m, n = 0, 1, \ldots, \nu - 1$

$$\begin{aligned} x_n &\in \mathcal{K}_n := \text{span}\,(x_0, Ax_0, \ldots, A^n x_0), \\ y_m &\in \mathcal{L}_m := \text{span}\,(y_0, A^H y_0, \ldots, (A^H)^m y_0) \end{aligned} \tag{2.1}$$

and

$$< y_m, x_n >_B = \begin{cases} 0 & \text{if} \quad m \neq n, \\ \delta_n \neq 0 & \text{if} \quad m = n. \end{cases} \tag{2.2}$$

$\mathcal{K}_n$ and $\mathcal{L}_m$ are Krylov subspaces of A and A^H, respectively. The condition (2.2) means that the sequences are biorthogonal. If A is real symmetric and $y_0 = x_0 \in \mathbf{R}^N$, it will turn out that $y_n = x_n \in \mathbf{R}^N$, $n = 0, \dots, \nu - 1$; therefore $\{x_n\}_{n=0}^{\nu-1}$ is a sequence of mutually orthogonal vectors in this case.

Clearly $\mathcal{K}_n \supseteq \mathcal{K}_{n-1}$, $\mathcal{L}_n \supseteq \mathcal{L}_{n-1}$, and from (2.2) it follows that equality cannot hold when $n < \nu$ and that for $n = 1, \dots, \nu - 1$

$$x_n \in \mathcal{K}_n \backslash \mathcal{K}_{n-1}, \quad y_n \in \mathcal{L}_n \backslash \mathcal{L}_{n-1}, \tag{2.3}$$

$$\mathcal{K}_n = \text{span}\,(x_0, x_1, \dots, x_n), \quad \mathcal{L}_n = \text{span}\,(y_0, y_1, \dots, y_n), \tag{2.4}$$

and

$$x_n \perp_B \mathcal{L}_{n-1}, \quad y_n \perp_B \mathcal{K}_{n-1}. \tag{2.5}$$

From (2.3) and (2.4) one concludes that for some complex constants $\tau_{k,n-1}$, $\tau_{k,n-1}^c$ ($k = 0, \dots, n$; $n = 1, \dots, \nu - 1$)

$$x_n \tau_{n,n-1} = A x_{n-1} - x_{n-1}\tau_{n-1,n-1} - \cdots - x_0 \tau_{0,n-1}, \tag{2.6a}$$

$$y_n \tau_{n,n-1}^c = A^H y_{n-1} - y_{n-1}\tau_{n-1,n-1}^c - \cdots - y_0 \tau_{0,n-1}^c, \tag{2.6b}$$

where $\tau_{n,n-1} \neq 0$ and $\tau_{n,n-1}^c \neq 0$ can be chosen arbitrarily for normalizing x_n and y_n. The choice will of course effect the constant δ_n and the further coefficients $\tau_{n,m}$ and $\tau_{n,m}^c$ ($m \geq n$).

For $n = \nu$ (2.6) can still be satisfied for any chosen nonzero values of $\tau_{\nu,\nu-1}$ and $\tau_{\nu,\nu-1}^c$ by some $x_\nu \perp_B \mathcal{L}_{\nu-1}$ and some $y_\nu \perp_B \mathcal{K}_{\nu-1}$, but these two vectors may be orthogonal to each other so that (2.2) does not hold. In particular, one or even both may be the zero vector o. Hence, (2.4)–(2.6) can be assumed to hold for $n = \nu$ also. To comply with our future notation we choose $\tau_{\nu,\nu-1}^c = \overline{\tau_{\nu,\nu-1}}$ and set $\gamma_{\nu-1} := \tau_{\nu,\nu-1}$.

Let us introduce the $N \times \nu$ matrices X and Y with columns $x_0, x_1, \dots, x_{\nu-1}$ and $y_0, y_1, \dots, y_{\nu-1}$, respectively, the $\nu \times \nu$ Hessenberg matrices $T = \{\tau_{m,n}\}$ and $T^c = \{\tau_{m,n}^c\}$, and the $\nu \times \nu$ diagonal matrix $D_\delta := \text{diag}\,[\delta_0, \delta_1, \dots, \delta_{\nu-1}]$. Then (2.2) and (2.6) become

$$Y^H B X = D_\delta \tag{2.7}$$

and

$$AX = XT + x_\nu \gamma_{\nu-1} e_\nu^T, \quad A^H Y = YT^c + y_\nu \overline{\gamma_{\nu-1}} e_\nu^T, \tag{2.8}$$

where e_ν^T is the last row of the $\nu \times \nu$ identity matrix. Using (2.7) and $x_\nu \perp_B \mathcal{L}_{\nu-1}$, $y_\nu \perp_B \mathcal{K}_{\nu-1}$ we conclude further that

$$Y^H B A X = D_\delta T, \quad X^H B^H A^H Y = \overline{D_\delta} T^c, \tag{2.9}$$

and hence

$$D_\delta T = (T^c)^H D_\delta. \tag{2.10}$$

Here, on the left-hand side we have a lower Hessenberg matrix, while on the right-hand side there is an upper Hessenberg matrix. Consequently, both T and T^c must be tridiagonal,

$$T := \begin{bmatrix} \alpha_0 & \beta_1 & & & \\ \gamma_0 & \alpha_1 & \beta_2 & & \\ & \gamma_1 & \alpha_2 & \ddots & \\ & & \ddots & \ddots & \beta_{\nu-1} \\ & & & \gamma_{\nu-2} & \alpha_{\nu-1} \end{bmatrix}, \quad T^c := \begin{bmatrix} \alpha_0^c & \beta_1^c & & & \\ \gamma_0^c & \alpha_1^c & \beta_2^c & & \\ & \gamma_1^c & \alpha_2^c & \ddots & \\ & & \ddots & \ddots & \beta_{\nu-1}^c \\ & & & \gamma_{\nu-2}^c & \alpha_{\nu-1}^c \end{bmatrix}. \tag{2.11}$$

From (2.10) one obtains

$$\alpha_n^c = \overline{\alpha_n} \tag{2.12}$$

and

$$\delta_{n-1}\beta_n = \delta_n \overline{\gamma_{n-1}^c}\,, \quad \delta_{n-1}\overline{\beta_n^c} = \delta_n \gamma_{n-1}\,. \tag{2.13}$$

Hence,

$$\beta_n \gamma_{n-1} = \overline{\beta_n^c \gamma_{n-1}^c}\,, \tag{2.14}$$

which can be satisfied by setting

$$\beta_n^c := \overline{\beta_n}\,, \ \ \gamma_{n-1}^c := \overline{\gamma_{n-1}}, \quad \text{i.e.,} \quad T^c := \overline{T}. \tag{2.15}$$

One could further set $\gamma_n := 1\,(\forall n)$ as in [33, 38, 39, 52], but we want to keep the freedom to scale x_n and y_n by a common factor since the standard BCG algorithm, which will be discussed in Sect. 3, makes use of it. Note that (2.15) implies in view of (2.13) that

$$\delta_n = \delta_{n-1}\beta_n/\gamma_{n-1}\,. \tag{2.16}$$

Another frequent choice [8, 20, 37, 51, 58] is

$$\beta_n^c := \overline{\gamma_{n-1}}\,, \ \ \gamma_{n-1}^c := \overline{\beta_n}, \quad \text{i.e.,} \quad T^c := T^H\,, \tag{2.17}$$

which according to (2.13) yields $\delta_n = 1$. Clearly the choice of δ_n only affects the scaling (including, possibly the sign, or rather the argument of all components) of x_n and y_n. Moreover, the choice between (2.15) and (2.17) affects only the relative scaling of the vectors x_n and y_n, and this is likewise more generally true for the freedom left in satisfying (2.13). In fact the free choice of δ_n and the freedom in satisfying (2.13) leads only to diagonal similarity transformations of the matrices T and T^c, cf. (2.22) and (2.24) below. It has been pointed out by Saad, cf. his Algorithm 3 in [58], that there is even more freedom in choosing y_n without affecting the sequence $\{x_n\}$. Here we work with (2.15) since the resulting parallelism in the recurrence formulas simplifies our treatment. It is also the appropriate choice for the generalization to the nongeneric case [25].

After shifting the index n by 1, (2.8) can be written as

$$Ax_n = x_{n+1}\gamma_n + x_n\alpha_n + x_{n-1}\beta_n, \tag{2.18a}$$

$$A^H y_n = y_{n+1}\overline{\gamma_n} + y_n\overline{\alpha_n} + y_{n-1}\overline{\beta_n}, \tag{2.18b}$$

$n = 0, \ldots, \nu - 1$, with $\beta_0 := 0$, $x_{-1} := y_{-1} := o$. Taking inner products of the first formula with y_{n-1}, y_n, and y_{n+1} one gets three relations involving δ_{n+1}, δ_n, δ_{n-1}, and the recurrence coefficients α_n, β_n, γ_n. From (2.16) one obtains two additional equations, thus leaving one degree of freedom that can be used to fix either δ_{n+1} or γ_n. Actually we just exploit here the fact that the relations (2.8) and (2.9) can be used to determine X, Y, T, and D_δ column by column. Altogether one gets:

Algorithm 1 (Biorthogonalization (BO) algorithm). *Choose x_0, $y_0 \in \mathbf{C}^N$ such that $\delta_0 := \, < y_0, x_0 >_B \neq 0$, and set $\beta_0 := 0$. For $n = 0, 1, \ldots$ compute*

$$\alpha_n := \, < y_n, Ax_n >_B /\delta_n\,, \tag{2.19a}$$

$$\beta_n := < y_{n-1}, Ax_n >_B / \delta_{n-1} = \gamma_{n-1}\delta_n/\delta_{n-1} \quad (\textit{if}\, n > 0), \tag{2.19b}$$

$$\tilde{x}_{n+1} := Ax_n - x_n\alpha_n - x_{n-1}\beta_n\,, \tag{2.19c}$$

$$\tilde{y}_{n+1} := A^H y_n - y_n\overline{\alpha_n} - y_{n-1}\overline{\beta_n}\,, \tag{2.19d}$$

$$\tilde{\delta}_{n+1} := < \tilde{y}_{n+1}, \tilde{x}_{n+1} >_B\,; \tag{2.19e}$$

if $\tilde{\delta}_{n+1} = 0$, *set for some* $\gamma_n \neq 0$

$$\nu := n+1, \quad x_\nu := \tilde{x}_\nu/\gamma_n\,, \quad y_\nu := \tilde{y}_\nu/\overline{\gamma_n}\,, \quad \delta_{n+1} := 0, \tag{2.19f}$$

and stop; otherwise, choose γ_n *and* δ_{n+1} *such that*

$$\gamma_n^2\delta_{n+1} = \tilde{\delta}_{n+1}\,, \tag{2.19g}$$

set

$$x_{n+1} := \tilde{x}_{n+1}/\gamma_n\,, \quad y_{n+1} := \tilde{y}_{n+1}/\overline{\gamma_n}\,, \tag{2.19h}$$

and proceed with the next step.

In view of the fact that the biorthogonality (2.2) was aimed at in our derivation, it is not surprising that conversely the formulas (2.19) of the BO algorithm imply that (2.2) and hence (2.5) hold:

THEOREM 2.1. *If (2.1) and (2.2) hold for* $m, n = 0, \ldots, \nu - 1$, *and the choice (2.15) for satisfying (2.14) is made, then the set of recurrence formulas (2.19) of the BO algorithm is valid for* $n = 0, \ldots, \nu - 2$, *but the algorithm may break down for* $n = \nu - 1$ *due to* $\tilde{\delta}_\nu = 0$. *(2.1) and (2.2) determine* $\{x_n\}_{n=0}^{\nu-1}$ *and* $\{y_n\}_{n=0}^{\nu-1}$ *uniquely up to scaling. If (2.1) and (2.2) hold also when* $m = \nu$ *or* $n = \nu$, *but with* $\delta_\nu = 0$, *then* x_ν *and* y_ν *are also determined uniquely up to scaling.*

Conversely, the BO algorithm produces two sequences $\{x_n\}_{n=0}^{\nu-1}$ *and* $\{y_n\}_{n=0}^{\nu-1}$ *satisfying (2.1) and (2.2), and, hence, (2.3)–(2.5). Moreover, (2.2) holds also for* $m = \nu$ *or* $n = \nu$ *except that* $\delta_\nu = 0$.

Proof. It remains to prove the second part: Assume that the sequences $\{x_n\}_{n=0}^{\nu-1}$ and $\{y_m\}_{m=0}^{\nu-1}$ have been generated by the BO algorithm and that (2.1) and (2.2) hold for $m, n = 0, \ldots, k$ $(< \nu)$. (For $k = 0$ this is clearly the case.) Then the validity of (2.1) for $m, n = k+1$ is obvious from (2.19c) and (2.19d). Moreover, by (2.19h), (2.19c) and (2.19d),

$$\begin{aligned} < y_m, x_{k+1} >_B &= y_m^H B(Ax_k - x_k\alpha_k - x_{k-1}\beta_k)/\gamma_k && \text{(2.20a)} \\ &= [\gamma_m y_{m+1}^H Bx_k + \alpha_k y_m^H Bx_k + \beta_m y_{m-1}^H Bx_k \\ &\quad - y_m^H Bx_k\alpha_k - y_m^H Bx_{k-1}\beta_k]/\gamma_k\,. && \text{(2.20b)} \end{aligned}$$

If $m \leq k-2$, all terms on the right-hand side of (2.20b) vanish. For $m = k-1$ we get by (2.19b) and (2.2) on the right-hand side of (2.20a) $(\beta_k\delta_{k-1} - 0 - \beta_k\delta_{k-1})/\gamma_k = 0$. Next, using (2.19a) and (2.2) one obtains for $m = k$ in (2.20a) $(\alpha_k\delta_k - \alpha_k\delta_k)/\gamma_k = 0$. Finally, by symmetry, $< y_{k+1}, x_n >_B = 0$ for $n = 0, \ldots, k$ also, and by (2.19e)–(2.19h) $< y_{k+1}, x_{k+1} >_B = \delta_k$. This completes the induction. □

In Algorithm 1 the parameters α_n, β_n, γ_n, and δ_{n+1} are defined uniquely except for the freedom in (2.19g) for choosing γ_n or δ_{n+1}. The classical choices for theoretical work are $\gamma_n := 1$

[38, 39, 52] or $\delta_{n+1} := 1$. The latter makes the two vector sequences orthonormal and yields $\beta_n = \gamma_{n-1}$, i.e., $T = T^T$, cf. (2.16). (Consequently, (2.15) and (2.17) coincide then.) For numerical computations the former choice is risky with respect to overflow [52], and the latter is inappropriate for real nonsymmetric matrices since it may lead to a complex γ_n. Therefore, in this situation

$$\gamma_n := \sqrt{\left|\tilde{\delta}_{n+1}\right|}\,, \quad \delta_{n+1} := \tilde{\delta}_{n+1}/\gamma_n^2 \tag{2.21a}$$

or

$$\gamma_n := \|\tilde{y}_{n+1}\|\,, \quad \delta_{n+1} := \tilde{\delta}_{n+1}/\gamma_n^2 \tag{2.21b}$$

are to be prefered.

If A and B are Hermitian and B is positive definite, the start with $y_0 = x_0$ and the natural choice $\delta_n > 0$ leads to $\overline{T} = T$ (i.e., T is real) and $y_n = x_n$ $(\forall n)$; moreover, one can choose $\gamma_n > 0$ $(\forall n)$, and then $\beta_n > 0$ $(\forall n)$ also; finally, if one lets $\delta_n = 1$ $(\forall n)$, then T is real symmetric. Likewise, if A is complex symmetric, one can attain $y_n = \overline{x}_n$ $(\forall n)$.

If γ_n is chosen independently of $\tilde{x}_{n+1}$ and $\tilde{y}_{n+1}$, the formulas (2.19) can be simplified, cf. Algorithm 2 below.

A most important point for the application of the Lanczos process (in all its variants discussed in this paper) is that the given matrix A need not be stored explicitly. Only procedures for computing Az and $A^H z$ for an arbitrary z must be available, and these are the only matrix-vector-products that come up. All the other operations are linear operations with vectors and inner products. Therefore the costs of these algorithms are dominated by the two mentioned matrix-vector-products, and there are always two such products required in each step. Hence, these costs are essentially the same for every algorithm based on the Lanczos process; there are only some minor differences with respect to the other operations and with respect to memory space.

The breakdowns of the BO algorithm have been discussed by many authors [33, 38, 39, 51, 52, 58]. They are due to either

(i) $x_\nu = o$ or $y_\nu = o$,

or

(ii) $x_\nu \neq o$ and $y_\nu \neq o$, but $< y_\nu, x_\nu >_B = 0$.

In view of (2.8) case (i) means that $\mathcal{K}_{\nu-1}$ or $\mathcal{L}_{\nu-1}$ is an invariant subspace of dimension ν of A or A^H, respectively. Since the dimension of any Krylov subspace of A of A^H is at most equal to the degree of the minimum polynomial of A [33], ν is in general smaller than the order N of the matrix, and one may have to restart the algorithm with a pair (x_ν, y_ν) that is biorthogonal to the pair of vector sequences constructed so far, cf. [38, 39, 52]. Conversely, if not before, the process breaks always down with $x_\nu = y_\nu = o$ when $n+1 = \nu$ reaches the degree of the minimal polynomial of A. If case (i) occurs before, we call it a *premature termination* (or breakdown).

Case (ii) is called a *serious breakdown*, and it is this breakdown (and the corresponding near-breakdown) that is addressed in a related paper [25]. It can be avoided if A and B are Hermitian and B is positive definite, since choosing $y_0 = x_0$ and $\delta_n > 0$ $(\forall n)$ yields then $y_n = x_n$ $(\forall n)$. On the other hand, the analysis in Sect. 4 will show that the breakdown can still occur with a real symmetric matrix if $y_0 \neq x_0$.

For the Euclidean norm (where $B = I$) it has been shown by Rutishauser [52] (for another proof see Householder [33]) that there exist x_0 and y_0 such neither a serious nor a premature breakdown occurs, i.e., such that the process does not break down before the degree of the minimal polynomial is attained. Unfortunately, such a pair (x_0, y_0) is in general not known. Moreover, any choice leading to a near-breakdown may in practice lead to stability problems. Rutishauser points also out that there always exists a nonsingular matrix S such that $A^T = SAS^{-1}$ and that choosing $y_0 = \overline{Sx_0}$ yields $y_n = \overline{Sx_n}$ $(n = 0, 1, \ldots, \nu - 1)$, so that in case (i) both vectors must be equal to the zero vector simultaneously. (Actually, Rutishauser's remark is restricted to real matrices, but generalizes in the above indicated way to the complex case. Incidentally, the transformation S is also crucial in a paper of Young and Jea [34, Def. 1.1, Thm. 4.1].) However, S is again in general not known. An exception is the case where $A = A^T$ is symmetric, and thus $S = I$.

If both $x_\nu = o$ and $y_\nu = o$, one can set $\gamma_{\nu-1} := \beta_\nu := 0$, so that after the restart the relations (2.8) hold even beyond this ν. If only this type of breakdown occurs, then after possibly several restarts the algorithm must terminate with $x_N = y_N = o$ and $\nu := N$, and it follows that

$$AX = XT, \quad A^H Y = Y\overline{T}, \tag{2.22}$$

and

$$Y^H BX = D_\delta = \text{diag}\,(\delta_0, \delta_1, \ldots, \delta_{N-1}), \tag{2.23}$$

with all the matrices being square of order N. The tridiagonal matrix T may have some elements $\gamma_j = \beta_{j+1} = 0$ due to the restarts. X and Y are nonsingular, hence, T is similar to A, and $\overline{T}$ is similar to A^H. The above mentioned result of Rutishauser therefore implies that to every matrix A there exists a similar tridiagonal matrix T with $\gamma_j = 0 \Longleftrightarrow \beta_{j+1} = 0$. In particular, if A has distinct eigenvalues, there exists one with $\gamma_j \neq 0$ $(j = 0, \ldots, N-2)$.

The formula $AX = XT$ points to the first main application of the BO process: The similarity reduction of a given matrix A to a tridiagonal matrix T. For the latter the computation of the eigenvalues is much less costly. Unfortunatly, in practice, inherent roundoff problems jeopardize the use of the BO process and related algorithms. Recall that in the nth step of the BO algorithm the constants α_n, β_n, γ_n, δ_{n+1} and the vectors x_{n+1} and y_{n+1} are determined so that $x_{n+1} \perp_B y_n$, $x_{n+1} \perp_B y_{n-1}$, and $< y_{n+1}, x_{n+1} >_B = \delta_{n+1}$. Nevertheless, biorthogonality holds theoretically for all m and n. However, due to roundoff, one often encounters a loss of this inherited biorthogonality when n becomes large. Recall that orthogonal projection necessarily reduces the size of a vector and thus its relative accuracy. If recursively generated projections of vectors are the relevant data used in the process of building up a similarity transformation, it is thus not surprising that working with finite precision may have a devastating effect. Lanczos [39, pp. 39-40] was already aware of this problem and suggested on the one hand reorthogonalization as a possible yet expensive remedy and, on the other hand, for the iterative solution of linear systems (see below), the modification of the right hand side by damping the components corresponding to the large eigenvalues. This second mean, which he called "purification" of the right hand side, is similar in spirit to what we now call polynomial preconditioning, but he applied it only before the Lanczos process and not at every step of it. By suitable preconditioning the convergence of the iterative linear equation solvers based on the Lanczos process is often improved considerably. The loss of biorthogonality is then also reduced and at the same time becomes less relevant since the error is damped away long before n is comparable to N in size. Incidentally, Lanczos [39, p. 40] wrote: "If by some means fast convergence can be enforced, the scheme might terminate in much fewer than N steps. Even if theoretically speaking the last vector vanishes exactly only after N iterations, it is quite possible that it may drop *practically* below negligible bounds after a relatively few iterations."

