SIAM J. NUMER. ANAL.
Vol. 19, No. 3, June 1982

0036-1429/82/1903-0003 $01.00/0

THE LANCZOS BIORTHOGONALIZATION ALGORITHM AND OTHER OBLIQUE PROJECTION METHODS FOR SOLVING LARGE UNSYMMETRIC SYSTEMS*

Y. SAAD†

Abstract. Many powerful methods for solving systems of equations can be regarded as projection methods. Most of the projection methods known for solving linear systems are orthogonal projection methods but little attention has been given to the class of nonorthogonal (or oblique) projection methods, which is particularly attractive for large nonsymmetric systems. The purpose of the paper is to present some methods in the general setting of oblique projection methods and to give some theoretical results. Some experiments comparing the various algorithms are reported.

1. Introduction. In an earlier paper [16], some algorithms based on orthogonalization techniques have been proposed for solving large unsymmetric systems. Of particular interest is the incomplete orthogonalization method without correction, where at every step the solution is taken to make the new residual orthogonal to the p previous residuals where p is some small integer. As will be seen, this method can be regarded as an oblique projection method. By nonorthogonal or oblique projection method, we mean a method which seeks a solution $\tilde{x}$ of $Ax = b$ by requiring that $\tilde{x}$ belong to a certain subspace K (called the right space) and that the residual $b - A\tilde{x}$ be orthogonal to another subspace L (called the left subspace).

The best example of an oblique projection method for solving linear systems is provided by the method of Lanczos [8], which is a version of the well-known conjugate gradient method in the symmetric case [6]. In that method, the right space K is a Krylov subspace $K = \text{span}[v_1, Av_1, \cdots, A^{m-1}v_1]$ where v_1 is a starting vector, while L is a Krylov subspace associated with A^H, $L = \text{span}[w_1, A^H w_1, \cdots, (A^H)^{m-1}w_1]$. The Lanczos algorithm has been neglected for a long time because of its instability as a method for tridiagonalizing a nonsymmetric matrix and computing its eigenvalues, although recently this fact has been reconsidered by Parlett and Taylor [14]. For solving linear systems, however, the method can be quite useful, especially when it is used in conjunction with a preconditioning technique. We should point out that the presence of A^H in the definition of L does not mean at all that the Lanczos method solves the normal equations $A^H Ax = A^H b$.

It is not the purpose of this paper to introduce a specific method effective for any large unsymmetric system, but rather to present and analyze a class of methods based upon oblique projection processes. Most of the algorithms presented are already known or can be trivially derived from known algorithms.

Section 2 sets the basic definitions and notations of the oblique projection methods and treats the important example of the Lanczos method. In § 3, other oblique projection methods, such as the incomplete orthogonalization method and the Orthomin (p) method [18], are considered. The convergence properties of the algorithm are studied in § 5, and some numerical experiments are described in the last section comparing some of the methods treated.

* Received by the editors October 23, 1980, and in revised form June 23, 1981. This research was supported in part by the U.S. Department of Energy under grant DE-AC02-76-ER02383.A003, and in part by the Office of Naval Research under contract N0014-76-C-0013.

† Université de Grenoble, Mathématiques Appliquées Informatique, B.P. 53, 38041 Grenoble Cedex, France. Now at Department of Computer Science, Yale University, New Haven, Connecticut 06520.

2. Oblique projection methods and the Lanczos algorithm.

2.1. Oblique projection methods. Basic theory and notation. Let us consider the linear system

$$b - Ax = 0, \tag{2.1}$$

where A is an $n \times n$ nonsymmetric real matrix. Let $V_m \equiv [v_1, \cdots, v_m]$ and $W_m \equiv [w_1, \cdots, w_m]$ be two systems of m linearly independent vectors in $\mathbb{R}^n$. The span of V_m (resp. W_m) will be denoted by K_m (resp. L_m) and will often be referred to as the right (resp. left) space. An oblique projection method onto K_m and orthogonal to L_m is any process that obtains an approximate solution x_m to problem (2.1), which belongs to K_m and which satisfies the relations:

$$b - Ax_m \perp w_j, \qquad j = 1, \cdots, m. \tag{2.2}$$

If a good guess x_0 for the solution is available, it is more appropriate to seek an approximate solution of the form

$$x_m = x_0 + z_m, \tag{2.3}$$

where z_m belongs to K_m and where x_m is required to satisfy the same condition (2.2). In that case, the new unknown z_m is the solution of the problem

$$r_0 - Az_m \perp w_j, \qquad j = 1, \cdots, m, \tag{2.4}$$

where r_0 is the initial residual $b - Ax_0$.

Note that the first formulation is a particular case of the second with $x_0 = 0$ and that the second formulation can be reduced to the first because it amounts to solving the problem

$$r_0 - Az = 0. \tag{2.5}$$

The second formulation is important for restarting the algorithm. The more general formulation (2.3), (2.4) will often be adopted. It will be assumed throughout that r_0 belongs to K_m. Another important assumption that we shall make is that

$$\det (W_m^T V_m) \neq 0 \tag{2.6}$$

and

$$\det (W_m^T A V_m) \neq 0. \tag{2.7}$$

In that case, the problem (2.5) has a solution z_m which can be obtained as

$$z_m = V_m y_m, \tag{2.8}$$

where y_m is given by

$$y_m = (W_m^T A V_m)^{-1} W_m^T r_0. \tag{2.9}$$

Notice that while the condition (2.7) ensures the existence of an approximate solution (given by (2.8)–(2.9)), the condition (2.6) guarantees the uniqueness of that solution. Notice that (2.6) can be decomposed as

$$\dim (L_m) = \dim (K_m) = m \quad \text{and no vector of } K_m \text{ is orthogonal to } L_m.$$

2.2. The Lanczos method of biorthogonalization.

2.2.1. The biorthogonalization process. A very attractive example of the oblique protection process described above is the method proposed by Lanczos in [8]. In that method, Lanczos suggested a simple way to generate biorthogonal systems W_m, V_m

such that the matrix $W_m^T A V_m$ in (2.7) has a tridiagonal form. A simple version of his algorithm can be described as follows:

ALGORITHM 1.