On the other hand, for the eigenvalue problem, where preconditioning is usually impossible, the loss of orthogonality is a problem even in the symmetric case, for which this numerical phenomenon was completely analyzed by Paige [42, 43, 44, 45]. (His results are also discussed in [9, 47].) It is coupled with the occurence of "spurious" copies of eigenvalues and eigenvectors. Parlett and his coworkers [47, 48, 49, 50, 51], Cullum and Willoughby [9], and Simon [63, 64] have explored ways to get around this problem in practice.

If the BO algorithm breaks down prematurely due to $x_\nu = o$ or $y_\nu = o$ one has at least found an invariant subspace of A or A^H. Then the set of eigenvalues of the $\nu \times \nu$ matrix T is a subset of the spectrum of A. In practice, the process is usually applied to very large matrices and stopped at some $\nu << N$ even when it does not break down. The ν eigenvalues of T are then considered as approximations of ν eigenvalues of A, and typically they tend to approximate those eigenvalues that are largest in modulus. Lanczos [38, p. 270] found also for this important phenomenon a heuristic explanation. Cybenko [10] gave explicit formulas for the characteristic polynomial of T in terms of the eigenvalues of A, and these formulas can also be used to explain heuristically why the approximations are often so good; however, these results seem not to be readily applicable in practice.

If the choice (2.15) is replaced by (2.17), one obtains instead of (2.22)

$$AX = XT, \quad A^H Y = YT^H, \tag{2.24}$$

as, e.g., in Parlett et al. [51]. Now, if T has the property that $\gamma_j = 0 \Longleftrightarrow \beta_{j+1} = 0$ ($j = 0, 1, \ldots, N-2$), then there is a diagonal matrix D_T such that $T^H = D_T \overline{T} D_T^{-1}$ and thus $A^H(YD_T) = (YD_T)\overline{T}$. Since the choice (2.15) together with a rule for choosing γ_n or δ_{n+1} leads to uniquely determined sequences $\{x_n\}$ and $\{y_n\}$ characterized by (2.8) and (2.9), it follows from (2.24) that when (2.15) is replaced by (2.17), but the same scale factors γ_n are kept, one obtains exactly the same sequence $\{x_n\}$ as before and a sequence $\{y_n\}$ differing from the previous one only by scalar factors.

Let us now turn to the application of the Lanczos BO algorithm to the problem of solving linear systems of equations $Az = b$. Of course, we assume now that A is nonsingular. One way to get to this application is by an analogy with the standard Omin or the Ores version of the conjugate gradient (CG) method for positive definite symmetric systems. If we denote by z^* the solution of $Az = b$, by z_0 the initial approximation of this solution, by $x_0 := b - Az_0$ the corresponding residual, and, as in (2.1), by $\mathcal{K}_n$ the nth Krylov subspace generated from x_0, then the CG method is characterized by the property that the nth iterate z_n minimizes the quadratic function $z \mapsto< (z^* - z), A(z^* - z) >$ among all $z \in z_0 + \mathcal{K}_{n-1}$. This optimal solution has then the property that the residuals $x_n = b - Az_n$ form an orthogonal sequence. Hence, necessarily $x_\nu = o$ for some $\nu < N$, and thus $z_\nu = z^*$.

In the BO method one constructs a pair of biorthogonal sequences $\{x_n\}$, $\{y_n\}$, and unless one encounters a serious breakdown, one finds $x_\nu = o$ or $y_\nu = o$ for some ν. Although one must expect that the possibility of $x_\nu \neq o, y_\nu = o$ will cause a problem, one may try to define iterates z_n in such a way that $x_n = b - Az_n$ is exactly the corresponding residual. Hence, the orthogonality condition $x_n = b - Az_n \perp_B \mathcal{K} n - 1$ of the CG method is replaced by the biorthogonality condition $x_n = b - Az_n \perp_B \mathcal{L} n - 1$ here. Since $x_n \in \mathcal{K}_n$, the resulting algorithm is necessarily a polynomial acceleration method (or semiiterative method [69] or gradient method [56]). For the matrix X of residual vectors generated by such a method holds the relation $AX = XH$ with an unreducible upper Hessenberg matrix H, which must satisfy the consistency condition that in each column the elements sum up to zero. In our case, where $H = T$, it follows that one must choose $\gamma_n := -\alpha_n - \beta_n$. (Recall that γ_n or δ_n can be chosen in the BO algorithm.) However, this requirement may lead to yet another type of breakdown, which we suggest to call the *normalization breakdown*: It happens when $\alpha_n + \beta_n = 0$.

Recalling that the formulas for the *BO algorithm* can be simplified if γ_n is independent of $\tilde{\delta}_{n+1}$ and guessing the right formula for the approximants z_n of the solution leads finally to

ALGORITHM 2 (NORMALIZED BO ALGORITHM FOR LINEAR EQUATIONS: BIORES). *For solving $Az = b$ choose an initial approximation z_0, set $x_0 := b - Az_0$, and choose y_0 such that $< y_0, x_0 >_B \neq 0$. Then apply Algorithm 1 with*

$$\gamma_n := -\alpha_n - \beta_n \,, \tag{2.25}$$

so that (2.19c)–(2.19h) simplify to

$$x_{n+1} := (Ax_n - x_n\alpha_n - x_{n-1}\beta_n)/\gamma_n \,, \tag{2.26a}$$

$$y_{n+1} := (A^H y_n - y_n\overline{\alpha_n} - y_{n-1}\overline{\beta_n})/\overline{\gamma_n} \,, \tag{2.26b}$$

$$\delta_{n+1} := < y_{n+1}, x_{n+1} >_B \;. \tag{2.26c}$$

Additionally compute the vectors

$$z_{n+1} := -(x_n + z_n\alpha_n + z_{n-1}\beta_n)/\gamma_n \,. \tag{2.26d}$$

If $x_{n+1} = o$ the process terminates and z_{n+1} is the solution; if $x_{n+1} \neq o$ but $\delta_{n+1} = 0$ the algorithm breaks down. In both cases we set $\dot{\nu} := n + 1$. If $\gamma_n = 0$, it breaks also down, and we set $\dot{\nu} := n$.

LEMMA 2.2. *In Algorithm 2 the vector x_n is the nth residual: $b - Az_n = x_n$ $(n = 0, 1, \ldots, \dot{\nu})$.*

Proof. Assuming $b - Az_n = x_n$ and $b - Az_{n-1} = x_{n-1}$ one gets, using (2.26), (2.19c), (2.19h), and (2.25)

$$\begin{aligned} b - Az_{n+1} &= b + (Ax_n + Az_n\alpha_n + Az_{n-1}\beta_n)/\gamma_n \\ &= b + (Ax_n - x_n\alpha_n - x_{n-1}\beta_n + b(\alpha_n + \beta_n))/\gamma_n = x_{n+1} \,, \end{aligned}$$

which is what we need for the induction. □

The matrix relations (2.7)–(2.9) (with $T^c = \overline{T}$) can be considered as a shorthand notation for the BO algorithm: (2.8) describes the recurrences for x_n and y_n, and (2.7) plus the first equation in (2.9) summarize the formulas for the determination of the elements α_n, β_n, γ_n, and δ_n of T and D_δ. Likewise, with $Z := [z_0, \ldots, z_{\dot{\nu}-1}]$ the recurrence (2.26d) for z_n can be written as

$$X = -ZT - z_{\dot{\nu}}\gamma_{\dot{\nu}-1}e_{\dot{\nu}}^T \,. \tag{2.27}$$

(Here, and on other occasions, *we reduce the size ν of matrices like T, X, Y, etc., from ν to $\dot{\nu}$ without explicit mentioning it.*) The column sum condition (2.25) can be expressed as

$$[1, 1, \ldots, 1]\, T = \gamma_{\dot{\nu}-1}e_{\dot{\nu}}^T \,. \tag{2.28}$$

(Since $\gamma_{\dot{\nu}-1}$ is not an element of the $\dot{\nu} \times \dot{\nu}$ matrix T, it does not matter if (2.25) is not satisfied for $n = \dot{\nu} - 1$.)

Algorithm 2 is mathematically equivalent and in the spirit identical to Lanczos/ORTHORES defined by Jea and Young [34] and implemented in [1] and the NSPCG package [40]. However, the formulas given by these authors differ considerably from ours. They write (2.26a) as

$$x_{n+1} = (x_n - Ax_n\gamma_{n+1}^{JY})\rho_{n+1}^{JY} + x_{n-1}(1-\rho_{n+1}^{JY}) \,, \tag{2.29}$$

so that by comparison with (2.26a)

$$\rho_{n+1}^{JY}\gamma_{n+1}^{JY} = -1/\gamma_n \,, \quad \rho_{n+1}^{JY} = -\alpha_n/\gamma_n \,,$$

from which we conclude that in view of (2.19a)

$$\gamma_{n+1}^{JY} = 1/\alpha_n = \delta_n / < y_n, Ax_n >_B \,, \tag{2.30}$$

and in view of (2.25) and (2.19b)

$$\frac{1}{\rho_{n+1}^{JY}} = 1 + \frac{\beta_n}{\alpha_n} = 1 + \frac{\gamma_{n-1}\delta_n}{\delta_{n-1}}\gamma_{n+1}^{JY} = 1 - \frac{\gamma_{n+1}^{JY}\delta_n}{\rho_n^{JY}\gamma_n^{JY}\delta_{n-1}} \,. \tag{2.31}$$

The last formula is a recurrence for $\{\rho_n^{JY}\}$, which replaces our formula (2.19b) for $\{\beta_n\}$.

We claim that by a small modification it is possible to avoid the normalization breakdown that may occur in Algorithm 2:

ALGORITHM 3 (UNNORMALIZED BIORES ALGORITHM). *Modify Algorithm 2 whenever* $\gamma_n := -\alpha_n - \beta_n$ *has a small absolute value by choosing instead some arbitrary* $\gamma_n \neq 0$.

One can conclude immediately:

LEMMA 2.3. *The breakdown indices* ν *of the BO algorithm and the unnormalized BIORES algorithm are identical, but the breakdown index* $\dot\nu$ *of the normalized BIORES algorithm can, but need not be smaller.*

Moreover, in view of the following result, Algorithm 3 delivers the solution of $Az = b$ whenever it does not break down.

LEMMA 2.4. *Let*

$$\dot\rho_0 := 1, \quad \dot\rho_{n+1} := -(\alpha_n\dot\rho_n + \beta_n\dot\rho_{n-1})/\gamma_n \,, \quad n = 0, 1, \dots, \nu. \tag{2.32}$$

Then x_n *and* z_n *are in Algorithm 3 related by*

$$x_n = b\dot\rho_n - Az_n \,. \tag{2.33}$$

Proof. For $n = 0$, (2.33) is correct. Assume it is correct up to the index n. Then by (2.26) and (2.32)

$$\begin{aligned} (b\dot\rho_{n+1} - Az_{n+1})\gamma_n &= -b(\alpha_n\dot\rho_n + \beta_n\dot\rho_{n-1}) + Ax_n + Az_n\alpha_n + Az_{n-1}\beta_n \\ &= Ax_n - x_n\alpha_n - x_{n-1}\beta_n = x_{n+1}\gamma_n \,. \end{aligned}$$

□

As a matter of fact the choice of γ_n only effects the scaling of x_n and z_n, but not the subspace each of them spans. From (2.33) it is clear that whenever $n < \dot\nu$ and $\dot\rho_n \neq 0$ one can rescale x_n and z_n to get the corresponding vectors of Algorithm 2. But once $\dot\rho_n = 0$, this is impossible and Algorithm 2 breaks down. In contrast, here one can still go on, and if $x_n = o$ and $\dot\rho_n \neq 0$ for some $n > \dot\nu$ one finds a solution that is not accessible through Algorithm 2 (using the same initial data).

Of course, in practice, where vanishing of $\dot\rho_n$ is unlikely, but near-vanishing matters, Algorithm 3 must be considered as a stabilized version of Algorithm 2 reducing the chance of overflow.

3. Biconjugation algorithms.

In his second paper on the subject Lanczos [39] suggested under the section heading "The complete algorithm for minimized iterations" an alternative algorithm for computing the sequences $\{x_n\}$, $\{y_n\}$ generated by the BO algorithm. He also discussed in detail how to apply this algorithm for solving linear systems of equations. In this connection a normalized version of it was later called the *biconjugate gradient (BCG) algorithm* by Fletscher [15] and *Lanczos/ORTHOMIN* by Jea and Young [34]. For the general, unnormalized procedure we use here the abbreviation *BOBC algorithm* since it produces both a biorthogonal pair and a biconjugate pair of vector sequences. It is related to the original Hestenes-Stiefel (OMIN) version of the conjugate gradient method [31] in the same way as the Lanczos BO algorithm is related to the ORES version of the conjugate gradient algorithm. For the BCG algorithm we use often the name *BIOMIN* in order to stress the analogy to the general OMIN method and the parallelism to BIORES and BIODIR. The latter is discussed later in this section.

ALGORITHM 4 (BOBC ALGORITHM). *Choose* $x_0, y_0 \in \mathbf{C}^N$ *such that* $\delta_0 := < y_0, x_0 >_B \neq 0$*, and set* $u_0 := x_0$*,* $v_0 := y_0$*. For* $n = 0, 1, \ldots$ *choose* $\gamma_n \neq 0$*, evaluate*

$$\varphi_n := < v_n, Au_n >_B / \delta_n \, , \tag{3.1a}$$

$$x_{n+1} := (Au_n - x_n\varphi_n)/\gamma_n \, , \tag{3.1b}$$

$$y_{n+1} := (A^H v_n - y_n\overline{\varphi_n})/\overline{\gamma_n} \, , \tag{3.1c}$$

$$\delta_{n+1} := < y_{n+1}, x_{n+1} >_B \, ; \tag{3.1d}$$

if $\varphi_n = 0$*, set* $\dot{\nu} := n$ *and stop; otherwise compute*

$$\psi_{n+1} := \delta_{n+1}\gamma_n/(\delta_n\varphi_n) \, , \tag{3.1e}$$

$$u_{n+1} := x_{n+1} - u_n\psi_{n+1} \, , \tag{3.1f}$$

$$v_{n+1} := y_{n+1} - v_n\overline{\psi_{n+1}} \, ; \tag{3.1g}$$

if $\delta_{n+1} = 0$*, set* $\dot{\nu} := n+1$ *and stop; otherwise proceed with the next step.*

The basic result concerning this algorithm is:

THEOREM 3.1. *The sequences* $\{x_n\}_{n=0}^{\dot{\nu}}$ *and* $\{y_n\}_{n=0}^{\dot{\nu}}$ *generated by the BOBC algorithm are biorthogonal, and the sequences* $\{u_n\}_{n=0}^{\dot{\nu}}$ *and* $\{v_n\}_{n=0}^{\dot{\nu}}$ *are biconjugate (with respect to A) except that* $< y_{\dot{\nu}}, x_{\dot{\nu}} >_B = 0$ *or* $< v_{\dot{\nu}}, Au_{\dot{\nu}} >_B = 0$*. For* $m, n = 0, 1, \ldots, \dot{\nu}$*,*

$$< y_m, x_n >_B = \begin{cases} 0\,, & m \neq n, \\ \delta_n\,, & m = n, \end{cases} \tag{3.2}$$

$$< v_m, Au_n >_B = \begin{cases} 0\,, & m \neq n, \\ \delta_n\varphi_n\,, & m = n, \end{cases} \tag{3.3}$$

where $\delta_n\varphi_n \neq 0$ *for* $0 \leq n \leq \dot{\nu} - 1$*. Moreover, for* $n = 1, \ldots, \dot{\nu} - 1$ *holds in addition to (2.3)*

$$u_n \in \mathcal{K}_n \backslash \mathcal{K}_{n-1} \, , \quad v_n \in \mathcal{L}_n \backslash \mathcal{L}_{n-1} \, . \tag{3.4}$$

Proof. (An adaptation of Fletcher's proof [15] to the complex case and to our adjustable normalization.) For $m = n = 0$ (3.2)–(3.4) and (2.3) clearly hold. Assume that they hold for $m, n = 0, \ldots, k$ $(< \dot{\nu})$. From (3.1b) and (3.1g) one gets

$$\begin{aligned} < y_m, x_{k+1} >_B &= y_m^H B(Au_k - x_k \varphi_k)/\gamma_k \\ &= [v_m^H BAu_k + \psi_m v_{m-1}^H BAu_k - y_m^H Bx_k \varphi_k]/\gamma_k \,. \end{aligned} \tag{3.5}$$

Here for $m \leq k - 1$ all terms are zero by assumption. On the other hand, if $m = k$, (3.1a) and (3.1d) inserted into (3.5) yield $< y_k, x_{k+1} >_B = (\varphi_k \delta_k + 0 - \varphi_k \delta_k)/\gamma_k = 0$, so that $< y_m, x_{k+1} >_B = 0$ for $m \leq k$. By symmetry, $< y_{k+1}, x_m >_B = 0$ for $m \leq k$ also and together with (3.1d) it follows that (3.2) holds up to $k + 1$.

Similarly, using (3.1f) and (3.1c) we get

$$\begin{aligned} < v_m, Au_{k+1} >_B &= v_m^H BA(x_{k+1} - u_k \psi_{k+1}) \\ &= \gamma_m y_{m+1}^H Bx_{k+1} + \varphi_m y_m^H Bx_{k+1} - v_m^H BAu_k \psi_{k+1} \,. \end{aligned} \tag{3.6}$$

Here too, for $m \leq k - 1$ all terms are known to be zero. If $m = k$, one obtains from (3.2), (3.3) and (3.1e) $< v_k, Au_{k+1} >_B = \gamma_k \delta_{k+1} + 0 - \delta_k \varphi_k \psi_{k+1} = 0$. Hence, $< v_m, Au_{k+1} >_B = 0$ for $m \leq k$, and by symmetry $< v_{k+1}, Au_m >_B = 0$ also. Finally, the equation in (3.3) for $m = n = k + 1$ results from (3.1a).

From the formulas (3.1b), (3.1c), (3.1f), (3.1g), (2.3), and (3.4) it follows clearly that $x_{k+1}, u_{k+1} \in \mathcal{K}_{k+1}$ and $y_{k+1}, v_{k+1} \in \mathcal{L}_{k+1}$. As in Sect. 2 $\delta_n \neq 0$ implies that $x_{k+1} \not\in \mathcal{K}_k$ and $y_{k+1} \not\in \mathcal{L}_{k+1}$. By (3.1f) and (3.1g) it thus follows that (3.4) holds for $n = k+1$. This completes the induction. □

The BOBC algorithm has also a matrix interpretation, which quickly reveals the relation to the BO algorithm. Define the $N \times \dot{\nu}$ matrices

$$X := (x_0, x_1, \ldots, x_{\dot{\nu}-1}) \,, \quad Y := (y_0, y_1, \ldots, y_{\dot{\nu}-1}) \,, \tag{3.7a}$$

$$U := (u_0, u_1, \ldots, u_{\dot{\nu}-1}) \,, \quad V := (v_0, v_1, \ldots, v_{\dot{\nu}-1}) \,, \tag{3.7b}$$

and the $\dot{\nu} \times \dot{\nu}$ matrices

$$L := \begin{bmatrix} \varphi_0 & & & & \\ \gamma_0 & \varphi_1 & & & \\ & \gamma_1 & \varphi_2 & & \\ & & \ddots & \ddots & \\ & & & \gamma_{\dot{\nu}-2} & \varphi_{\dot{\nu}-1} \end{bmatrix} , \quad R := \begin{bmatrix} 1 & \psi_1 & & & \\ & 1 & \psi_2 & & \\ & & 1 & \ddots & \\ & & & \ddots & \psi_{\dot{\nu}-1} \\ & & & & 1 \end{bmatrix} , \tag{3.8}$$

which are lower and upper bidiagonal, respectively. Then, according to (3.1b), (3.1c), (3.1f), and (3.1g),

$$AU = XL + x_{\dot{\nu}} \gamma_{\dot{\nu}-1} e_{\dot{\nu}}^T \,, \quad A^H V = Y\overline{L} + y_{\dot{\nu}} \overline{\gamma_{\dot{\nu}-1}} e_{\dot{\nu}}^T \,, \tag{3.9}$$

and

$$X = UR \,, \quad Y = V\overline{R} \,. \tag{3.10}$$

After setting

$$T := LR\,, \quad T_1 := RL + e_{\dot\nu}\gamma_{\dot\nu-1}\psi_{\dot\nu}e_{\dot\nu}^T \tag{3.11}$$

we conclude by eliminating U and V or X and Y, respectively, that

$$AX = XT + x_{\dot\nu}\gamma_{\dot\nu-1}e_{\dot\nu}^T\,,\,, \quad A^HY = Y\overline{T} + y_{\dot\nu}\overline{\gamma_{\dot\nu-1}}e_{\dot\nu}^T \tag{3.12}$$

and

$$AU = UT_1 + u_{\dot\nu}\gamma_{\dot\nu-1}e_{\dot\nu}^T\,,\,, \quad A^HV = V\overline{T_1} + v_{\dot\nu}\overline{\gamma_{\dot\nu-1}}e_{\dot\nu}^T\,. \tag{3.13}$$

The second term in the formula (3.11) for T_1 modifies only the element in the lower right corner of RL by adding $\gamma_{\dot\nu-1}\psi_{\dot\nu}$ to it. This has to do with the fact that the factors of the UL decomposition of the nth principal submatrix of T_1 are *not* the nth principal submatrices of R and L; but since T_1 is tridiagonal, only its (n, n) element needs to be modified.

Since (3.12) is identical with (2.8) if $T^c = \overline{T}$ there, one obtains the following:

THEOREM 3.2. *If the same starting vectors x_0 and y_0 and the same parameters γ_n are chosen in the BO and the BOBC algorithms, then the same biorthogonal vector sequences $\{x_n\}$ and $\{y_n\}$ are produced, except that the BOBC algorithm may break down earlier due to $\varphi_n = 0$. The matrices L and R of the recurrence coefficients of the BOBC algorithm are obtained by LU decomposition of the matrix T containing the recurrence coefficients of the BO algorithm. The possible earlier breakdown of the BOBC algorithm is due the possible nonexistence of an LU decomposition (without pivoting) for the tridiagonal matrix T generated by the BO algorithm.*

The usual application of the BO algorithm is to solving a linear system of equations $Az = b$. Like in the application of the BO algorithm to this task one has to stick to a particular choice of γ_n in order to fulfill the consistency condition for polynomial acceleration methods, namely

$$\gamma_n := -\varphi_n\,. \tag{3.14}$$

Again, one can then define the iterates z_n in such a way that x_n is the nth residual.

ALGORITHM 5 (BOBC ALGORITHM FOR LINEAR EQUATIONS: BIOMIN OR BICONJUGATE GRADIENT (BCG) ALGORITHM). *For solving $Az = b$ choose an initial approximation z_0, set $z_0 := b - Ax_0$, and apply Algorithm 4 with $\gamma_n := -\varphi_n$, so that the nth step consists of:*

$$\varphi_n := < v_n, Au_n >_B /\delta_n\,; \tag{3.15a}$$

if $\varphi_n = 0$, set $\dot\nu := n$ and stop; otherwise compute

$$\omega_n := 1/\varphi_n\,, \tag{3.15b}$$

$$x_{n+1} := x_n - Au_n\omega_n\,, \tag{3.15c}$$

$$y_{n+1} := y_n - A^Hv_n\overline{\omega_n}\,, \tag{3.15d}$$

$$\delta_{n+1} := < y_{n+1}, x_{n+1} >_B\,, \tag{3.15e}$$

$$\psi_{n+1} := -\delta_{n+1}/\delta_n\,, \tag{3.15f}$$

$$u_{n+1} := x_{n+1} - u_n \psi_{n+1} \,, \tag{3.15g}$$

$$v_{n+1} := y_{n+1} - v_n \overline{\psi_{n+1}} \,, \tag{3.15h}$$

$$z_{n+1} := z_n + u_n \omega_n \,. \tag{3.15i}$$

If $x_{n+1} = 0$ the process terminates and z_{n+1} is the solution; if $\delta_{n+1} = 0$ (hence, $\psi_{n+1} = 0$), but $x_{n+1} \neq 0$, the algorithm breaks down. In both cases we set $\dot{\nu} := n + 1$. If $\varphi_n = 0$ it breaks also down, and we set $\dot{\nu} := n$.

Assuming $b - Az_n = x_n$ and using (3.15c) and (3.15i) one gets in fact

$$b - Az_{n+1} = b - Az_n - Au_n\omega_n = x_n - Au_n\omega_n = x_{n+1},$$

so that by induction $b - Az_n = x_n$ for $n = 0, 1, \ldots, \dot{\nu}$. Consequently, if $x_{\dot{\nu}} = 0$, then $z_{\dot{\nu}}$ is the solution of the system.

Note that by definition of $\dot{\nu}$

$$\delta_n \neq 0, \quad \varphi_n \neq 0, \quad \psi_n \neq 0, \quad \omega_n \neq 0, \quad n = (0), 1, \ldots, \dot{\nu} - 1, \tag{3.16a}$$

$$\delta_{\dot{\nu}} \neq 0 \Longrightarrow \varphi_{\dot{\nu}} = 0, \;\; \psi_{\dot{\nu}} \neq 0, \tag{3.16b}$$

$$\delta_{\dot{\nu}} = 0 \Longrightarrow \psi_{\dot{\nu}} = 0, \;\; \varphi_{\dot{\nu}} \;\text{ undefined}, \tag{3.16c}$$

If $\delta_{\dot{\nu}} = 0$ and $< v_{\dot{\nu}}, Av_{\dot{\nu}} >_B \neq 0$, one could set $\omega_{\dot{\nu}} = 0$, which would then yield $x_{\dot{\nu}+1} = x_{\dot{\nu}}$, $y_{\dot{\nu}+1} = y_{\dot{\nu}}$, $\delta_{\dot{\nu}+1} = 0$, $u_{\dot{\nu}+1} \in \mathcal{K}_{\dot{\nu}}$, $v_{\dot{\nu}+1} \in \mathcal{L}_{\dot{\nu}}$ (for any $\psi_{\dot{\nu}+1}$), i.e., the algorithm *stalls.* However, here we take the position that $\omega_{\dot{\nu}}$ is undefined if $\delta_{\dot{\nu}} = 0$.

In analogy to the situation in BIORES one may wonder whether it is possible to avoid one type of breakdown by using an unnormalized version of BIOMIN:

ALGORITHM 6 (UNNORMALIZED BIOMIN ALGORITHM). *Modify Algorithm 5 by choosing $\gamma_n \neq 0$ arbitrarily in Algorithm 4 (hence, by using formulas (3.1)) and by replacing the additional assignment (3.15i) by*

$$z_{n+1} := -z_n\varphi_n/\gamma_n - u_n/\gamma_n \,. \tag{3.17}$$

In this algorithm the vectors x_n and z_n are again related by (2.33), with the same scale factors $\dot{\rho}_n$, which, however, can now be generated according to

$$\dot{\rho}_{n+1} = -\dot{\rho}_n\varphi_n/\gamma_n. \tag{3.18}$$

(Induction step: $(b\dot{\rho}_{n+1} - Az_{n+1})\gamma_n = -b\dot{\rho}_n\varphi_n + Az_n\varphi_n + Au_n = -(x_n\varphi_n - Au_n) = x_{n+1}\gamma_n$.)

However, Algorithm 4, and hence also Algorithm 6 break in any case down when $\varphi_n = 0$. Therefore, Algorithm 6 does not bring any substancial advantage. Concerning the relation to Algorithm 2 holds the following:

LEMMA 3.3. *If the same starting vectors are used, then the three algorithms BIORES (Algorithm 2), BIOMIN (BCG, Algorithm 5), and unnormalized BIOMIN (Algorithm 6) are mathematically equivalent, i.e., produce the same iterates z_n, and break down at the same time. (Hence, $\dot{\nu}$ is the same for all three.)*

Proof. The relation between the BO and the BOBC algorithm was established in Theorem 3.2. Compared to the BO algorithm BIORES requires additionally that $\gamma_n := -\alpha_n - \beta_n \neq 0$. But since γ_n is the same for the BO and the BOBC algorithm, and $\gamma_n := -\varphi_n$ in BIOMIN, this condition is equivalent to $\varphi_n \neq 0$, which has to be observed in the BOBC algorithm anyway, and which, conversely, is the only additional condition there compared to the BO algorithm. □

In other words, if the BO and the BOBC algorithms are used as a method to produce the matrix decompositions (3.12) and (3.9) + (3.10), respectively, the ν in the BO algorithm may be larger than $\acute{\nu}$ in the BOBC algorithm, since the matrix T of the BO algorithm may not have an LU decomposition. However, if they are used for solving a linear system and normalized in order to fulfill the consistency condition for polynomial acceleration methods (i.e., if (2.25) and (3.14) are observed), then the implicitly imposed scaling of the tridiagonal matrix is such that the $\acute{\nu}$ of the BIORES algorithm is as small as the $\acute{\nu}$ of the BCG (BIOMIN) algorithm. With unnormalized BIOMIN, this reduction of ν is avoided.

Formula (3.15i) shows clearly that the approximations z_n are modified by moving along the directions u_n, and it allows us to express $z_n - z_0$ as sum of corrections:

$$z_n = z_0 + \sum_{j=0}^{n-1} u_j \omega_j \,, \tag{3.19}$$

a formula that was actually the starting point of Lanczos' application of the BOBC algorithm to linear systems [39, p. 37]. Since the directions u_n and v_n are biconjugate with respect to A (and the inner product based on B), cf. (3.3), the names biconjugate direction method for the nonsymmetric case and conjugate direction method for the case of a Hermitian matrix A (where $u_n = v_n$ if $x_0 = y_0$) are indeed appropriate. When A is Hermitian positive definite and $B = I$, one can say that for finding z_{n+1} one moves along the straigth line determined by the approximation z_n and the direction u_n until one reaches the minimum of the quadratic function $z \mapsto \frac{1}{2} z^H A z - b^H z$ on this line, or equivalently, the minimum of $z \mapsto (z^* - z)^H A (z^* - z)$. This is then also the minimum among all $z \in z_0 + \mathcal{K}_{n-1}$. At z_n the gradient (direction of steepest ascent) of this function happens to be $-x_n$. Thus the gradients (residuals) are orthogonal to each other. This geometric interpretation leads readily to a variety of generalizations of the conjugate gradient method to more general minimization problems. Moreover, one can do the same construction with respect to another inner product, in particular the one based on $B = A$. Then, one obtains the minimum of $z \mapsto (z^* - z)^H A^2 (z^* - z)$, which is also the minimum of $||b - Az||^2$, hence of $||b - Az||$, and the residuals are conjugate to each other. Therefore the resulting algorithm is called *minimal residual algorithm* or *conjugate residual algorithm.* Here we obtain more generally for non-Hermitian matrices a *biconjugate residual algorithm* if $B = A$ for which, however, the minimum residual property does no longer hold. In contrast, for the *generalized minimum residual algorithm* (ORTHOMIN with $B = A^*$ [73] or GMRES [59]) this property still holds if the recurrence is not truncated.

Formula (3.15i) suggest also that the sequence $\{z_n\}$ can be computed by building up the biconjugate directions u_n and v_n and using the vectors x_n and y_n only for the determination of the steplength $1/\varphi_n$, which depends on them, cf. (3.1a). Since the biconjugate sequences can be thought of as being biorthogonal with respect to the formal inner product $< v, u >_{BA} := v^H B A u$ (which is a true inner product if BA is Hermitian positive definite), they can be built up using the BO process if the new inner product is inserted there. The analogue of (2.19b) can be simplified by noting that due to (3.3) and (3.4) $< A^H v_{n-1}, A u_n >_B = < v_n, A u_n >_B \gamma'_{n-1}$. Since the direction vectors produced may differ in scaling from those generated by BIOMIN, we call them here u'_n and v'_n. (The prime does *not* denote transposition.)

ALGORITHM 7 (BO ALGORITHM FOR THE BA-NORM: "BICONJUGATION (BC) ALGORITHM"). *Choose u'_0, $v'_0 \in \mathbf{C}^N$ such that $\delta'_0 := < v'_0, Au'_0 >_B \neq 0$, and set $\beta'_0 := 0$. For $n = 0, 1, \ldots$ compute*

$$\alpha'_n := < A^H v'_n, Au'_n >_B / \delta'_n \,, \tag{3.20a}$$

$$\beta'_n := < v'_n, Au'_n >_B \gamma'_{n-1}/\delta'_{n-1} = \gamma'_{n-1}\delta'_n/\delta'_{n-1} \quad (\textit{if}\, n > 0), \tag{3.20b}$$

$$\tilde{u}_{n+1} := Au'_n - u'_n\alpha'_n - u'_{n-1}\beta'_n \,, \tag{3.20c}$$

$$\tilde{v}_{n+1} := A^H v'_n - v'_n\overline{\alpha'_n} - v'_{n-1}\overline{\beta'_n} \,, \tag{3.20d}$$

$$\tilde{\delta}'_{n+1} := < \tilde{v}_{n+1}, A\tilde{u}_{n+1} >_B ; \tag{3.20e}$$

if $\tilde{\delta}'_{n+1} = 0$, set for some $\gamma'_n \neq 0$

$$\nu' := n+1 \,, \quad u'_{\nu'} := \tilde{u}_{\nu'}/\gamma'_n \,, \quad v'_{\nu'} := \tilde{v}_{\nu'}/\overline{\gamma'_n} \,, \quad \delta'_{n+1} := 0, \tag{3.20f}$$

and stop; otherwise, choose γ'_n and δ'_{n+1} such that

$$(\gamma'_n)^2\delta'_{n+1} = \tilde{\delta}'_{n+1} \,, \tag{3.20g}$$

set

$$u'_{n+1} := \tilde{u}_{n+1}/\gamma'_n, \quad v'_{n+1} := \tilde{v}_{n+1}/\overline{\gamma'_n} \,, \tag{3.20h}$$

and proceed with the next step.

From Theorem 2.1 we conclude immediately:

COROLLARY 3.4. *The sequences $\{u'_n\}_{n=0}^{\nu'}$ and $\{v'_n\}_{n=0}^{\nu'}$ generated by the BC algorithm are biconjugate (with respect to A) except that $< v'_{\nu'}, Au'_{\nu'} >_B = 0$: For $m, n = 0, 1, \ldots, \nu'$,*

$$< v'_m, Au'_n >_B = \begin{cases} 0\,, & m \neq n, \\ \delta'_n\,, & m = n, \end{cases} \tag{3.21}$$

where $\delta_n \neq 0$ for $0 \leq n \leq \nu' - 1$. Moreover, for $n = 1, \ldots, \nu' - 1$ statement (3.4) is valid for u'_n and v'_n. Conversely, the sequences $\{u'_n\}_{n=0}^{\nu'}$ and $\{v'_n\}_{n=0}^{\nu'}$ are uniquely determined up to scaling by the condition (3.20) with $\delta_n \neq 0$ $(n = 0, 1, \ldots, \nu' - 1)$, $\delta_{\nu'} \neq 0$, and the assumption $u_n \in \mathcal{K}_n$, $v_n \in \mathcal{L}_n$ $(n = 0, 1, \ldots, \nu')$.

For the application to linear equations again an additional recurrence for z_n is needed. From the beginning we want to take advantage of the freedom in scaling the direction vectors u'_n, i.e., of the free choice of γ'_n, in order to avoid the breakdown due to $\varphi_n = 0$ in BIOMIN. We have then to compensate for this scaling in order to get the same approximants z_n and the same residuals x_n. However, this occurs implicitly and is not so apparent from the formulas:

ALGORITHM 8 (BC ALGORITHM FOR LINEAR EQUATIONS: BIODIR). *For solving $Az = b$ choose an initial approximation z_0, set $u'_0 := x_0 := b - Az_0$, and apply Algorithm 7, computing additionally*

$$\omega'_n := < v'_n, x_n >_B / \delta'_n \,, \tag{3.22a}$$

$$z_{n+1} := z_n + u'_n\omega'_n\,. \tag{3.22b}$$

$$x_{n+1} := x_n - Au'_n\omega'_n\,, \tag{3.22c}$$

If $x_{n+1} = o$ the process terminates and z_{n+1} is the solution; if $x_{n+1} \neq o$ but $\delta'_{n+1} = 0$ the algorithm breaks down. In both cases we set $nu' := n + 1$.

Again, x_n is the nth residual. For by assuming that $b - Az_n = x_n$ and using (3.22b) and (3.22c) we get

$$b - Az_{n+1} = b - Az_n - Au'_n\omega'_n = x_n - Au'_n\omega'_n = x_{n+1}\,,$$

as required.

Note that Biodir can only break down due to $\delta'_{n+1} = 0$. Even when $\omega'_n = 0$, the algorithm can recover: although $z_{n+1} = z_n$ and $x_{n+1} = x_n$, a new direction u'_{n+1} is created. In this case the method cannot be equivalent to Biores or Biomin since necessarily $x_{n+1} \neq x_n$ in both; but if $\omega'_n \neq 0$, it is indeed equivalent to these two methods as was noted in [34, p. 411] and is proven next. In summary we get the following comparison of breakdown conditions:

Theorem 3.5. *Assume that the same initial approximation z_0 and the same initial vectors $u_0 := u'_0 := x_0 := b - Az_0$ and $v_0 := v'_0 := y_0$ are used in BIODIR, normalized or unnormalized BIOMIN and normalized and unnormalized BIORES. Let ν', ν, and $\dot{\nu}$ be the breakdown indices of BIODIR, unnormalized BIORES, and the other three algorithms, respectively.*

Then $\dot{\nu} = \min\{\nu, \nu'\}$, and the following four conditions are equivalent:

(i') $\omega'_m \neq 0$ $(m < n)$, $< v'_n, x_n >= 0$ in BIODIR,

(ii') $\delta_n = 0$ in BIOMIN,

(iii') $\delta_n = 0$ in BIORES;

(iv') $n = \dot{\nu} = \nu$.

Condition (i') implies that either $\omega_n = 0$ or $\delta'_n = 0$ (in which case ω_n is undefined and $n = \nu'$).

Likewise the next four conditions are equivalent:

(i") $\delta'_n = 0$ in BIODIR,

(ii") $\varphi_n = 0$ in BIOMIN,

(iii") $\alpha_n + \beta_n = 0$ in BIORES.

(iv") $n = \dot{\nu} = \nu'$.

Except (i') for BIODIR and (iii") for unnormalized Biores *all these conditions are breakdown conditions for the respective algorithm.*

The three normalized algorithms produce the same approximants $z_1, \ldots, z_{\dot{\nu}}$ and hence the same residuals $x_1, \ldots, x_{\dot{\nu}}$. Also the biconjugate vectors u_n and v_n $(n = 0, \ldots, \dot{\nu})$ produced by BIOMIN are the same as the vectors u'_n and v'_n produced by BIODIR if in the latter

$$\gamma'_n := \gamma_n \; (= -\varphi_n = -1/\omega_n) \tag{3.23}$$

is chosen, which also implies that for $0 \leq n \leq \dot{\nu} - 1$ the parameters $\alpha'_n, \beta'_n, \gamma'_n$ of BIODIR are the elements of the matrix T_1 of (3.11). (Except for $\gamma'_{\dot{\nu}-1}$ which "lies outside" the matrix.) Otherwise holds more generally

$$u_n\Gamma_n = u'_n\Gamma'_n\,, \quad v_n\overline{\Gamma_n} = v'_n\,\overline{\Gamma'_n}\,, \tag{3.24}$$

where

$$\Gamma_n := \gamma_0\gamma_1\cdots\gamma_{n-1}\,, \quad \Gamma'_n := \gamma'_0\gamma'_1\cdots\gamma'_{n-1}\,, \tag{3.25}$$

and the tridiagonal matrix T_1' *with those parameters is given by*

$$D_{\Gamma'}^{-1} T_1' D_{\Gamma'} = D_{\Gamma}^{-1} T_1 D_{\Gamma} = D_{\Gamma}^{-1}(RL + e_{\dot\nu}\gamma_{\dot\nu-1}\psi_{\dot\nu} e_{\dot\nu}^T) D_{\Gamma} \,, \tag{3.26a}$$

where

$$D_{\Gamma} := \mathrm{diag}\,[1, \Gamma_1, \ldots, \Gamma_{\dot\nu-1}] \,, \quad D_{\Gamma'} := \mathrm{diag}\,[1, \Gamma_1', \ldots, \Gamma_{\dot\nu-1}'] \,, \tag{3.26b}$$

$$L = (S - I) D_{\gamma} = (I - S) D_{\varphi} \,. \tag{3.26c}$$

The steplength ω_n' *in* BIODIR *can be expressed as*

$$\omega_n' = \omega_n \Gamma_n' / \Gamma_n = -\Gamma_n' / \Gamma_{n+1} \,, \quad n = 0, 1, \ldots, \dot\nu - 1. \tag{3.27}$$

Proof. It is clear from the foregoing that up to the termination or breakdown of BIOMIN the formulas (3.20) of the BC algorithm produce the same biconjugate sequences $\{u_n'\}_{n=0}^{\dot\nu} = \{u_n\}_{n=0}^{\dot\nu}$ and $\{v_n'\}_{n=0}^{\dot\nu} = \{v_n\}_{n=0}^{\dot\nu}$ as BIOMIN if γ_n' is chosen appropriately, namely equal to the $(n+1, n)$-element of the matrix T_1 satisfying (3.13) and related to T by (3.11). (In fact, according to Theorem 2.1, applied with $B := BA$, these sequences are uniquely determined by the biconjugacy condition and the scaling.) Since T and T_1 have then the same subdiagonal elements, we need $\gamma_n' = \gamma_n$ for identical sequences. Choosing γ_n' differently, rescales the vectors u_n' and v_n', and from (3.1b), (3.1c), (3.1f), and (3.1g), it follows easily that (3.24) and (3.25) hold, which in view of (3.13) lead readily to (3.26).

In the formulas (3.22) a scale factor for u_n' and the conjugate factor for v_n' yield the inverse factor for ω_n', which cancels when x_{n+1} and z_{n+1} are evaluated. Hence, to prove that the latter two vectors are the same as in normalized BIOMIN it suffices to verify these formulas for a fixed scaling, say for the one induced by $\gamma_n' := \gamma_n = -\varphi_n$, when u_n' and v_n' are the same as u_n and v_n in BIOMIN. Assuming that the pair x_n, z_n is the same as in BIOMIN also and comparing (3.22b) and (3.22c) with (3.15c) and (3.15i) we see that the pair x_{n+1}, z_{n+1} is again the same as in BIOMIN if and only if

$$\omega_n' = \omega_n \quad (\text{if } \gamma_n' = \gamma_n \,, \text{ i.e., } \gamma_n' = -\varphi_n = -1/\omega_n). \tag{3.28}$$

By (3.2), (3.4) and (3.15h), $< y_n, x_n >_B = < v_n, x_n >_B = < v_n', x_n >_B$. Furthermore, if $\varphi_n \neq 0$, then by (3.2), (3.4), (3.20e)–(3.20h), and (3.22a) indeed

$$\omega_n = \frac{1}{\varphi_n} = \frac{\delta_n}{< v_n, Au_n >_B} = \frac{< y_n, x_n >_B}{< v_n', Au_n' >_B} = \frac{< v_n', x_n >_B}{\delta_n'} = \omega_n' \,.$$

This line exhibits also that $\omega_n' \neq 0, n = 0, \ldots, \dot\nu - 1$. Since $< y_n, x_n >_B = < v_n, x_n >_B$ is still valid for $n = \dot\nu$ the equivalence of (i')–(iii') holds, and since $\delta_{\dot\nu} = 0$ is the common breakdown condition of normalized and unnormalized BIORES, (iv) is equivalent too.

In view of $\Gamma_{\dot\nu} \neq 0, \Gamma_{\dot\nu}' \neq 0$, and (3.24), the condition $\varphi_{\dot\nu} = 0$ (which by definition of $\dot\nu$ implies $\delta_{\dot\nu} \neq 0$) is clearly equivalent to $\delta_{\dot\nu}' = 0$. Furthermore, since the normalized versions of BIOMIN and BIORES are related by $\varphi_n = -\gamma_n = \alpha_n + \beta_n$ and since α_n, β_n are the same in unnormalized BIORES, the equivalence of (i")–(iv") follows. Finally, since all conditions except (i') for BIODIR and (iii') for unnormalized BIORES imply a breakdown of the respective algorithm, and since no other types of breakdown exist, it follows that $\dot\nu \leq \nu'$ and $\dot\nu \leq \nu$, and that always one equal sign holds. □

What can be said about the case where $\dot{\nu} < \nu'$, i.e., $\omega'_n = 0$ for some $n < \nu'$? According to the formulas (3.22) there is no reason for the BIODIR algorithm to break down at such an n. The sequences $\{u'_m\}_{m=0}^{\nu'-1}$ and $\{v'_m\}_{m=0}^{\nu'-1}$ are still biconjugate (since they are generated by the BC algorithm), and x_m is still the residual at z_m. The following result will be proven in Sect. 4:

THEOREM 3.6. *The sequence $\{x_n\}_{n=0}^{\nu'-1}$ of residuals generated by BIODIR satisfies*

$$x_{n+1} \perp_B \mathcal{L}_n \quad \textit{if} \quad \omega'_n \neq 0\,, \tag{3.29a}$$

$$x_{n+1} \perp_B \mathcal{L}_{n-1} \quad \textit{if} \quad \omega'_n = 0\,, \tag{3.29b}$$

Moreover, if $\omega'_n = 0$ for some n, then $\omega'_{n-1} \neq 0$ (if $n > 0$) and $\omega'_{n+1} \neq 0$ (if $n < \nu' - 1$).

The formulas given by Jea and Young [34] for BIODIR (for real matrices) differ from ours for the equivalent BIODIR in that those authors also generate the sequence $\{y_n\}_{n=0}^{\nu'-1}$, using the recurrence

$$y_{n+1} := y_n - A^H v'_n \overline{\omega'_n}\,. \tag{3.30}$$

They then find ω'_n by taking the arithmetic mean between (3.22a) and the analogue formula involving u'_n and y_n:

$$\omega'_n := \frac{< v'_n, x_n >_B + < y_n, u'_n >_B}{2\delta'_n}\,. \tag{3.31}$$

As mentioned in Theorem 3.5, for $n = 0, \ldots, \dot{\nu} - 1$, the parameters $\alpha'_n, \beta'_n \gamma'_n$ of the BC algorithm and BIODIR are related to the paramters $\alpha_n, \beta_n, \gamma_n$ of the BO algorithm and BIORES by (3.26) and the equality $T = LR$. The additional parameters φ_n and ψ_n of the BOBC algorithm and BIOMIN appear as elements of the matrices R and L. Hence these three sets of parameters can be converted into each other. If we assume for simplicity that $\gamma'_n = \gamma_n$ ($n = 0, 1, \ldots, \dot{\nu} - 1$) we obtain (after setting $\psi_0 := \psi'_0 := 0$):

(i) from $T = LR$:

$$\alpha_n = \varphi_n + \gamma_{n-1}\psi_n\,, \quad \beta_{n+1} = \varphi_n \psi_{n+1}\,, \tag{3.32}$$

(ii) from $T_1 = RL$:

$$\alpha'_n = \varphi_n + \gamma_n \psi_{n+1}\,, \quad \beta'_{n+1} = \varphi_{n+1}\psi_{n+1}\,, \tag{3.33}$$

We will come across these formulas again in connection with the qd algorithm.

Let us at this point summarize the matrix relations that describe the five discussed iterative methods for nonsymmetric linear systems. Recall that depending on the method the order of the square matrices and the number of columns of the matrices X, Y, U, V, Z is $\nu, \dot{\nu}$ or ν', i.e., the column index goes from 0 to $\nu - 1, \dot{\nu} - 1$ or $\nu' - 1$. We set $e := [1, 1, \ldots, 1]^T$ and let S be the downshift matrix:

$$S := \begin{bmatrix} 0 & & & & \\ 1 & 0 & & & \\ & 1 & 0 & & \\ & & \ddots & \ddots & \\ & & & 1 & 0 \end{bmatrix}. \tag{3.34}$$

Normalized BIORES:

$$AX = XT + x_{\dot\nu}\gamma_{\dot\nu-1}e_{\dot\nu}^T, \quad A^HY = Y\overline{T} + y_{\dot\nu}\overline{\gamma_{\dot\nu-1}}e_{\dot\nu}^T\,, \tag{3.35a}$$

$$X = -ZT - z_{\dot\nu}\gamma_{\dot\nu-1}e_{\dot\nu}^T\,, \tag{3.35b}$$

where

$$Y^HBX = D_\delta\,, \quad Y^HBAX = D_\delta T\,, \tag{3.35c}$$

$$e^TT = \alpha_{\dot\nu-1} + \beta_{\dot\nu-1}e_{\dot\nu}^T\,. \tag{3.35d}$$

Unnormalized BIORES: (3.35a)-(3.35c) hold with $\dot\nu$ replaced by ν.
Normalized BIOMIN:

$$AUD_\omega = X(I-S) - x_{\dot\nu}e_{\dot\nu}^T\,, \quad A^HV\overline{D_\omega} = Y\overline{L} - y_{\dot\nu}e_{\dot\nu}^T\,, \tag{3.36a}$$

and

$$X = UR, \quad Y = V\overline{R}, \tag{3.36b}$$

$$UD_\omega = Z(S-I) + z_{\dot\nu}e_{\dot\nu}^T\,, \tag{3.36c}$$

where

$$Y^HBX = D_\delta\,, \quad V^HBAU = D_\delta D_\omega^{-1}\,, \tag{3.36d}$$

$$R = D_\delta^{-1}(S^T + I)D_\delta\,. \tag{3.36e}$$

Unnormalized BIOMIN*:*

$$AU = XL + x_{\dot\nu}\gamma_{\dot\nu-1}e_{\dot\nu}^T\,, \quad A^HV = Y\overline{L} + y_{\dot\nu}\overline{\gamma_{\dot\nu-1}}e_{\dot\nu}^T\,, \tag{3.37a}$$

and

$$X = UR, \quad Y = V\overline{R}, \tag{3.37b}$$

$$U = -ZL - z_{\dot\nu}\gamma_{\dot\nu-1}e_{\dot\nu}^T\,, \tag{3.37c}$$

where

$$Y^HBX = D_\delta\,, \quad V^HBAU = D_\delta D_\varphi\,, \tag{3.37d}$$

$$L = D_\varphi + SD_\gamma\,, \quad R = I + D_\varphi^{-1}D_\gamma D_\delta^{-1}S^TD_\delta \tag{3.37e}$$

BIODIR:

$$AU' = U'T_1' + u_{\dot\nu}'\gamma_{\dot\nu-1}'e_{\dot\nu}^T\,, \quad A^HV' = V'\overline{T_1'} + v_{\dot\nu}'\overline{\gamma_{\dot\nu-1}'}e_{\dot\nu}^T\,. \tag{3.38a}$$

$$U'D_{\omega'} = Z(S-I) + z_{\nu'}e_{\nu'}^T\,, \quad AU'D_{\omega'} = X(I-S) + x_{\nu'}e_{\nu'}^T\,, \tag{3.38b}$$

where one may choose $\gamma'_n = 1$ $(\forall n)$, and where for some strictly lower triangular matrix K',

$$V'^H BX = D_{\delta'}D_{\omega'} + K'\,, \quad V'^H BAU' = D_{\delta'}\,, \quad V'^H BA^2U' = D_{\delta'}T'_1\,. \tag{3.38c}$$

If $\nu = \nu' = \dot\nu$, or, in any case, if T and T'_1 are replaced by their $\dot\nu \times \dot\nu$ principal submatrix, the normalized methods are related by the common matrices X, Y, Z, by (3.26) and by

$$UD_\Gamma = U'D_{\Gamma'}\,, \quad V\overline{D_\Gamma} = V'\overline{D_{\Gamma'}}\,, \tag{3.39}$$

$$D_{\omega'} = D_{\Gamma'}D_\omega D_\Gamma^{-1}\,, \quad D_\omega = D_\varphi^{-1} = -D_\gamma^{-1}\,. \tag{3.40}$$

If, for the moment, the matrices of the unnormalized methods are distinguished by a dot, one has

$$\dot X = XD_{\dot\rho}\,, \quad \dot Y = Y\overline{D_{\dot\rho}}\,, \quad \dot Z = ZD_{\dot\rho}\,, \tag{3.41}$$

$$D_{\dot\delta} = D_\delta D_{\dot\rho}^2\,, \quad D_{\dot\rho}\dot T = TD_{\dot\rho}\,. \tag{3.42}$$

In all cases holds the residual property in the form

$$X = -AZ + be^T D_{\dot\rho}\,, \tag{3.43}$$

where $D_{\dot\rho} = I$ for the normalized methods (including Biodir).

In view of

$$(I-S)^{-1} = I + S + S^2 + \cdots + S^{\dot\nu-1} = \begin{pmatrix} 1 & & & & \\ 1 & 1 & & & \\ 1 & 1 & 1 & & \\ \vdots & \vdots & \vdots & \ddots & \\ 1 & 1 & 1 & \cdots & 1 \end{pmatrix} \tag{3.44}$$

formula (3.17) can be written as

$$Z - z_0e^T = UD_\omega[(I-S)^{-T} - I]\,. \tag{3.45}$$

By right-multiplication with $(I-S)^T$ one can regain (3.36c).

4. The relation to formal orthogonal polynomials and Padé approximation. Any vector x in the Krylov space $\mathcal{K}_n$ defined in (2.1) can be written in the form $x = \rho(A)x_0$, where ρ is a polynomial of degree at most n, in which the matrix A is substituted for the argument. Associated to the sequence $\{x_n\}_{n=0}^{\nu}$ defined by the BO algorithm there is therefore a sequence $\{\rho_n\}_{n=0}^{\nu}$ of polynomials, and from the relation (2.18a) one gets readily a three-term recurrence formula for these polynomials. The analogue is true for the sequence $\{y_n\}_{n=0}^{\nu}$ with $y_n \in \mathcal{L}_n$ and the operator A^H, and in view of (2.18b) the coefficients of the corresponding polynomials are just complex conjugate to those of the polynomials ρ_n. Similarly, from the formulas (3.1) of the BOBC algorithm it is readily seen that the vectors u_n and v_n can be represented in this way also. Recurrences for the corresponding polynomials σ_n follow from these formulas (3.1) as well as from the formulas (3.20) of the BC algorithm, which is known to produce the same vector sequences if $\gamma'_n = \gamma_n$, and rescaled ones if $\gamma'_n \neq \gamma_n$. Altogether one obtains:

THEOREM 4.1. *Let $\{x_n\}_{n=0}^{\nu}$ and $\{y_n\}_{n=0}^{\nu}$ be the biorthogonal vector sequences generated by the BO algorithm, and let $\{u_n\}_{n=0}^{\nu}$ and $\{v_n\}_{n=0}^{\nu}$ be the biconjugate vector sequences generated by the BC algorithm using $\gamma'_n = \gamma_n$. Then there is a pair of sequences of polynomials, $\{\rho_n\}_{n=0}^{\nu}$ and $\{\sigma_n\}_{n=0}^{\nu}$, such that*

$$x_n = \rho_n(A)x_0 \,, \quad y_n = \overline{\rho_n}(A^H)y_0 \,, \quad n = 0, 1, \ldots, \nu, \tag{4.1}$$

and

$$u_n = \sigma_n(A)u_0, \quad v_n = \overline{\sigma_n}(A^H)v_0 \,, \quad n = 0, 1, \ldots, \nu. \tag{4.2}$$

For $n < \dot{\nu} := \min\{\nu, \nu\}$ they satisfy the mixed three-term recurrences

$$\begin{aligned} &\rho_0(\zeta) :\equiv 1, \quad \rho_{n+1}(\zeta) := [\zeta\sigma_n(\zeta) - \varphi_n\rho_n(\zeta)]\,/\gamma_n \,, \\ &\sigma_0(\zeta) :\equiv 1, \quad \sigma_{n+1}(\zeta) := \rho_{n+1}(\zeta) - \psi_{n+1}\sigma_n(\zeta), \quad n = 0, \ldots, \dot{\nu} - 1. \end{aligned} \tag{4.3}$$

They also satisfy individual three-term recurrences, namely

$$\begin{aligned} &\rho_0(\zeta) :\equiv 1, \quad \rho_1(\zeta) := (\zeta - \alpha_0)\rho_0(\zeta)/\gamma_0 \,, \\ &\rho_{n+1}(\zeta) := [(\zeta - \alpha_n)\rho_n(\zeta) - \beta_n\rho_{n-1}(\zeta)]\,/\gamma_n \,, \quad n = 0, 1, \ldots, \nu - 1, \end{aligned} \tag{4.4}$$

and

$$\begin{aligned} &\sigma_0(\zeta) :\equiv 1, \quad \sigma_1(\zeta) := (\zeta - \alpha'_0)\sigma_0(\zeta)/\gamma_0 \,, \\ &\sigma_{n+1}(\zeta) := [(\zeta - \alpha'_n)\sigma_n(\zeta) - \beta'_n\sigma_{n-1}(\zeta)]\,/\gamma_n \,, \quad n = 0, 1, \ldots, \nu' - 1, \end{aligned} \tag{4.5}$$

respectively. Both ρ_n and σ_n have exact degree n, and both have the leading coefficient Γ_n^{-1}, where

$$\Gamma_n := \gamma_0\gamma_1 \cdots \gamma_{n-1} \,. \tag{4.6}$$

If $\gamma_n = -\varphi_n$ $(\forall n)$ as in (3.14) or, equivalently, if $\gamma_n = -\alpha_n - \beta_n$ $(\forall n)$ as in (2.25), then

$$\rho_n(0) = 1, \quad n = 1, 2, \ldots, \nu. \tag{4.7}$$

Since x_n is the nth residual of the three normalized Lanczos based algorithms for solving linear systems that have been discussed in Sects. 2 and 3, ρ_n is for each of these methods the so-called *residual polynomial.* Property (4.7) is the standard consistency condition for residual polynomials of polynomial acceleration methods. The polynomials ρ_n are sometimes called *Lanczos polynomials* [23]. In classical analysis they also go, depending on the normalization, under the names *Hankel polynomials* and *Hadamard polynomials* [30]. If $x_{\dot{\nu}} = o$, then $\rho_{\dot{\nu}}$ is the minimal polynomial of A with respect to the invariant Krylov subspace $\mathcal{K}_{\dot{\nu}-1}$.

The properties (2.2) and (3.3) translate in view of (4.1), (4.2), and $BA = AB$ into

$$y_0^H B\rho_m(A)\rho_n(A)x_0 = \delta_{mn}\delta_n \,, \quad n = 0, 1, \ldots, \nu, \tag{4.8}$$

and

$$y_0^H B\sigma_m(A)A\sigma_n(A)x_0 = \delta_{mn}\delta_n\varphi_n \,. \quad n = 0, 1, \ldots, \nu'. \tag{4.9}$$

Let us introduce two linear functionals Φ and Φ_1 defined on the set $\mathcal{P}$ of all polynomials by the values they take on the monomial bases:

$$\Phi(\zeta^k) := \mu_k, \quad \Phi_1(\zeta^k) := \mu_{k+1} \quad (k \in \mathbf{N}), \tag{4.10}$$

where

$$\mu_k := < y_0, A^k x_0 >_B = y_0^H B A^k x_0 \,. \tag{4.11}$$

μ_k is called the kth *moment* or *Schwarz constant* associated to A, x_0, and y_0. Now (4.8) and (4.9) become

$$\Phi(\rho_m \rho_n) = \delta_{mn}\delta_n \,, \quad m, n = 0, 1, \ldots, \nu, \tag{4.12}$$

$$\Phi_1(\sigma_m \sigma_n) = \delta_{mn}\delta_n \varphi_n \,, \quad m, n = 0, 1, \ldots, \nu', \tag{4.13}$$

which by definition means that $\{\rho_n\}_{n=0}^{\nu}$ is a sequence of *(regular) formal orthogonal polynomials* [6] with respect to the functional Φ, and that $\{\sigma_n\}_{n=0}^{\nu'}$ is such a sequence for the functional Φ_1. From the linearity one concludes that

$$\Phi(\zeta^m \rho_n) = \begin{cases} 0 & \text{if } 0 \le m < n \,, \\ \delta_n \Gamma_n & \text{if } m = n \,, \end{cases} \tag{4.14}$$

and

$$\Phi_1(\zeta^m \pi_n) = \begin{cases} 0 & \text{if } 0 \le m < n \,, \\ \delta_n \varphi_n \Gamma_n & \text{if } m = n \,, \end{cases} \tag{4.15}$$

which is another way to characterize (or define) these polynomial sequences. If ρ_n and σ_n are expressed in the monomial basis,

$$\rho_n(\zeta) =: \sum_{k=0}^{n} \rho_k^{(n)} \zeta^k \tag{4.16}$$

and

$$\sigma_n(\zeta) =: \sum_{k=0}^{n} \sigma_k^{(n)} \zeta^k \,, \tag{4.17}$$

then the first n equations of (4.14) and (4.15) become homogenous $n \times (n+1)$ Hankel systems for the coefficients $\rho_k^{(n)}$ and $\sigma_k^{(n)}$, respectively:

$$\begin{bmatrix} \mu_0 & \mu_1 & \cdots & \mu_n \\ \mu_1 & \mu_2 & & \mu_{n+1} \\ \vdots & & & \vdots \\ \mu_{n-1} & \mu_n & \cdots & \mu_{2n-1} \end{bmatrix} \begin{bmatrix} \rho_0^{(n)} \\ \rho_1^{(n)} \\ \vdots \\ \rho_n^{(n)} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix} , \tag{4.18}$$

$$\begin{bmatrix} \mu_1 & \mu_2 & \cdots & \mu_{n+1} \\ \mu_2 & \mu_3 & & \mu_{n+2} \\ \vdots & & & \vdots \\ \mu_n & \mu_{n+1} & \cdots & \mu_{2n} \end{bmatrix} \begin{bmatrix} \sigma_0^{(n)} \\ \sigma_1^{(n)} \\ \vdots \\ \sigma_n^{(n)} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix} . \tag{4.19}$$

Since there are more unknowns than equations, there must be a nontrivial subspace of solutions. In fact, since (4.14) and (4.15) are equivalent to (2.1)–(2.2) and (3.3)–(3.4), respectively, it follows from the uniqueness statements in Theorem 2.1 and Corollary 3.4 that for each n, $0 \leq n \leq \nu$, the solution of (4.18) is unique up to a scalar factor and it has a nonvanishing last component $\rho_n^{(n)} \neq 0$, and, likewise, for each n, $0 \leq n \leq \nu'$, (4.19) has up to a scalar factor a unique solution and this solution has a nonvanishing last component $\sigma_n^{(n)} \neq 0$. Hence, given $\{\Gamma_m\}_{m=0}^n$, there is a unique solution with $\rho_n^{(n)} = \sigma_n^{(n)} = 1/\Gamma_n$, which must be the one corresponding to the polynomials ρ_n and σ_n satisfying the recurrence (4.3). Clearly, this existence and uniqueness statement is equivalent to the nonsingularity, for $0 \leq n \leq \nu$, or $0 \leq n \leq \nu'$, respectively, of the submatrices obtained by deleting the last column in the matrices of (4.18) and (4.19); we denote these matrices by M_{n-1} and M'_{n-1}:

LEMMA 4.2. *Under the assumptions of Theorem 4.1 the Hankel matrices*

$$M_n := \begin{bmatrix} \mu_0 & \mu_1 & \cdots & \mu_n \\ \mu_1 & \mu_2 & & \mu_{n+1} \\ \vdots & & & \vdots \\ \mu_n & \mu_{n+1} & \cdots & \mu_{2n} \end{bmatrix}, \quad M'_n := \begin{bmatrix} \mu_1 & \mu_2 & \cdots & \mu_{n+1} \\ \mu_2 & \mu_3 & & \mu_{n+2} \\ \vdots & & & \vdots \\ \mu_{n+1} & \mu_{n+2} & \cdots & \mu_{2n+1} \end{bmatrix} \tag{4.20}$$

are for $n = 0, 1, \ldots, \nu - 1$ and $n = 0, 1, \ldots, \nu' - 1$, respectively, nonsingular.

The matrices M_n and M'_n are the *moment matrices* for the functionals Φ and Φ_1, respectively. The above lemma is well-known in Padé theory [22], which also says that M_ν and $M'_{\nu'}$ are singular, as we will see in a moment.

In the above derivation we have made heavy use of the existence and uniqueness statements in Theorem 2.1 and Corollary 3.4, which we obtained in a somewhat indirect way. There is another more direct approach to them and to the lemma:

Let

$$P_n := \begin{bmatrix} \rho_0^{(0)} & \rho_0^{(1)} & \cdots & \rho_0^{(n)} \\ & \rho_1^{(1)} & & \rho_1^{(n)} \\ & & \ddots & \vdots \\ & & & \rho_n^{(n)} \end{bmatrix} \tag{4.21}$$

be the upper triangular $(n+1) \times (n+1)$ matrix containing in its columns the coefficients of the polynomials ρ_m, and let X_n and Y_n be the $N \times (n+1)$ matrices with columns x_m and y_m, $m = 0, 1, \ldots, n$, respectively. (Hence, $X_{\nu-1} = X$, $Y_{\nu-1} = Y$ in the notation of Sects. 2 and 3.) Then, in view of (4.1),

$$[x_0, Ax_0, \ldots, A^n x_0] = X_n P_n^{-1}, \quad [y_0, A^H y_0, \ldots, (A^H)^n y_0] = Y_n \overline{P}_n^{-1}, \tag{4.22}$$

which is just a reformulation of condition (2.1). On the other hand,

$$\begin{bmatrix} y_0^H \\ y_0^H A \\ \vdots \\ y_0^H A^n \end{bmatrix} B[x_0, Ax_0, \ldots, A^n x_0] = M_n \,. \tag{4.23}$$

Inserting (4.22) and the orthogonality property $Y_n^H B X_n = D_{\delta,n} := \text{diag}[\delta_0, \delta_1, \ldots, \delta_n]$ (cf. (2.7)) into (4.23) yields

$$P_n^{-T} D_{\delta,n} P_n^{-1} = M_n \,. \tag{4.24}$$

Since P_n^{-1} is an upper triangular matrix (which contains the coeffcents of the monomials when expressed as linear combinations of the orthogonal polynomials), (4.24) means that $P_n^T D_{\delta,n} P_n$ is a (symmetric) LDU decomposition of the nth moment matrix M_n if the polynomials are monic. (Otherwise it is a nonstandard LDU decomposition with prescribed diagonal elements of P_n^{-1}.)

Conversely, if we have such an LDU decomposition, it yields after inversion of P^{-1} the coefficients of monic formal orthogonal polynomials ρ_m $(m = 0, 1, \dots, n)$ associated with the moments μ_k. They satisfy $\Phi(\rho_m \rho_k) = \delta_{m,k}\delta_m$ $(m, k = 0, 1, \dots, n)$.

Now, the LDU factors of M_n are the principal submatrices of order $n+1$ of the LDU factors of M_{n+1}, if the LDU decomposition (with no pivoting) of the latter matrix exist. For the existence we need that $\delta_n \neq 0$, i.e., that M_n is nonsingular. If then $\delta_{n+1} = 0$, M_{n+2} will not have an LDU decomposition however (unless all elements that have to be divided by the pivot $\delta_{n+1} = 0$ are zero). In other words, the computation of the LDU decomposition of the infinite moment matrix M_∞ breaks down after we have encountered the first singular principal submatrix M_ν. Hence, Lemma 4.2 and the singularity of M_ν and $M'_{\nu'}$ follow.

Next, we let

$$F(\zeta) := < y_0, (\zeta I - A)^{-1} x_0 >_B = \sum_{k=0}^{\infty} \frac{\mu_k}{\zeta^{k+1}}\,, \tag{4.25}$$

a definition given on the first page of Rutishauser's paper [54]. From the well-known explicit formula expressing $(\zeta I - A)^{-1}$ as a quotient of a matrix (whose elements are the algebraic complements of the elements of $(\zeta I - A)^{-1}$) and the determinant of $(\zeta I - A)^{-1}$ it follows that F is a rational function of order at most N, which is analytic at ∞. Since $\mu_0 \neq 0$ by the assumption that x_0 and y_0 are nonorthogonal (in Algorithms 1–8), the numerator degree is known to be exactly by 1 smaller than the denominator degree.

The products $F(\zeta)\rho_n(\zeta)$ and $\zeta F(\zeta)\sigma_n(\zeta)$ can be considered as Laurent series with terms ζ^k, $-\infty < k < n$ and $-\infty < k \leq n$, respectively. But in view of (4.18) and (4.19) n of the terms in these series are zero, and if we choose the polynomials $\pi_n \in \mathcal{P}_{n-1}$ and $\tau_n \in \mathcal{P}_n$ equal to the 'analytic parts' of the series, it follows that

$$F(\zeta)\rho_n(\zeta) - \pi_n(\zeta) = O(\zeta^{-n-1})\,, \tag{4.26}$$

$$\zeta F(\zeta)\sigma_n(\zeta) - \tau_n(\zeta) = O(\zeta^{-n-1})\,, \tag{4.27}$$

as $\zeta \to \infty$. (This means that the right-hand sides are Laurent series with terms ζ^{-k}, $k \geq n+1$.) At this point we have to recall the definition of the Padé approximants of F:

Definition. Let F be given by the series in (4.25), and let π^* and ρ^* be polynomials of degree at most $m-1$ and n, respectively, such that

$$F(\zeta)\rho^*(\zeta^{-1}) - \zeta^{-1}\pi^*(\zeta^{-1}) = O(\zeta^{-m-n-1}), \tag{4.28}$$

or, equivalently, let $\pi(\zeta) := \zeta^{m-1}\pi^*(\zeta^{-1})$ and $\rho(\zeta) := \zeta^n \rho^*(\zeta^{-1})$ be two such polynomials satisfying

$$F(\zeta)\rho(\zeta) - \zeta^{m-n}\pi(\zeta) = O(\zeta^{-m-1}), \tag{4.29}$$

Then the rational function

$$\frac{\zeta^{-1}\pi^*(\zeta^{-1})}{\rho^*(\zeta^{-1})} = \zeta^{n-m}\frac{\pi(\zeta)}{\rho(\zeta)} \tag{4.30}$$

is called the (m,n) *Padé approximant of F at ∞*.[3] (Note that if $m \geq n$ and the numerator and the denominator are expressed in ζ, the type of the (m,n) approximant is $(n-1,n)$.)

In view of (4.26)–(4.27) and $\partial\pi_n = n-1$, $\partial\rho_n = \partial\tau_n = \partial\sigma_n = n$ the rational fractions

$$\frac{\pi_n(\zeta)}{\rho_n(\zeta)} \quad \text{and} \quad \frac{\tau_n(\zeta)}{\zeta\sigma_n(\zeta)} \tag{4.31}$$

are therefore the (n,n) and the $(n+1,n)$ Padé approximant, respectively, of F at ∞. Since $\rho_n^*(0) = \rho_n^{(n)} \neq 0$ and $\sigma_n^*(0) = \sigma_n^{(n)} \neq 0$, one has also

$$F(\zeta) - \frac{\pi_n(\zeta)}{\rho_n(\zeta)} = O(\zeta^{-2n-1}), \quad F(\zeta) - \frac{\tau_n(\zeta)}{\zeta\sigma_n(\zeta)} = O(\zeta^{-2n-2}), \tag{4.32}$$

i.e., these Padé approximants interpolate F at ∞ in the Hermite sense up to the indicated order.

These approximants lie on two adjacent *diagonals* of the *Padé table*, a table occupying the lower right quadrant of the (m,n)-plane (with the m-axis pointing downwards) and containing as its (m,n) entry the (m,n) Padé approximant. The approximants $\{\pi_n/\rho_n\}_{n=0}^{\nu}$, which are related to the BO algorithm (and to unnormalized BIORES) lie on the main diagonal, the approximants $\{\tau_n/\sigma_n\}_{n=0}^{\nu'}$, which belong to the BC algorithm (and to BIODIR) furnish the initial section of the first subdiagonal, and the approximants corresponding to the BOBC algorithm (and to BIOMIN), if written in the order $\pi_0/\rho_0 = 0$, $\tau_0/(\zeta\sigma_0)$, π_1/ρ_1, $\tau_1/(\zeta\sigma_1)$, π_2/ρ_2, $\tau_2/(\zeta\sigma_2), \ldots, \pi_{\dot\nu}/\rho_{\dot\nu}$, $\tau_{\dot\nu}/(\zeta\sigma_{\dot\nu})$ form a finite *staircase* in this table. In the next section we will describe how the same Padé approximants can be obtained as convergents ("partial sums") of a continued fraction.

Since

$$\beta_n \neq 0, \qquad n = 1,2,\ldots,\nu-1, \tag{4.33a}$$
$$\beta'_n \neq 0, \qquad n = 1,2,\ldots,\nu'-1, \tag{4.33b}$$
$$\psi_n \neq 0, \qquad n = 1,2,\ldots,\dot\nu-1, \tag{4.33c}$$
$$\varphi_n \neq 0, \qquad n = 0,1,\ldots,\dot\nu-1, \tag{4.33d}$$

it follows from the recurrences (4.3)–(4.5) that in each of the three cases the listed approximants are distinct (except for $\pi_{\dot\nu}/\rho_{\dot\nu} = \tau_{\dot\nu}/(\zeta\sigma_{\dot\nu})$ if $\psi_{\dot\nu} = 0$), which is another fact well-known in Padé and continued fraction theory. We will see in Sect. 5 that the numerators π_n and τ_n satisfy the same recurrence formulas as the denominators. Therefore, if we try to extend the recurrences beyond the breakdown index, we see that the following equivalences hold:

$$\beta_\nu = 0 \iff \pi_n u/\rho_n u = \pi_{\nu+1}/\rho_{\nu+1}\,, \tag{4.34a}$$
$$\beta'_{\nu'} = 0 \iff \tau_{\nu'}/(\zeta\sigma_{\nu'}) = \tau_{\nu'+1}/(\zeta\sigma_{\nu'+1})\,, \tag{4.34b}$$
$$\psi_{\dot\nu} = 0 \iff \pi_n u/\rho_n u = \tau_{\dot\nu}/(\zeta\sigma_{\dot\nu})\,, \tag{4.34c}$$
$$\varphi_{\dot\nu} = 0 \iff \tau_{\dot\nu}/(\zeta\sigma_{\dot\nu}) = \pi_{\dot\nu+1}/\rho_{\dot\nu+1}\,, \tag{4.34d}$$

In other words, the breakdowns of the algorithms discussed in Sects. 2 and 3 are closely related to the occurence of identical entries in the Padé table of the function F defined in

[3] This definition is particularly adapted to (4.25), which is a series in ζ^{-1} with a vanishing constant term. The standard definition for Padé approximants is for formal power series in ζ and is therefore suitable for functions analytic at $\zeta = 0$, which are to be approximated there [22, p. 11].

FIG. 1. *Breakdown of the BOBC algorithm due to the occurence of nontrivial blocks in the Padé table of F. In both cases the block can be larger than shown, but its upper or right border, respectively, must lie on the dashed line. In the first case, the BO algorithm breaks also down, and the BC algorithm breaks down if the block has more than one entry in common with the first subdiagonal of the table. In the second case, the BC algorithm breaks also down, and the BO algorithm breaks down if the block has more than one entry in common with the main subdiagonal of the table.*

(4.25). The following basic result on identical entries in Padé tables plays therefore a role for understanding and possibly avoiding breakdowns:

THEOREM 4.3 (BLOCK STRUCTURE THEOREM FOR THE PADÉ TABLE). *Let the rational function $\zeta^{-1}\pi^*(\zeta^{-1})/\sigma^*(\zeta^{-1})$ be a nonzero Padé approximant of some power series F in ζ^{-1} with vanishing constant term. Then there is Δ, which is either a positive integer or ∞, so that this rational function is the (m,n) Padé approximant of F for all pairs (m,n) with $\partial\pi^* + 1 \leq m \leq \partial\pi^* + \Delta$, $\partial\sigma^* \leq n \leq \partial\sigma^* + \Delta - 1$ and for no other pairs, i.e., each set of identical entries in the Padé table fills a finite or infinite square with upper left corner at $(m,n) = (\partial\pi^*+1, \partial\sigma)$.*

If $\mu_k = 0$, $k = 0, \ldots, l-1$, and $\mu_l \neq 0$, then the first $l+1$ rows of the Padé table are filled with the zero function, which, on the other hand, appears nowhere else.

For a proof see Gragg [22].

If the above mentioned staircase has more than one entry in common with a block of size $\Delta > 1$, the BOBC algorithm necessarily breaks down since then $\rho_{n+1} = \zeta\sigma_n$ or $\sigma_{n+1} = \rho_{n+1}$, which means that $\varphi_n = 0$ or $\psi_{n+1} = 0$, cf. (4.3). Then $\varphi_{\dot\nu} = 0$ or $\psi_{\dot\nu} = 0$, respectively, by definition of $\dot\nu$. The two situations are depicted in Figure 1. Similar statements hold for the BO and the BC algorithms and the corresponding diagonals of the table:

THEOREM 4.4. *On the main diagonal of the Padé table of the function F defined by (4.25) the column index n of the first entry which is followed by an identical entry is equal to the breakdown index ν of the BO algorithm (and of unnormalized BIORES). Likewise, on the first subdiagonal of the Padé table the column index n of the first entry followed by an identical entry is the breakdown index ν' of the BC algorithm (and of BIODIR). The breakdown index $\dot\nu = \min\{\nu, \nu'\}$ of the BOBC algorithm (and of BIOMIN) is then the column index of the first entry followed by an identical entry on the staircase (0,0), (1,0), (1,1), (2,1), (2,2), ... of the table.*

The case where unnormalized BIORES does not break down, but BIODIR and BIOMIN do, can now be characterized as the situation where the main diagonal hits the upper right corner of a block of size $\Delta > 0$. Since the subdiagonal and the staircase have then more than one entry in common with this block, BIODIR and BIOMIN must break down. Analogously, the case where BIODIR does not break down, but BIORES (even unnormalized) and BIOMIN do, is characterized by the subdiagonal hitting the lower left corner of a block of size $\Delta > 0$.

Now we are also ready for

Proof of Theorem 3.6. Up to column ν' the first subdiagonal of the Padé table does not have more than one entry in common with a single block. Thus, since blocks are squares, on the main diagonal at most two successive entries can be identical. In fact, say, $\rho_n = \rho_{n+1}$ if and only if $\omega'_n = 0$. If these conditions are satisfied, ρ_n is a regular formal orthogal polynomial and thus satisfies the orthonality conditions (4.14); consequently, $x_{n+1} = x_n \perp \mathcal{L}_{n-1}$. If $\omega'_n \neq 0$, $x_{n+1} \neq x_n$, and hence ρ_{n+1} satisfies these conditions (with n replaced ny $n+1$). □

Since the Padé table of F plays such a prominent role in this discussion, it is worth looking at the function F more closely. Unfortunately, it is difficult to deduce from F information on the block structure of its table, except for obvious implications of symmetry, as, e.g., when F is even or odd. Then the Padé table is made up of 2×2 or larger blocks, hence only either unnormalized BIORES or BIODIR have a chance to work. But the determination of the exact order of F allows us at least to locate the position of the infinite block, where the Padé approximant is equal to F, and where all algorithms must terminate or break down if they have not broken down before.

Let us for a few lines assume that the matrices A and B are diagonalizable. Then, since they commute, they have a common complete system of eigenvalues [72, p. 52]. Hence, there is a nonsingular $N \times N$ matrix W of eigenvectors such that

$$AW = WD_\lambda\,, \quad BW = WD_\kappa \tag{4.35}$$

where D_λ and D_κ are diagonal matrices containing the eigenvalues $\lambda_1, \ldots, \lambda_N$ of A and the eigenvalues $\kappa_1, \ldots, \kappa_N$ of B. Then $A^H W^{-H} = W^{-H} D_\lambda^H$, i.e., the columns of W^{-H} are a set of eigenvectors of A^H. Set

$$W =: [w_1, w_2, \ldots, w_N], \quad W^{-H} =: [\hat{w}_1, \hat{w}_2, \ldots, \hat{w}_N]\,, \tag{4.36}$$

and represent x_0 in the basis $\{w_k\}$, y_0 in the basis $\{\hat{w}_k\}$:

$$x_0 =: \sum_{j=1}^N w_j \xi_j = W \begin{bmatrix} \xi_1 \\ \vdots \\ \xi_N \end{bmatrix}, \quad y_0 =: \sum_{j=1}^N \hat{w}_j \eta_j = W^{-H} \begin{bmatrix} \eta_1 \\ \vdots \\ \eta_N \end{bmatrix}. \tag{4.37}$$

Then

$$\mu_k := < y_0, A^k x_0 >_B = [\overline{\eta}_1, \ldots, \overline{\eta}_N] D_\kappa D_\lambda^k \begin{bmatrix} \xi_1 \\ \vdots \\ \xi_N \end{bmatrix} = \sum_{j=1}^N \kappa_j \lambda_j^k \xi_j \overline{\eta}_j\,, \tag{4.38}$$

and

$$F(\zeta) := < y_0, (\zeta I - A)^{-1} x_0 >_B = [\overline{\eta}_1, \ldots, \overline{\eta}_N] D_\kappa (\zeta I - D_\lambda)^{-1} \begin{bmatrix} \xi_1 \\ \vdots \\ \xi_N \end{bmatrix} = \sum_{j=1}^N \frac{\kappa_j \xi_j \overline{\eta}_j}{\zeta - \lambda_j}. \tag{4.39}$$

(4.38) means that μ_k is the kth moment of the *discrete measure* with the masses $\kappa_j \xi_j \overline{\eta}_j$ at the N points λ_j.

If A and B are *not* diagonalizable, most of the above still holds if D_λ and D_κ denote the Jordan canonical forms of A and B and if (4.35) holds. Only the sums in (4.38) and (4.39) have to be modified. In $(\zeta I - D_\lambda)^{-1}$ each Jordan subblock of size, say, $l \times l$, belonging to a nonlinear divisor has the form

$$\begin{bmatrix} \zeta - \lambda & -1 & & & \\ & \zeta - \lambda & -1 & & \\ & & \zeta - \lambda & \ddots & \\ & & & \ddots & -1 \\ & & & & \zeta - \lambda \end{bmatrix}^{-1} = \begin{bmatrix} (\zeta-\lambda)^{-1} & (\zeta-\lambda)^{-2} & (\zeta-\lambda)^{-3} & \cdots & (\zeta-\lambda)^{-l} \\ & (\zeta-\lambda)^{-1} & (\zeta-\lambda)^{-2} & & (\zeta-\lambda)^{-l+1} \\ & & (\zeta-\lambda)^{-1} & & (\zeta-\lambda)^{-l+2} \\ & & & \ddots & \vdots \\ & & & & (\zeta-\lambda)^{-1} \end{bmatrix}.$$

Consequently, if we assume $B = I$ for simplicity, (4.39) has to be replaced by
(4.40)

$$F(\zeta) := < y_0, (\zeta I - A)^{-1} x_0 >_I = [\overline{\eta}_1, \ldots, \overline{\eta}_N](\zeta I - D_\lambda)^{-1} \begin{bmatrix} \xi_1 \\ \vdots \\ \xi_N \end{bmatrix} = \sum_{j=1}^{N} \sum_{i=0}^{\iota(j)} \frac{\xi_{j+i} \overline{\eta}_j}{(\zeta - \lambda_j)^{i+1}} ,$$

where $\iota(j)$ is equal to the difference of j and the column index of the first column that belongs in $D\lambda$ to the same Jordan subblock as column j. This formula allows us to understand the exact order of F as a function of the components ξ_j and η_j of x_0 and y_0. In particular, the maximum order, which is equal to the degree of the minimum polynomial of A, is attained if and only if for every distinct eigenvalue the sum $\sum \xi_j \overline{\eta}_j$ taken over those indices j that belong to the principal vectors of maximum grade (for that eigenvalue) does not vanish. (Maximum is to be understood as maximum with respect to all principal vectors corresponding to that particular eigenvalue. If there is just one Jordan subblock corresponding to that eigenvalue, or just one of maximum size, the sum reduces to one term, and hence the corresponding components ξ_j and η_j must not vanish.)

At best, the spectral information that can be gained from the Lanczos process is the one in the minimum polynomial of A. Geometric multiplicity of eigenvalues gets lost, which is a great advantage when a linear system has to be solved: Only the distinct eigenvalues matter then and determine the maximum number of iteration steps. In contrast, algebraic multiplicity of an eigenvalue with just one eigenvector (i.e., a nonlinear divisor) does not lead to a reduction of the maximum number of iteration steps, a fact that may actually also be an advantage, since the "pseudo-spectrum" corresponding to such an eigenvalue is large [68], and therefore a multiple zero in the residual polynomial at (or near) this eigenvalue is useful.

On the other hand, as already noted by Lanczos [38, p. 259], an unfortunate choice of the initial vectors x_0 and y_0 may cause to vanish some of the relevant components ξ_j or η_j, or at least the above mentioned sum over the terms of maximum grade. This can be an advantage since the maximum Krylov space generated from A and x_0 may have a smaller dimension, but it may also cause serious problems when this dimension is not as much reduced as the order of the function F is.

The nongeneric algorithms developped in [25] are able to jump from one block in the Padé table to the next and to reach the infinite block, where the Padé approximant equals F. There are theoretically no breakdowns before the order of F is attained, but, of course, information not contained in F cannot be found. Therefore, unfortunate choice of the initial vectors may still lead to a termination without having found the solution, but the chance for this is greatly reduced. Moreover, after all, it is stability that counts, and in this respect jumping from one block well-defined block to the next one is an advantage.

We took it so far for granted that the recurrence coefficients in (4.3)–(4.5) are generated by a Lanczos process. This need not be so. For example, they can now be expressed in terms of the functionals Φ and Φ_1:

(4.41a)
$$\varphi_n = \Phi(\zeta \sigma_n^2)/\delta_n = \Phi_1(\sigma_n^2)/\delta_n ,$$

(4.41b)
$$\delta_{n+1} = \Phi(\rho_{n+1}^2) ,$$

(4.41c)
$$\psi_{n+1} = \gamma_n \delta_{n+1}/(\delta_n \varphi_n) ;$$

(4.42a)
$$\alpha_n = \Phi(\zeta \rho_n^2)/\delta_n = \Phi_1(\rho_n^2)/\delta_n ,$$

$$\beta_n = \Phi(\zeta\rho_{n-1}\rho_n)/\delta_{n-1} = \Phi_1(\rho_{n-1}\rho_n)/\delta_{n-1} = \gamma_{n-1}\delta_n/\delta_{n-1} \quad (\text{if } n > 0)\,, \tag{4.42b}$$

$$\delta_{n+1} = \Phi(\rho_{n+1}^2)\,; \tag{4.42c}$$

$$\alpha'_n = \Phi_1(\zeta\sigma_n^2)/\delta'_n = \Phi_2(\sigma_n^2)/\delta'_n\,, \tag{4.43a}$$

$$\beta'_n = \Phi_1(\sigma_n^2)/\delta'_{n-1} = \Phi_2(\sigma_{n-1}\sigma_n)/\delta'_{n-1} = \gamma'_{n-1}\delta'_n/\delta'_{n-1} \quad (\text{if } n > 0)\,, \tag{4.43b}$$

$$\delta'_{n+1} = \Phi_1(\sigma_{n+1}^2)\,. \tag{4.43c}$$

Φ_2 appearing in (4.43a) and (4.43b) is in analogy to Φ_1 defined as the linear functional on $\mathcal{P}$ taking the values $\Phi_2(\zeta^k) := \mu_{k+2}$ $(k \in \mathbf{N})$.

There are many ways to rewrite some of these formulas. In particular, due to the orthogonality properties of the polynomials a factor $\varphi_n, \sigma_n, \varphi_{n-1}$, or σ_{n-1} can often be replaced by ζ^n/Γ_n or ζ^{n-1}/Γ_{n-1} respectively. A less trivial example for an equivalent formulation is the following. Starting from (4.42a) and using (4.4) and orthogonality we get:

$$\alpha_n\delta_n\Gamma_n = \Phi(\alpha_n\rho_n\zeta^n) = \Phi(\zeta^n[\zeta\rho_n - \gamma_n\rho_{n+1} - \beta\rho_{n-1}]) = \Phi(\zeta^{n+1}\rho_n) - \beta_n\Phi(\zeta^n\rho_{n-1}).$$

Hence

$$\alpha_n = \frac{\Phi(\zeta^{n+1}\rho_n)}{\delta_n\Gamma_n} - \frac{\Phi(\zeta^n\rho_{n-1})}{\delta_{n-1}\Gamma_n}. \tag{4.44}$$

If ρ_n is given in the algebraic form (4.16), if the definition of Φ is applied, and if $\delta_n^* := \delta_n\Gamma_n$ is introduced, the formulas (4.42) can thus be replaced by

$$\delta_n^* := \Phi(\zeta^n\rho_n) = \sum_{k=0}^{n} \rho_k^{(n)}\mu_{k+n}\,, \tag{4.45a}$$

$$\varepsilon_n := \Phi(\zeta^{n+1}\rho_n) = \sum_{k=0}^{n} \rho_k^{(n)}\mu_{k+n+1}\,, \tag{4.45b}$$

$$\alpha_n = \varepsilon_n\delta_n^* - \varepsilon_{n-1}/(\gamma_{n-1}\delta_{n-1}^*)\,, \tag{4.45c}$$

$$\beta_n = \delta_n^*/\delta_{n-1}^*\,. \tag{4.45d}$$

Likewise by noting that

$$\varphi_n\delta_n\Gamma_n = \Phi_1(\zeta^n\sigma_n)\,,$$

$$\psi_{n+1} = \gamma_n\delta_{n+1}/(\delta_n\varphi_n) = \delta_{n+1}\Gamma_{n+1}/(\delta_n\Gamma_n\varphi_n)\,,$$

the formulas (4.41) can be replaced by

$$\varepsilon'_n := \Phi_1(\zeta^n\sigma_n) = \sum_{k=0}^{n} \sigma_k^{(n)}\mu_{k+n+1}\,, \tag{4.46a}$$

$$\text{(4.46}b) \qquad \delta^*_{n+1} := \Phi(\zeta^{n+1}\rho_{n+1}) = \sum_{k=0}^{n} \rho_k^{(n+1)} \mu_{k+n+1}\,,$$

$$\text{(4.46}c) \qquad \varphi_n = \varepsilon'_n / \delta^*_n$$

$$\text{(4.46}d) \qquad \psi_{n+1} = \delta^*_{n+1}/\varepsilon'_n\,.$$

It is worth noting that the formulas (4.46) have already been given by Lanczos [38, p. 262].

In (4.24) we have also seen that, theoretically, LDU decomposition of the moment matrix and inversion of P_ν^{-1} allows one to compute formal orthogonal polynomials up to the moment where for the first time one such polynomial is orthogonal to itself. This is, however, a very inefficient method to compute these polynomials. In fact, it seems more natural to find directly the recurrence coefficients $\alpha_n, \beta_n, \gamma_n$. If the moments are defined by (4.11) and have not yet been computed, then the BO algorithm serves exactly this purpose: It generates these recurrence coefficients from A, x_0 and y_0, without making explicit use of the moments. If, however, the moments are prescribed, one could apply the above formulas (4.45). Another, though not recommeded method, is the qd algorithm, cf. Sect. 6. A third, and the oldest approach is the *Chebyshev algorithm*, which we can derive in a few lines: We define the $N \times \infty$ matrix of the Krylov vectors generated by A^H from y_0,

$$\text{(4.47)} \qquad \hat{Y} := [y_0, A^H y_0 (A^H)^2 y_0, \ldots]\,,$$

and the $\infty \times \nu$ lower triangular matrix

$$\text{(4.48)} \qquad \Sigma := \hat{Y}^H B X\,,$$

whose (m,n)-element is $< A^m y_0, x_n >_B$. If S denotes now the $\infty \times \infty$ shift matrix, one has $A^H \hat{Y} = \hat{Y} S$, and therefore, using (2.8) and $BA = AB$, and assuming $x_\nu = 0$,

$$\text{(4.49)} \qquad \Sigma T = \hat{Y}^H B X T = \hat{Y} \hat{Y}^H B A Y = \hat{Y}^H A B X = S^H \hat{Y}^H B X = S^T \Sigma.$$

Equating here the (m,n)-element yields, if $\Sigma =: [\sigma_{m,n}]$ and $\gamma_{\nu-1} := 0$ (which is permitted since $x_\nu = 0$):

$$\text{(4.50)} \qquad \gamma_n \sigma_{m,n+1} + \alpha_n \sigma_{m,n} + \beta_n \sigma_{m,n-1} = \sigma_{m+1,n} \quad (m \geq 0,\ 0 \leq n \leq \nu - 1)\,,$$

which can be solved for $\sigma_{m,n+1}$ since $\gamma_n \neq 0$:

$$\text{(4.51)} \qquad \sigma_{m,n+1} := (\sigma_{m+1,n} - \alpha_n \sigma_{m,n} - \beta_n \sigma_{m,n-1})/\gamma_n \quad (m \geq 0,\ 0 \leq n \leq \nu - 1)\,.$$

Hence, if T is known, the matrix Σ can be built up from left to right, starting form the first column, which contains the moments $\sigma_{m,0} = \mu_m$. However, amazingly, one can do this even when T is unknown, since the elements α_n and β_n of T can be computed while we are building up Σ. (γ_n can be chosen freely.) For $m < n - 1$ both sides of (4.50) are zero, but $m = n - 1$ and $m = n$ yield

$$\beta_n \sigma_{n-1,n-1} = \sigma_{n,n} \quad (1 \leq n \leq \nu - 1),$$

$$\alpha_n \sigma_{n,n} + \beta_n \sigma_{n,n-1} = \sigma_{n+1,n} \quad (0 \leq n \leq \nu - 1).$$

By solving for α_n and β_n one obtains

$$\beta_n := \frac{\sigma_{n,n}}{\sigma_{n-1,n-1}} \quad (n = 1, \ldots, \nu - 1), \tag{4.52a}$$

$$\alpha_n := \frac{\sigma_{n+1,n}}{\sigma_{n,n}} - \frac{\sigma_{n,n-1}}{\sigma_{n-1,n-1}} \quad (n = 0, 1, \ldots, \nu - 1). \tag{4.52b}$$

Hence, α_n and β_n can be determined from elements in rows n and $n-1$ of Σ, actually those on the diagonal and subdiagonal, but in (4.50) α_n and β_n get only used for producing the next row (row $n+1$) of Σ. (4.51) and (4.52) make up the Chebyshev algorithm.

Unfortunately, the problem of computing the recurrence coefficients α_n and β_n is in most cases ill-conditioned, so that the numerical effects cause stability problems. This difficulty gave rise to using *modified moments* instead of ordinary moments. These modified moments are then used as the initial data of the *modified Chebyshev algorithm* for computing $\alpha_0, \beta_1, \alpha_1, \ldots, \alpha_{\nu-1}, \beta_{\nu-1}$. See, e.g., [18] and [19] for results and references on this subject.

5. The relation to continued fractions. A continued fraction is a finite or infinite construction of the form[4]

$$b_0 + \cfrac{a_1}{b_1 + \frac{a_2}{b_2 + \frac{a_3}{b_3 + \cdots}}}$$

often written as

$$b_0 + \mathop{\mathrm{K}}_{j=1}^{J} \frac{a_j}{b_j} \qquad \text{or} \qquad b_0 + \frac{a_1 |}{| b_1} + \frac{a_2 |}{| b_2} + \frac{a_3 |}{| b_3} + \cdots, \tag{5.1}$$

where $J \leq \infty$. In our application a_j and b_j are real or complex numbers or polynomials of degree 1. They are called the jth *partial numerator* and the jth *partial denominator*, respectively. One also says that a_j and b_j are the *elements* of (5.1). The jth "partial sum", which includes all terms up to a_j/b_j is called the jth *convergent* (or *approximant*). The following basic result, which is often called "Euler recurrence formula", is due to Wallin:

LEMMA 5.1. *Define u_j and v_j by the recurrences*

$$u_{-1} := 1, \quad u_0 := b_0, \quad u_j := b_j u_{j-1} + a_j u_{j-2}, \ j = 1, 2, \ldots\ , \tag{5.2a}$$

$$v_{-1} := 0, \quad v_0 := 1, \quad v_j := b_j v_{j-1} + a_j v_{j-2}, \ j = 1, 2, \ldots\ . \tag{5.2b}$$

Then

$$w_j := b_0 + \mathop{\mathrm{K}}_{k=1}^{j} \frac{a_k}{b_k} = \frac{u_j}{v_j}\,. \tag{5.3}$$

u_j and v_j are called the jth *numerator* and the jth *denominator* of the continued fraction.[5]

Proof. The proof is by induction. For $j = 0$ and $j = 1$ the equality in (5.3) is clearly correct. From the definition of w_j it is seen that w_{j+1} is obtained by replacing b_j by $b_j + a_{j+1}/b_{j+1}$, provided that $b_{j+1} \neq 0$. Hence, if the lemma holds up to j, one has

$$\begin{aligned} w_{j+1} &= \frac{(b_j + a_{j+1}/b_{j+1}) u_{j-1} + a_j u_{j-2}}{(b_j + a_{j+1}/b_{j+1}) v_{j-1} + a_j v_{j-2}} \\ &= \frac{u_j + (a_{j+1}/b_{j+1}) u_{j-1}}{u_j + (a_{j+1}/b_{j+1}) u_{j-1}} = \frac{u_{j+1}}{v_{j+1}}\,. \end{aligned}$$

[4] In this section roman letters are also denoting scalar quantities.

[5] u_j and v_j should not be confused with the vectors u_n and v_n that appear in the other sections.

If $b_{j+1} = 0$ and $a_{j+1} \neq 0$, then $a_{j+1}/b_{j+1} = \infty$; i.e., w_{j+1} is obtained by replacing in (5.1) a_j by 0, so that $w_{j+1} = w_{j-1}$. The recurrences (5.2) yield in fact $u_{j+1} = a_{j+1}u_{j-1}$, $v_{j+1} = a_{j+1}v_{j-1}$, hence $u_{j+1}/v_{j+1} = u_{j-1}/v_{j-1} = w_{j-1}/w_{j+1}$. □

Two continued fraction are called *equivalent* if they have the same convergents [35, p. 31]. One can show [35, Theorem 2.6, p. 31]:

LEMMA 5.2. *The two continued fractions with elements b_0, a_1, b_1, $a_2, \ldots, a_J$, b_J and $\tilde{b}_0$, $\tilde{a}_1$, $\tilde{b}_1$, $\tilde{a}_2, \ldots, \tilde{a}_J$, $\tilde{b}_J$ are equivalent if and only if there exists a sequence $\{e_j\}_{j=0}^{J}$ with $e_0 := 1$, $e_j \neq 0$ $(\forall j)$ such that*

$$\tilde{a}_j = e_j e_{j-1} a_j \quad (j \geq 1), \qquad \tilde{b}_j = e_j b_j \quad (j \geq 0). \tag{5.4}$$

The numerators and denominators u_j, v_j and $\tilde{u}_j, \tilde{v}_j$ defined by (5.2) are then related by

$$\tilde{u}_j = e_0 \cdots e_j u_j \,, \qquad \tilde{v}_j = e_0 \cdots e_j v_j \,. \tag{5.5}$$

A formal power series (FPS) in ζ^{-1} like the one of F in (4.25) can be expanded in several ways into a continued fraction. For example, we may write

$$F(\zeta) = \sum_{k=0}^{\infty} \frac{\mu_k}{\zeta^{k+1}} = 0 + \frac{\mu_0/\zeta}{1 + E_1(\zeta)}, \qquad E_1(\zeta) = \frac{\mu_0^{(1)}/\zeta}{1 + E_2(\zeta)},$$

etc., where $1 + E_1$ is the reciprocal of the FPS of $(F(\zeta) - 0)\zeta/\mu_0 = 1 + O(\zeta^{-1}) =: (1 + \mu_0^{(1)}\zeta^{-1} + \ldots)^{-1}$, and $1 + E_2$ is the reciprocal of the FPS of $E_1(\zeta)\zeta/\mu_0^{(1)}$. The process carries on as long as the generated coefficient $\mu_0^{(k)}$ of ζ^{-1} in the FPS for E_k does not vanish.

If one could proceed ad infinitum, one would obtain a regular C-fraction in ζ^{-1} [35, p. 128]:

$$F(\zeta) = \frac{\mu_0/\zeta|}{|1} + \frac{\mu_0^{(1)}/\zeta|}{|1} + \frac{\mu_0^{(2)}/\zeta|}{|1} + \frac{\mu_0^{(3)}/\zeta|}{|1} + \cdots ,$$

which after an equivalence transform and after setting $a_k := \mu_0^{(k-1)}$ becomes

$$F(\zeta) = \frac{a_1|}{|\zeta} + \frac{a_2|}{|1} + \frac{a_3|}{|\zeta} + \frac{a_4|}{|1} + \cdots . \tag{5.6}$$

In our application the process will be seen to break down. Depending on whether this happens after, say, $2\dot{\nu}$ or $2\dot{\nu} + 1$ steps one obtains either

$$F(\zeta) = \frac{a_1|}{|\zeta} + \frac{a_2|}{|1} + \frac{a_3|}{|\zeta} + \frac{a_4|}{|1} + \cdots + \frac{a_{2\dot{\nu}-1}|}{|\zeta} + \frac{a_{2\dot{\nu}}|}{|1} + \frac{E_{2\dot{\nu}}(\zeta)|}{|1} \tag{5.7a}$$

or

$$F(\zeta) = \frac{a_1|}{|\zeta} + \frac{a_2|}{|1} + \frac{a_3|}{|\zeta} + \frac{a_4|}{|1} + \cdots + \frac{a_{2\dot{\nu}}|}{|1} + \frac{a_{2\dot{\nu}+1}|}{|\zeta} + \frac{\zeta E_{2\dot{\nu}+1}(\zeta)|}{|1}. \tag{5.7b}$$

If $E_{2\dot{\nu}} = 0$ or $E_{2\dot{\nu}+1} = 0$, respectively, one has a "finite C-fraction", but of course this is not the only possible cause of breakdown: $E_{2\dot{\nu}}$ or $E_{2\dot{\nu}+1}$ can be any FPS in ζ^{-1} with a vanishing linear term. The index $\dot{\nu}$ will turn out to be our breakdown index of BIOMIN, but at the moment we think of it just as the index where the construction of the continued fraction breaks down.

In order to determine the partial numerators a_j of this continued fraction in the case of our interest, where F is given by (4.25), we have first to recall yet another basic result from continued fraction theory, namely the connection with Padé approximation [35, p. 190]. It is a consequence of the following standard lemma:

LEMMA 5.3. *For the continued fraction (5.1) holds*

$$w_{j+1} - w_j = \frac{a_{j+1}}{v_j} \, \frac{u_{j-1}v_j - v_{j-1}u_j}{b_{j+1}v_j - a_{j+1}v_{j-1}} \tag{5.8}$$

and

$$u_{j-1}v_j - v_{j-1}u_j = (-1)^n \prod_{j=1}^{n} a_j \,. \tag{5.9}$$

Proof. Express $w_{j+1} = u_{j+1}/v_{j+1}$ according to the recurrence formulas (5.2). □

THEOREM 5.4. *The numerators u_j and the denominators v_j of the continued fraction in (5.7a) [and (5.7b)] (which we assume to have nonvanishing partial numerators a_j) are polynomials in ζ, hence, the convergents $w_j := u_j/v_j$ are rational functions. For $j = 0, 1, \ldots, 2\dot{\nu}$ [$2\dot{\nu} + 1$] and $2n = j$ or $2n + 1 = j$ holds:*

(i) $\partial u_{2n} = n - 1$, $\partial u_{2n+1} = n$, $\partial v_{2n} = n$, $\partial v_{2n+1} = n + 1$;

(ii) v_j is monic;

(iii) $v_{2n+1}(0) = 0$;

(iv) $w_{j+1} - w_j = O(\zeta^{-j-1})$;

(v) w_{2n} and w_{2n+1} are the (n, n) and $(n+1, n)$ Padé approximants at $\zeta = \infty$ of the formal power series F in ζ^{-1} (with vanishing constant term) used for constructing the continued fraction.

Proof. For the continued fractions in (5.7) one has $b_0 = 0$, $b_{2n} = 1$ and $b_{2n+1} = \zeta$. Inserting these quantities into the recurrences (5.2) and using induction we get the statements (i)–(iii). Inserting them into (5.8) and (5.9) shows that the difference in (5.9) is actually a constant, and gives (iv), which is valid even when $j + 1 = 2\dot{\nu}$ [$j + 1 = 2\dot{\nu} + 1$]. But (iv) implies that w_j determines the terms up to ζ^{-j} of a formal power series $\tilde{F}(\zeta)$ in ζ^{-1}, so that $\tilde{F}(\zeta) - w_j = O(\zeta^{-j-1})$. Therefore, in view of (i),

$$\tilde{F}(\zeta)v_{2n}(\zeta) - u_{2n}(\zeta) = O(\zeta^{-n-1}) \,,$$

$$\tilde{F}(\zeta)v_{2n+1}(\zeta) - u_{2n+1}(\zeta) = O(\zeta^{-n-2}) \,,$$

which, by definition, shows that the statement (v) is true, cf. (4.26) and (4.27). From w_1 we see that $\tilde{F}(\infty) = 0$. Moreover, by construction, $w_{2\dot{\nu}+1} = F$ [or $w_{2\dot{\nu}+2} = F$, respectively], hence, one may assume that $\tilde{F} = F$. □

We have already seen in (4.31) that the (n, n) Padé approximant of F is π_n/ρ_n and that the $(n + 1, n)$ Padé approximant is $\tau_n/(\zeta\sigma_n)$. Since the mixed recurrence relations (4.3) for the denominators of these approximants must be equivalent to the recurrence (5.2b), it is now an easy matter to read off the values of the partial numerators a_j. The monic denominator polynomials are

$$v_{2n}(\zeta) = \rho_n(\zeta)\Gamma_n, \quad v_{2n+1}(\zeta) = \zeta\sigma_n(\zeta)\Gamma_n \,, \tag{5.10}$$

and thus the comparison of (4.3) and (5.2b) yields

$$\begin{array}{ll} a_{2n} = -\varphi_{n-1} \;\; (n \geq 1), & b_{2n} = 1 \;\; (n \geq 1) \\ a_{2n+1} = -\gamma_{n-1}\psi_n \;\; (n \geq 1), & b_{2n+1} = \zeta \;\; (n \geq 0). \end{array} \tag{5.11}$$

a_1 does not follow from this comparison, but by the definition of F clearly $a_1 = \mu_0$.

From our derivation it should be clear that these connections are valid up to the index where the BOBC algorithm breaks down. If the breakdown is due to $\delta_{\dot\nu} = 0$ or $\varphi_{\dot\nu} = 0$, then (5.7a) or (5.7b), respectively, holds. Conversely, if these continued fraction expansions are valid, the denominators of the convergents are formal orthogonal polynomials, and according to (4.1) and (4.2) one can associate to them vectors satisfying (2.1) and (3.2)–(3.4); hence, the BOBC algorithm does not break down earlier than for $n + 1 = \dot\nu$.

Summarizing we have derived the following first main result of this section:

THEOREM 5.5. *Depending on whether the BOBC method terminates with* $\delta_{\dot\nu} = 0$ *(i.e.,* $\psi_{\dot\nu} = 0$*) or* $\varphi_{\dot\nu} = 0$, *the function* F *of (4.25) can be expanded into a continued fraction*

$$(5.12a)\quad F(\zeta) = \frac{\mu_0\rfloor}{\lceil \zeta} - \frac{\varphi_0\rfloor}{\lceil 1} - \frac{\gamma_0\psi_1\rfloor}{\lceil \zeta} - \frac{\varphi_1\rfloor}{\lceil 1} - \frac{\gamma_1\psi_2\rfloor}{\lceil \zeta} - \cdots - \frac{\gamma_{\dot\nu-2}\psi_{\dot\nu-1}\rfloor}{\lceil \zeta} - \frac{\varphi_{\dot\nu-1}\rfloor}{\lceil 1} + \frac{E_{2\dot\nu}(\zeta)\rfloor}{\lceil 1},$$

or

$$(5.12b)\quad F(\zeta) = \frac{\mu_0\rfloor}{\lceil \zeta} - \frac{\varphi_0\rfloor}{\lceil 1} - \frac{\gamma_0\psi_1\rfloor}{\lceil \zeta} - \frac{\varphi_1\rfloor}{\lceil 1} - \frac{\gamma_1\psi_2\rfloor}{\lceil \zeta} - \cdots - \frac{\varphi_{\dot\nu-1}\rfloor}{\lceil 1} - \frac{\gamma_{\dot\nu-1}\psi_{\dot\nu}\rfloor}{\lceil \zeta} + \frac{\zeta E_{2\dot\nu+1}(\zeta)\rfloor}{\lceil 1}$$

respectively, where the parameters φ_n, γ_n, *and* ψ_n *are those of the BOBC algorithm, and* $E_{2\dot\nu}$ *and* $E_{2\dot\nu+1}$ *are formal power series in* ζ^{-1} *with vanishing constant and linear terms. (Thus the continued fractions cannot be expanded further keeping the same structure.)*

Conversely, if F *can be expanded according to (5.12a) or (5.12b), but no further, the BOBC algorithm breaks down with* $\delta_{\dot\nu} = 0$ *or* $\varphi_{\dot\nu} = 0$, *respectively.*

The numerators u_j *and the monic denominators* v_j *of (5.12a) and (5.12b) are, for* $j = (0), 1, \dots, 2\dot\nu - 1$ *and* $j = (0), 1, \dots, 2\dot\nu$, *respectively, and* $j = 2n$ *or* $j = 2n + 1$,

$$(5.13)\qquad \begin{aligned} u_{2n}(\zeta) &= \Gamma_n \pi_n(\zeta), & v_{2n}(\zeta) &= \Gamma_n \rho_n(\zeta), \\ u_{2n+1}(\zeta) &= \Gamma_n \tau_n(\zeta), & v_{2n+1}(\zeta) &= \Gamma_n \zeta \sigma_n(\zeta). \end{aligned}$$

They are related through (4.1) and (4.2) to the vectors generated by the BOBC algorithm. The convergents

$$(5.14)\qquad w_{2n}(\zeta) = \frac{\pi_n(\zeta)}{\rho_n(\zeta)}, \quad w_{2n+1}(\zeta) = \frac{\tau_n(\zeta)}{\zeta\sigma_n(\zeta)}$$

are the (n,n) and the (n+1,n) Padé approximants of F *at* ∞. *(Their types as rational functions of* ζ *are* $(n-1, n)$ *and* $(n, n+1)$, *respectively.)*

The *odd and even parts* of a continued fraction (5.1) with convergents w_n are continued fractions whose convergents are $w_1, w_3, \dots$ and $w_2, w_4, \dots$, respectively. They are of course only determined up to an equivalence transform, but we are interested here in those where the numerators and the denominators are conserved and not only the quotients (convergents). (This is not true for the standard formulas given in [35].) The operation of forming the odd or even part is analogous to one step of *cyclic reduction* (or *odd-even reduction*) which is commonly used to vectorize tridiagonal linear system or three-term recurrence relation [32]. Cyclic reduction is based on first reordering simultaneously the unknowns and the equations in order to separate odd-indexed from even-indexed ones, and then eliminating either the even-indexed or the odd-indexed at once by a block Gauss elimination step. The resulting linear system, whose matrix is a Schur complement, is again tridiagonal. The process can therefore be iterated, but at the moment we are only interested in the first step.

For the continued fraction (5.12b) the three-term recurrence (5.2b) for the denominators is equivalent to (4.3), where the even and odd 'unknowns' ρ_n and $\zeta\sigma_n$, respectively, have already separate names. Using the matrices L and R of (3.8) this recurrence (4.3) can in analogy to (3.9) and (3.10) be expressed as

$$\zeta[\sigma_0, \sigma_1, \dots, \sigma_{\dot\nu-1}] - [\rho_0, \rho_1, \dots, \rho_{\dot\nu-1}]\, L = [0, \dots, 0, \rho_{\dot\nu}\gamma_{\dot\nu-1}]\,, \tag{5.15a}$$

$$[\rho_0, \rho_1, \dots, \rho_{\dot\nu-1}] - [\sigma_0, \sigma_1, \dots, \sigma_{\dot\nu-1}]\, R = [0, \dots, 0, 0]\,. \tag{5.15b}$$

Elimination of the odd-indexed convergents, i.e., of the denominators $\zeta\sigma_n$ yields, with $T := LR$,

$$[\rho_0, \rho_1, \dots, \rho_{\dot\nu-1}](\zeta I - T) = [0, \dots, 0, \rho_{\dot\nu}\gamma_{\dot\nu-1}]\,, \tag{5.16}$$

and likewise, elimination of the denominators ρ_n gives, with $T_1 := RL + e_{\dot\nu}\gamma_{\dot\nu-1}\psi_{\dot\nu}e_{\dot\nu}^T$,

$$[\sigma_0, \sigma_1, \dots, \sigma_{\dot\nu-1}](\zeta I - T) = [0, \dots, 0, \sigma_{\dot\nu}\gamma_{\dot\nu-1}]\,. \tag{5.17}$$

These two relations are of course just (4.4) and (4.5) rewritten as vector-matrix equations, and the even-odd reduction applied is exactly equivalent to the elimination of the matrices U and X from (3.9) and (3.10), which lead to (3.12) and (3.13), respectively. Therefore (5.16) and (5.17) may actually hold beyond $\dot\nu$, namely until ν and ν', respectively.

Finally, we have to recall that the monic denominators of the continued fraction (5.12b) are not ρ_n and $\zeta\sigma_n$, but $v_{2n} = \rho_n\Gamma_n$ and $v_{2n+1} = \zeta\sigma_n\Gamma_n$. In order to get the recurrences for these even and odd denominators, a diagonal similarity transformation $D_\Gamma := \mathrm{diag}[1, \Gamma_1, \Gamma_2, \dots, \Gamma_{\dot\nu-1}]$ has to be applied to the tridiagonal matrices $\zeta I - T$ and $\zeta I - T_1$ of (5.16) and (5.17), i.e., the recurrence coefficients are the elements of

$$D_\Gamma^{-1}(\zeta I - T)D_\Gamma \quad \text{and} \quad D_\Gamma^{-1}(\zeta I - T_1)D_\Gamma \tag{5.18}$$

namely, in the nth recurrence (column)

$$a_n := -\beta_n\gamma_{n-1} \;\; (n \ge 1), \quad b_n := \zeta - \alpha_n \;\; (n \ge 0) \tag{5.19a}$$

and

$$a_n := -\beta_n'\gamma_{n-1} \;\; (n \ge 1), \quad b_n := \zeta - \alpha_n' \;\; (n \ge 0), \tag{5.19b}$$

respectively. By equating the first convergent of the two new continued fractions to the first and second convergent of (5.12b) one concludes moreover that

$$b_0 := 0, \quad a_1 := \mu_0 \tag{5.19c}$$

and

$$b_0 := \mu_0/\zeta, \quad a_1 := \mu_1/\zeta\,, \tag{5.19d}$$

respectively.

(5.19d) follows also from (5.19c) and the following lemma, which will also be implicitly used in Sect. 6, and which follows readily from the definition of the Padé approximant, so that its verification is left to the reader.

Lemma 5.6. *Let $m > n$. Then $\zeta^{-1}\pi^*(\zeta^{-1})/\sigma^*(\zeta^{-1})$ is the (n,n) Padé approximant of*

$$F_{m-n}(\zeta) := \zeta^{m-n}[F(\zeta) - \sum_{k=0}^{m-n-1} \frac{\mu_k}{\zeta^{k+1}}], \tag{5.20}$$

at ∞ if and only if

$$\sum_{k=0}^{m-n-1} \frac{\mu_k}{\zeta^{k+1}} + \zeta^{n-m-1}\frac{\pi^*(\zeta^{-1})}{\sigma^*(\zeta^{-1})} \tag{5.21}$$

is the (m,n) Padé approximant of F at ∞. F_{m-n} is again a power series in ζ^{-1} with vanishing constant coefficient.

According to this lemma, if we write

$$F(\zeta) = \frac{\mu_0}{\zeta} + \frac{1}{\zeta}F_1(\zeta)\,, \tag{5.22}$$

then the $(n+1,n)$ Padé approximants of F_1 have the same denominators as the (n,n) Padé approximant of F, and hence F_1 can be expanded into a continued fraction of the same form as the one of F. Inserting this continued fraction into (5.22) yields one for F, which must be identical to the one with elements (5.19b) and (5.19d).

Concerning the connection between the extendability of these continued fractions and the breakdowns of the BO algorithm and the BC algorithm one can draw analogous conclusions as in Theorem 5.5. Summarizing we get:

Theorem 5.7. *(i) The function F of (4.25) can be expanded into a continued fraction*

$$F(\zeta) = \frac{\mu_0\;|}{|\;\zeta-\alpha_0} - \frac{\beta_1\gamma_0\;|}{|\;\zeta-\alpha_1} - \frac{\beta_2\gamma_1\;|}{|\;\zeta-\alpha_2} - \cdots - \frac{\beta_{\nu-1}\gamma_{\nu-2}\;|}{|\;\zeta-\alpha_{\nu-1}} - \frac{R_\nu(\zeta)\;|}{|\;1}, \tag{5.23}$$

where the parameters α_n, β_n, and γ_n are those of the BO algorithm, ν is the breakdown index of the BO algorithm (i.e., $\delta_\nu = 0$), and $R_\nu(\zeta)$ is a FPS in ζ^{-1} with $R_\nu(\zeta) = O(\zeta^{-2})$. (Thus, (5.23) cannot be expanded further.) Conversely, if F can be expanded in this way, but no further, the BO algorithm breaks down with $\delta_\nu = 0$. For $n = (0), 1, 2, \ldots, \nu$ the numerators and the monic denominators of (5.23) are $\Gamma_n\pi_n(\zeta)$ and $\Gamma_n\rho_n(\zeta)$, respectively, and the convergents $\pi_n(\zeta)/\rho_n(\zeta)$ are the (n,n) Padé approximants of F at ∞.

The even parts of the continued fractions (5.12a) and (5.12b) are equal to the continued fraction (5.23) if in the case $\nu > \dot{\nu}$ the $\nu - \dot{\nu} + 1$ last terms of (5.23) are compressed to yield the tail $R_{\dot{\nu}}(\zeta)$. The elements of (5.12) are related to those of the shortened fraction (5.23) according to (3.32).

(ii) The function F_1 in (5.22) (and thus F itself) can be expanded into a continued fraction

$$F_1(\zeta) = \frac{\mu_1\;|}{|\;\zeta-\alpha'_0} - \frac{\beta'_1\gamma'_0\;|}{|\;\zeta-\alpha'_1} - \frac{\beta'_2\gamma'_1\;|}{|\;\zeta-\alpha'_2} - \cdots - \frac{\beta'_{\nu'-1}\gamma'_{\nu'-2}\;|}{|\;\zeta-\alpha'_{\nu'-1}} - \frac{R^1_{\nu'}(\zeta)\;|}{|\;1}, \tag{5.24}$$

where the parameters α'_n, β'_n, and γ'_n are those of the BC algorithm, ν' is the breakdown index of the BC algorithm (i.e., $\delta'_{\nu'} = 0$), and $R^1_{\nu'}(\zeta)$ is a FPS in ζ^{-1} with $R^1_{\nu'}(\zeta) = O(\zeta^{-2})$. (Thus, (5.24) cannot be expanded further.) Conversely, if F_1 can be expanded in this way, but no further, the BC algorithm breaks down with $\delta'_{\nu'} = 0$. For $n = (0), 1, 2, \ldots, \nu' - 1$ the numerators and the monic denominators of (5.24) are $\Gamma_n\tau_n(\zeta)$ and $\Gamma_n\zeta\sigma_n(\zeta)$, respectively, and the convergents $\tau_n(\zeta)/(\zeta\sigma_n(\zeta))$ are the $(n+1,n)$ Padé approximants of F at ∞.

The odd part of the continued fraction (5.12b) is equal to the continued fraction (5.24) if in the case $\nu' > \acute{\nu}$ the $\nu' - \acute{\nu} + 1$ last terms of (5.24) are compressed to yield the tail $R^1_{\acute{\nu}}(\zeta)$. The elements of (5.12) are related to those of the shortened fraction (5.24) according to (3.33) if $\gamma'_n = \gamma_n$ $(\forall n)$.

What happens when we apply another odd-even reduction to the continued fractions (5.23) and (5.24), and, in particular, is there any way to carry this process over to the Lanczos algorithms? In case of (5.23) this would mean, that we generate directly the sequences $\{x_{2m}\}$ and $\{y_{2m}\}$. In the context of the linear system solvers this looks like a very intersting feature. However, the costs for going from x_{2m} to x_{2m+2} turn not out to be smaller than before. On the other hand, the resulting method would be a chance to avoid a breakdown due to $x_{2m} = x_{2m+1}$. It is in fact a special case of the more general approach to avoiding breakdowns that lead to the nongeneric algorithms proposed in [25]; there the 'index stepsize' is chosen adaptively, in accordance with the size of the blocks in the Padé table.

6. The qd algorithm. In this section we assume that

$$F(\zeta) := \sum_{k=0}^{\infty} \frac{\mu_k}{\zeta^{k+1}} \tag{6.1}$$

is either any formal power series for which the Padé table has on and below the diagonal distinct entries or that F is a rational function given by (4.25) which has exact denominator degree ν and whose Padé table has the property that all entries that lie in the columns with indices 0 up to $\nu - 1$ on or below the diagonal are distinct. An equivalent requirement is that [22]:

$$\det \begin{bmatrix} \mu_l & \mu_{l+1} & \cdots & \mu_{l+n-1} \\ \mu_{l+1} & \mu_{l+2} & & \mu_{l+n} \\ \vdots & & & \vdots \\ \mu_{l+n-1} & \mu_{l+n} & \cdots & \mu_{l+2n-2} \end{bmatrix} \begin{cases} \neq 0, & l = 0,1,\ldots, \ n = 1,2,\ldots,\nu-1, \\ = 0, & l = 0,1,\ldots, \ n = \nu, \nu+1,\ldots. \end{cases} \tag{6.2}$$

Under this assumption the breakdown indices ν, $\acute{\nu}$, and ν' all coincide. In the first case, where F is given by (6.1), we set $\nu := \infty$, and then (6.2) still holds.

In Sect. 3 we have already seen that the parameters γ_n, φ_n and ψ_n of the BOBC algorithm can be obtained from the paramters α_n, β_n and γ_n of the BO algorithm by LU decomposition of the tridiagonal matrix $T_0 := T$ of (2.11) into the lower bidiagonal matrix $L_0 := L$ and the upper bidiagonal matrix $R_0 := R$ of (3.8). From L_0 and R_0 one then obtains the tridiagonal matrix $T_1 = R_0 L_0$ whose elements α'_n, β'_n and γ_n are the parameters of the BC algorithm with $\gamma'_n := \gamma_n$ $(\forall n)$. (Note that the correction term in (3.11) vanishes since $\psi_{\acute{\nu}} = 0$.) Under the above assumption T_1 has an LU decomposition $T_1 = L_1 R_1$, and one can proceed further and compute the tridiagonal matrix $T_2 = R_1 L_1$, then decompose it into a lower times an upper bidiagonal matrix, etc.:

$$L_l R_l := T_l\,, \quad T_{l+1} := R_l L_l\,, \quad l = 0,1,\ldots. \tag{6.3}$$

By repeating our discussion preceeding the definition of Algorithm 7 (the BC algorithm) one concludes that T_l contains the parameters that are generated when the BO algorithm is applied to A, x_0, and y_0 using the inner product $< v, u >_{BA^l} := v^H B A^l u$.

An important point is that $T_1 = R_0 L_0 = R_0 T_0 R_0^{-1}$, and in general,

$$T_{l+1} = R_l L_l = R_l T_l R_l^{-1} = R_l \cdots R_1 R_0 T_0 R_0^{-1} R_1^{-1} \cdots R_l^{-1}\,, \tag{6.4}$$

whence the tridiagonal matrices T_l that are generated in this way from $T_0 = T$ are all similar to T.

In his *LR algorithm* Rutishauser [53, 54, 55] applied the same idea also to full matrices and, in particular, to banded matrices, since the band structure is preserved. Later the same idea was at the origin of Francis' QR algorithm [16].

In Sect. 5 we have seen that the LR step from T to T_1 can be understood as a step from one continued fraction representation of F to another. Now that the LR algorithm applied to T can be continued ad infinitum, a full sequence of such continued fraction representations exist; the lth continued fraction has as the term b_0 the lth partial sum of the power series of F, so that as l increases the continued fractions "start later and later". This corresponds to the appearance of these partial sums in Lemma 5.6. The condition for all these continued fractions and LU decompositions to exist up to the term (column) $\nu - 1$ is exactly our assumption (6.2). In the following theorem we summarize what we can conclude from Sect. 5 in this situation:

THEOREM 6.1. *Let the function F of (4.25) be given and assume that either F is a proper rational function of exact denominator degree ν and (6.2) holds or that (6.2) is valid with $\nu := \infty$. Then F has for every $l \in \mathbf{N}$ the continued fraction representations*

$$F(\zeta) = \sum_{k=0}^{l-1} \frac{\mu_k}{\zeta^{k+1}} + \frac{\mu_l}{\zeta^l} \cdot \frac{1 \rfloor}{\lceil \zeta - \alpha_0^{(l)}} - \frac{\beta_1^{(l)}\gamma_0 \rfloor}{\lceil \zeta - \alpha_1^{(l)}} - \frac{\beta_2^{(l)}\gamma_1 \rfloor}{\lceil \zeta - \alpha_2^{(l)}} - \cdots \left(- \frac{\beta_{\nu-1}^{(l)}\gamma_{\nu-2} \rfloor}{\lceil \zeta - \alpha_\nu^{(l)}} \right) \tag{6.5}$$

and

$$\begin{aligned} F(\zeta) \;=\; & \sum_{k=0}^{l-1} \frac{\mu_k}{\zeta^{k+1}} + \frac{\mu_l}{\zeta^l} \cdot \frac{1 \rfloor}{\lceil \zeta} - \frac{\varphi_0^{(l)} \rfloor}{\lceil 1} - \frac{\gamma_0 \psi_1^{(l)} \rfloor}{\lceil \zeta} - \frac{\varphi_1^{(l)} \rfloor}{\lceil 1} \\ & - \frac{\gamma_1 \psi_2^{(l)} \rfloor}{\lceil \zeta} - \cdots \left(- \frac{\gamma_{\nu-2}\psi_{\nu-1}^{(l)} \rfloor}{\lceil \zeta} - \frac{\varphi_{\nu-1}^{(l)} \rfloor}{\lceil 1} \right) \end{aligned} \tag{6.6}$$

with freely choosable $\gamma_n \neq 0$, $n = 0, \ldots, \nu - 2$. The continued fraction (6.5) is the even part of the fraction (6.6) with the same l and the odd part of the fraction with the previous value of l.

The nth convergent of (6.5) and the $2n$th convergent of (6.6) are equal to the $(l + n, n)$ Padé approximant of F at ∞ and its representation $\zeta^{-l}\pi_n^{(l)}(\zeta)/\rho_n^{(l)}(\zeta)$ can be computed either by the recurrences

$$\pi_{-1}^{(l)}(\zeta) :\equiv 1, \quad \pi_0^{(l)}(\zeta) := \sum_{k=0}^{l-1} \mu_k \zeta^{l-k-1}, \tag{6.7a}$$

$$\pi_{n+1}^{(l)}(\zeta) := [(\zeta - \alpha_n^{(l)})\pi_n^{(l)}(\zeta) - \beta_n^{(l)}\pi_{n-1}^{(l)}(\zeta)]/\gamma_n\,, \quad n = 0, 1, \ldots, \nu - 1, \tag{6.7b}$$

$$\rho_{-1}^{(l)}(\zeta) :\equiv 0, \quad \rho_0^{(l)}(\zeta) :\equiv 1\,, \tag{6.8a}$$

$$\rho_{n+1}^{(l)}(\zeta) := [(\zeta - \alpha_n^{(l)})\rho_n^{(l)}(\zeta) - \beta_n^{(l)}\rho_{n-1}^{(l)}(\zeta)]\gamma_n\,, \quad n = 0, 1, \ldots, \nu - 1, \tag{6.8b}$$

or by the recurrences

$$\pi_{-1}^{(l+1)}(\zeta) := \pi_{-1}^{(l)}(\zeta) :\equiv 1\,, \tag{6.9a}$$

FIG. 2. *The qd table and the two rhombus rules to generate it.*

$$\pi_{n+1}^{(l)}(\zeta) := [\zeta\pi_n^{(l+1)}(\zeta) - \pi_n^{(l)}(\zeta)\varphi_n^{(l)}]/\gamma_n\,, \tag{6.9b}$$

$$\pi_{n+1}^{(l+1)}(\zeta) := \pi_{n+1}^{(l)}(\zeta) - \pi_n^{(l+1)}(\zeta)]\psi_{n+1}^{(l)}\,, \quad n = 0, 1, \ldots, \nu - 1, \tag{6.9c}$$

$$\rho_{-1}^{(l+1)}(\zeta) := \rho_{-1}^{(l)}(\zeta) :\equiv 0\,, \tag{6.10a}$$

$$\rho_{n+1}^{(l)}(\zeta) := [\zeta\rho_n^{(l+1)}(\zeta) - \rho_n^{(l)}(\zeta)\varphi_n^{(l)}]/\gamma_n\,, \tag{6.10b}$$

$$\rho_{n+1}^{(l+1)}(\zeta) := \rho_{n+1}^{(l)}(\zeta) - \rho_n^{(l+1)}(\zeta)]\psi_{n+1}^{(l)}\,, \quad n = 0, 1, \ldots, \nu - 1, \tag{6.10c}$$

Recurrences (6.9c) and (6.10c) can also be used to generate the polynomials $\pi_n^{(l+1)}$ *and* $\rho_n^{(l+1)}$ $(n = 0, 1, \ldots, \nu - 1)$ *on the* $(l+1)$*th diagonal if those on the* l*th are already known.*

The elements of the continued fractions (6.5) and (6.6) can be generated from those of the first fraction of type (6.5) by applying the LR algorithm (6.3) to the tridiagonal matrix T*: If, in analogy to (2.11), the tridiagonal matrix* T_l *has the elements* $\beta_n^{(l)}$, $\alpha_n^{(l)}$, γ_n, *and if, in analogy to (3.8), the bidiagonal matrices* L_l *and* R_l *have the elements* $\varphi_n^{(l)}$, γ_n *and* $\psi_n^{(l)}$, *1, respectively, then, element by element, this means that after setting* $\psi_0^{(l)} := 0$ $(\forall l)$,

$$\alpha_n^{(l)} = \varphi_n^{(l)} + \gamma_{n-1}\psi_n^{(l)}\,, \quad \beta_{n+1}^{(l)} = \varphi_n^{(l)}\psi_{n+1}^{(l)}\,, \tag{6.11a}$$

$$\alpha_n^{(l+1)} = \varphi_n^{(l)} + \gamma_n\psi_{n+1}^{(l)}\,, \quad \beta_{n+1}^{(l+1)} = \varphi_{n+1}^{(l)}\psi_{n+1}^{(l)}\,, \tag{6.11b}$$

and

$$\varphi_n^{(l)} + \gamma_n\psi_{n+1}^{(l)} = \varphi_n^{(l+1)} + \gamma_{n-1}\psi_n^{(l+1)}\,, \quad n = 0, \ldots, \nu - 1, \ \ l = 0, 1, \ldots\,, \tag{6.12a}$$

$$\varphi_{n+1}^{(l)}\psi_{n+1}^{(l)} = \varphi_n^{(l+1)}\psi_{n+1}^{(l+1)}\,, \quad n = 0, \ldots, \nu - 1, \ \ l = 0, 1, \ldots\,. \tag{6.12b}$$

The two relations (6.12) are Rutishauser's *rombus rules.* Rutishauser assumed $\gamma_n = 1$ $(\forall n)$ and used the notation $q_n^{(l)}$, $e_n^{(l)}$ instead of $\varphi_{n-1}^{(l)}, \psi_n^{(l)}$. According to the above, the rhombus rules can be used to generate the fractions (6.6) for all l from the first one $(l = 0)$, i.e., to proceed diagonal by diagonal downwards in the *qd table*, cf. Fig. 2. This is the *progressive qd algorithm.*

Theoretically the rhombus rules can also be used to build up the qd table from left to right. In the first column of the table, one has

$$\varphi_0^{(l)} = \mu_{l+1}/\mu_l\,, \tag{6.13}$$

since by definition of the continued fraction (6.6)

$$\frac{\mu_l}{\zeta^l}\frac{1}{\zeta-\varphi_0^{(l)}} = \frac{\mu_l}{\zeta^{l+1}} + \frac{\mu_{l+1}}{\zeta^{l+2}} + O(\zeta^{-l-3})\,.$$

Hence, given the moments μ_k $(k = 0, \ldots, 2\nu-1)$ (6.13) can be used to start up the (ordinary) *qd algorithm* that will produce a triangular part of the qd table with $\varphi_0^{(0)}$, $\psi_1^{(0)}$, $\varphi_1^{(0)}, \ldots, \psi_{\nu-1}^{(0)}$, $\varphi_{\nu-1}^{(0)}$ on the upper edge. Unfortunately, in addition to the fact that the problem of producing these coefficients of the recurrences (4.3) from the moments is typically very ill-conditioned, the qd algorithm is often unstable since it requires the Hankel determinants (6.2) corresponding to a whole triangle of the qd table (or the Padé table) to be nonvanishing. (Any near-vanishing of such a Hankel determinant may cause numerical instability.) The qd algorithm is therefore even more prone to breakdown than the *Chebyshev algorithm* briefly mentioned in Sect. 4, which serves the same purpose, and which requires only that the Hankel determinants corresponding to the section on the main diagonal of the Padé table do not vanish.

Rutishauser based the general convergence theory for the qd algorithm on results from Hadamard's thesis [26], which, incidentally, appeared in the same year, 1892, as the Padé thesis [41]. We prefer to cite here the reformulation of the Hadamard result as a convergence theorem for the columns in the Padé table; it was given 10 years after the publication of Hadamard's thesis by de Montessus de Ballore. Algorithmically, the roots of the qd algorithm go back to Bernoulli's method for finding polynomical zeros and on Aitken's work on this algorithm.

THEOREM 6.2. *Let F be meromorphic in $|\zeta| > \kappa$ (including ∞) with poles $\zeta_k \neq \infty$ of multiplicity μ_k $(k = 1, \ldots, K)$ and total multiplicity $n = \sum \mu_k$. Let F be continuous in*

$$\Omega_F := \{\zeta \in \overline{\mathbf{C}}; |\zeta| \geq \kappa,\, \zeta \neq \zeta_k \,(k = 1, \ldots, K)\}\,. \tag{6.14}$$

Then, for $m \to \infty$, the (m, n) Padé approximant of F at ∞ converges on every closed subset of the interior of Ω_F uniformly to F. In particular, the monic denominator of the (m, n) Padé approximant converges to

$$\prod_{k=1}^{K} (\zeta - \zeta_k)^{\mu_k}\,. \tag{6.15}$$

An elegant proof of this theorem can be based on an application of the Hermite error formula, a complex integral formula for the error in polynomial interpolation [60, p.39].

According to the de Montessus theorem, if n is such that—counting multiplicity—exactly n poles of F lie outside a circle (with radius κ) around 0, then the monic denominator of the (m, n) Padé approximant approaches (6.15) as $m \to \infty$. Recall that, if $l := m - n$, this denominator is identical to the monic denominator of the nth convergent of (6.5) or the $2n$th convergent of (6.6). Moreover, the $(2n + 1)$th convergent of (6.6), which is the $(m + 1, n)$ Padé approximant, must approach the same limit, so that necessarily the $2n$th and the $(2n + 1)$th convergent of (6.6) approach each other, which means that $\psi_n^{(l)} \to 0$ as $l \to \infty$. If n' is the next index with the same property and K' the corresponding number of distinct poles, then the quotients of the denominators in the n'th and nth column approach therefore

$$\prod_{k=K+1}^{K'} (\zeta - \zeta_k)^{\mu_k}\,. \tag{6.16}$$

This is related to what Henrici calls *Rutishauser's rule* [30, p. 642], [54, Eq. (17), p. 20], where, however, not the quotients of the n'th and the nth denominators are formed, but a sequence of polynomials of degree $n' - n$ converging to (6.16) is directly computed from the entries between the columns containing $\psi_n^{(l)}$ and $\psi_{n'}^{(l)}$. Rutishauser gave no proof for this rule, but Stewart [66] established it using functional analysis, and, more recently, Seewald [61] gave a relatively short proof. In the simplest case, $K' = K + 1$ and $\mu_k = 1$ $(\forall k)$, so that the product in (6.16) reduces to the one term $\zeta - \zeta_{K+1}$.

The classical theory of the qd algorithm, which is developped in brief style in Rutishauser's original paper [54], is excellently presented in Henrici work [28, 29, 27, 30]. It makes heavy use of quotients of Hankel determinants and is therefore no longer valid in the nongeneric case where some of these determinants vanish. For the nongeneric case the qd algorithm is formulated and treated in [25].

REFERENCES

[1] R. Abbasian, Lanczos algorithms for the acceleration of nonsymmetrizable iterative methods, Tech. Rep. CNA-193, Center for Numerical Analysis, Univ. of Texas at Austin, 1984.

[2] N. I. Akhiezer, *The Classical Moment Problem*, Oliver & Boyd, Edinburgh/London, 1965.

[3] S. F. Ashby, T. A. Manteuffel, and P. E. Saylor, A taxonomy for conjugate gradient methods, SIAM J. Numer. Anal., (1988). (to appear).

[4] O. Axelsson, Conjugate gradient type methods for unsymmetric systems of linear equations, Linear Algebra Appl., 29 (1980), pp. 1–17.

[5] D. L. Boley, D. G. Truhlar, Y. Saad, R. E. Wyatt, and L. E. Collins, eds., *Practical Iterative Methods for Large Scale Computations*, Proceedings of the Minnesota Supercomputer Institute Workshop, Minneapolis, Oktober 1988, 1989, North-Holland, Amsterdam.

[6] C. Brezinski, *Padé-Type Approximants and General Orthogonal Polynomials*, ISNM, Vol.50, Birkhäuser, Basel/Bern/Stuttgart, 1980.

[7] P. Concus and G. H. Golub, A generalized conjugate gradient method for nonsymmetric systems of linear equations, Tech. Rep. STAN-CS-76-535, Computer Science Department, Stanford University, 1976.

[8] J. K. Cullum, W. Kerner, and R. Willoughby, A generalized nonsymmetric Lanczos procedure, Computer Physics Comm., 53 (1989), pp. 19–48.

[9] J. K. Cullum and R. A. Willoughby, *Lanczos Algorithms for Large Symmetric Eigenvalue Computations (2 Vols.)*, Birkhäuser, Boston-Basel-Stuttgart, 1985.

[10] G. Cybenko, An Explicit Formula for Lanczos Polynomials, Linear Algebra Applics., 88/89 (1987), pp. 99–115.

[11] S. C. Eisenstat, A note on the generalized conjugate gradient method, SIAM J. Numer. Anal., 20 (1983), pp. 358–361.

[12] H. C. Elman, *Iterative Methods for Large, Sparse Nonsymmetric Systems of Linear Equations*, PhD thesis, Computer Science Dept., Yale University, 1982.

[13] V. Faber and T. Manteuffel, Necessary and sufficient conditions for the existence of a conjugate gradient method, SIAM J. Numer. Anal., 21 (1984), pp. 352–362.

[14] ———, Orthogonal error methods, SIAM J. Numer. Anal., 24 (1987), pp. 170–187.

[15] R. Fletcher, Conjugate gradient methods for indefinite systems, in Numerical Analysis, G. A. Watson, ed., vol. 506 of Lecture Notes in Mathematics, Springer, Berlin/Heidelberg/New York, 1976, pp. 73–89.

[16] J. G. F. Francis, The QR transformation—a unitary analogue to the LR transformation, Comput. J., 4 (1961), pp. 265–271, 332–345.

[17] W. Gautschi, On the construction of Gaussian quadrature rules from modified moments, Math. Comp., 24 (1970), pp. 245–260.

[18] ———, On the sensitivity of orthogonal polynomials to perturbations in the moments, Numer. Math., 48 (1986), pp. 369–382.

[19] G. Golub and M. Gutknecht, Modified moments for indefinite weight functions, Numer. Math. (to appear).

[20] G. Golub and C. van Loan, *Matrix Computations*, John Hopkins University Press, Baltimore, 1983.

[21] G. H. Golub and J. H. Welsch, Calculation of Gauss quadrature rules, Math. Comp., 23 (1969), pp. 221–230.

[22] W. B. Gragg, The Padé table and its relation to certain algorithms of numerical analysis, SIAM Rev., 14 (1972), pp. 1–62.

[23] ———, Matrix interpretations and applications of the continued fraction algorithm, Rocky Mountain J. Math., 4 (1974), pp. 213–225.

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

[25] M. H. Gutknecht, A completed theory of the unsymmetric Lanczos process and related algorithms.

[26] J. Hadamard, *Essai sur l'étude des fonctions donnée par leurs développement de Taylor*, Gauthier-Villard, Paris, 1892.

[27] P. Henrici, Quotient-difference algorithm, in Mathematical Methods for Digital Computers, Vol. 2, A. Ralston and H. S. Wilf, eds., Wiley, New York, 1967.

[28] ———, *The quotient-difference algorithm*, vol. 49 of Applied Mathematics, National Bureau of Standards, 1958, ch. Further contributions to the solution of simultaneous linear equations and the determination of eigenvalues, pp. 23–46.

[29] ———, Some applications of the quotient-difference algorithm, Proc. Symp. Appl. Math. Amer. Math. Soc., 15 (1963), pp. 159–183.

[30] ———, *Applied and Computational Complex Analysis, Vol. 1*, Wiley, New York, 1974.

[31] M. R. Hestenes and E. Stiefel, Methods of conjugate gradients for solving linear systems, J. Res. Nat. Bureau Standards, 49 (1952), pp. 409–435.

[32] R. Hockney and C. Jesshope, *Parallel Computers*, Adam Hilger, Bristol, 1981.

[33] A. S. Householder, *The Theory of Matrices in Numerical Analysis*, Dover Publications, New York, 1964.

[34] K. C. Jea and D. M. Young, On the simplification of generalized conjugate-gradient methods for nonsymmetrizable linear systems, Linear Algebra Appl., 52 (1983), pp. 399–417.

[35] W. B. Jones and W. J. Thron, *Continued Fractions. Analytic Theory and Applications*, Addison-Wesley, Reading, Mass., 1980.

[36] W. D. Joubert and D. M. Young, Necessary and sufficient conditions for the simplification of generalized conjugate gradient algorithms, Tech. Rep. CNA-204, Center for Numerical Analysis, University of Texas at Austin, 1986.

[37] M. D. Kent, *Chebyshev, Krylov, Lanczos: Matrix Relationships and Computations*, PhD thesis, Stanford University, 1989.

[38] C. Lanczos, An iteration method for the solution of the eigenvalue problem of linear differential and integral operators, J. Res. Nat. Bureau Standards, 45 (1950), pp. 255–281.

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

[40] T. C. Oppe, W. D. Joubert, and D. R. Kincaid, NSPCG User's Guide. Version 1.0. A Package for Solving Large Sparse Linear Systems by Various Iterative Methods, Tech. Rep. CNA-216, Center for Numerical Analysis, University of Texas at Austin, 1988.

[41] H. Padé, Sur la représentation approchée d'une fonction par des fractions rationelles, Ann. de l'Ecole Normale Sup., 3ième Série, 9 (1892), pp. 3–93.

[42] C. C. Paige, *The computations of eigenvalues and eigenvectors of very large s parse matrices*, PhD thesis, University of London, 1971.

[43] ———, Computational variants of the Lanczos method for the eigenproblem, J. Inst. Math. Appl., 10 (1972), pp. 373–381.

[44] ———, Error analysis of the Lanczos algorithms for tridiagonalizing a symmetric matrix, J. Inst. Math. Appl., 18 (1976), pp. 341–349.

[45] ———, Accuracy and effectiveness of the Lanczos algorithm for the symmetric eigenproblem, Linear Algebra Appl., 34 (1980), pp. 235–258.

[46] B. Parlett, Reduction to tridiagonal form and minimal realizations. Preprint, Jan. 1990.

[47] B. N. Parlett, *The Symmetric Eigenvalue Problem*, Prentice-Hall, Englewood Cliffs, N.J., 1980.

[48] B. N. Parlett and B. Nour-Omid, Towards a black box Lanczos program, Computer Physics Comm., 53 (1989), pp. 169–179.

[49] B. N. Parlett and J. K. Reid, Tracking the progress of the Lanczos algorithm for large symmetric matrices, IMA J. Numer. Anal., 1 (1981), pp. 135–155.

[50] B. N. Parlett and D. S. Scott, The Lanczos algorithm with selective reorthogonalization, Math. Comp., 33 (1979), pp. 217–238.

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

[52] H. Rutishauser, Beiträge zur Kenntnis des Biorthogonalisierungs-Algorithmus von Lanczos, Z. Angew. Math. Phys., 4 (1953), pp. 35–56.

[53] ———, Une méthode pour la détermination des valeurs propres d'une matrice, C.R. Acad. Sci. Paris, 240 (1955), pp. 34–36.

[54] ———, *Der Quotienten-Differenzen-Algorithmus*, Mitt. Inst. angew. Math. ETH, Nr.7, Birkhäuser, Basel/Stuttgart, 1957.

[55] ———, *Solution of eigenvalue problems with the LR transformation*, vol. 49 of Applied Mathematics, National Bureau of Standards, 1958, ch. Further contributions to the solution of simultaneous linear equations and the determination of eigenvalues, pp. 1–22.

[56] ———, *Theory of gradient methods*, Mitt. Inst. angew. Math. ETH Zürich, Nr. 8, Birkhäuser, Basel, 1959, pp. 24–49.

[57] Y. Saad, Krylov subspace methods for solving large unsymmetric systems, Math. Comp., 37 (1981), pp. 105–126.

[58] ———, The Lanczos biorthogonalization algorithm and other oblique projection methods for solving large unsymmetric systems, SIAM J. Numer. Anal., 2 (1982), pp. 485–506.

[59] Y. Saad and M. H. Schultz, GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems, SIAM J. Sci. Stat. Comput., 7 (1986), pp. 856–869.

[60] E. B. Saff, Polynomial and rational approximation in the complex domain, in Approximation Theory, C. de Boor, ed., vol. 36 of Proc. Symposia Applied Mathematics, Amer. Math. Soc., 1986, pp. 21–49.

[61] W. Seewald, Quotienten-Differenzen-Algorithmus: Beweis der Regeln von Rutishauser, Numer. Math., 40 (1982), pp. 93–98.

[62] A. Sidi, Extrapolation vs. projection methods for linear systems of equations, J. Comput. Appl. Math., 22 (1988), pp. 71–88.

[63] H. D. Simon, Analysis of the symmetric Lanczos algorithm with reorthogonalization methods, Linear Algebra Appl., 61 (1984), pp. 101–131.

[64] ———, The Lanczos algorithm with partial reorthogonalization, Math. Comp., 42 (1984), pp. 115–142.

[65] P. Sonneveld, CGS, a fast Lanczos-type solver for nonsymmetric linear systems, SIAM J. Sci. Stat. Comput., 10 (1989), pp. 36–52.

[66] G. W. Stewart, On a companion operator for analytic functions, Numer. Math., 18 (1971), pp. 26–43.

[67] ———, Conjugate direction methods for solving systems of linear equtions, Numer. Math., 21 (1973), pp. 285–297.

[68] L. N. Trefethen, Non-normal matrices and pseudo-eigenvalues in numerical analysis. (to appear).

[69] R. S. Varga, *Matrix Iterative Analysis*, Prentice-Hall, Englewood Cliffs, N.J., 1962.

[70] P. K. W. Vinsome, ORTHOMIN—an iterative method for solving sparse sets of simultaneous linear equations, in Proc. Fourth SPE Symposium on Reservoir Simulation, Los Angeles, 1976, pp. 149–160.

[71] O. Widlund, A Lanczos method for a class of nonsymmetric systems of linear equations, SIAM J. Numer. Anal., 15 (1978), pp. 801–811.

[72] J. H. Wilkinson, *The algebraic eigenvalue problem*, Clarendon Press, Oxford, 1965.

[73] D. M. Young and K. C. Jea, Generalized conjugate-gradient acceleration of nonsymmetrizable iterative methods, Linear Algebra Appl., 34 (1980), pp. 159–194.