1. Choose v_1 and w_1 such that $(v_1, w_1) = 1$.
2. For $j = 1, 2, \cdots, m$ do

(2.10) • $\hat{v}_{j+1} := Av_j - \alpha_j v_j - \beta_j v_{j-1}$

(2.11) • $\hat{w}_{j+1} := A^T w_j - \alpha_j w_j - \delta_j w_{j-1}$

(when $i = 1$ take $\beta_1 v_0 := \delta_1 w_0 := 0$)

(2.12) with $\alpha_j := (Av_j, w_j)$

• choose δ_{j+1} and β_{j+1} such that

(2.13) $\delta_{j+1}\beta_{j+1} := (\hat{v}_{j+1}, \hat{w}_{j+1})$,

(2.14) • $v_{j+1} := \dfrac{\hat{v}_{j+1}}{\delta_{j+1}}$,

(2.15) $w_{j+1} := \dfrac{\hat{w}_{j+1}}{\beta_{j+1}}$.

It can be easily shown that when the algorithm does not break down for a null inner product $(\hat{v}_{j+1}, \hat{w}_{j+1})$, then the vectors v_i and w_i satisfy the biorthonormality property:

$$(v_i, w_j) = \delta_{ij}, \qquad i, j = 1, \cdots, m. \tag{2.16}$$

An interesting choice for δ_{j+1} and β_{j+1} in (2.13) is the following:

$$\delta_{j+1} = |(\hat{v}_{j+1}, \hat{w}_{j+1})|^{1/2}, \qquad \beta_{j+1} = \delta_{j+1} \operatorname{sign}(\hat{v}_{j+1}, \hat{w}_{j+1}). \tag{2.17}$$

Although there are various ways of choosing δ_{j+1}, β_{j+1} satisfying (2.13), it is of interest to notice that the product $\|v_{j+1}\|\|w_{j+1}\|$ will not depend upon which choice is taken because

$$\|v_{j+1}\|\|w_{j+1}\| = \frac{\|\hat{v}_{j+1}\|\|\hat{w}_{j+1}\|}{\delta_{j+1}\beta_{j+1}} = \frac{\|\hat{v}_{j+1}\| \cdot \|\hat{w}_{j+1}\|}{|(\hat{v}_{j+1}, \hat{w}_{j+1})|},$$
$$\|v_{j+1}\|\|w_{j+1}\| = \frac{1}{\cos\theta(\hat{v}_{j+1}, \hat{w}_{j+1})}, \tag{2.18}$$

where $\theta(x, y)$ denotes the acute angle between the vectors x and y. The angle $\theta(\hat{v}_{j+1}, \hat{w}_{j+1})$ is a function of A, v_1, w_1 and j only because, as will be seen later on, the vectors $\hat{v}_{j+1}$, $\hat{w}_{j+1}$ are uniquely determined by v_1, w_1 apart from a normalizing factor. This angle can be equal to $\pi/2$, causing the algorithm to stop. As shown in an example of Wilkinson [19, p. 390], this can occur even when A is well conditioned, and should not be attributed to any shortcoming of the matrix A. It is interesting to note that $\hat{v}_{j+1}$ and $\hat{w}_{j+1}$ can be written as $\hat{v}_{j+1} = p_j(A)v_1$, $\hat{w}_{j+1} = p_j(A^T)w_1$, where p_j denotes a polynomial of degree j.

2.2.2. Solution of the linear system by the Lanczos method. The previous algorithm builds a system of biorthogonal vectors $\{v_i, w_i\}_{i=1,m}$ but does not provide explicitly an approximate solution for (2.1). In [8] Lanczos has proposed an interesting way to build up such an approximation. His algorithm, which was published at about

the same time as the conjugate gradient method of Hestenes and Stiefel [6], can be considered as a version of the conjugate gradient algorithm. The algorithm proposed by Lanczos decouples each of the relations (2.10) and (2.11) in two other relations involving a new sequence of vectors (the conjugate directions) in which the solution is easily expressed. The approximate solution x_m provided by Lanczos' algorithm belongs to $\{x_0\}+K_m$ and its residual, which is proportional to the vector v_{m+1}, is orthogonal to the left space L_m. It is therefore equal to the solution that would be obtained by an oblique projection method, using the subspaces $K_m = \text{span}\,(V_m)$ as a right space and $L_m = \text{span}\,(W_m)$ as a left space, where V_m, W_m is the biorthogonal system built by Algorithm 1.

In the following, we show how the approximate solution can be obtained directly as a combination of the vectors v_i. Another algorithm simpler and closer to the conjugate gradient method will be given in the next subsection.

Suppose that Algorithm 1 is started with[1] $v_1 = r_0/\|r_0\|$ and $w_1 = v_1$, and let us consider the component vector of y_m of z_m given by (2.9). From the algorithm and the biorthogonality property (2.16), it can be easily shown [19] that the $m \times m$ matrix $T_m = W_m^T A V_m$ has the tridiagonal form

$$T_m = \begin{bmatrix} \alpha_1 & \beta_2 & & \\ \delta_2 & \ddots & \ddots & \\ & \ddots & \ddots & \beta_m \\ & & \delta_m & \alpha_m \end{bmatrix}. \tag{2.19}$$

Notice that with the determination (2.17) of δ_{j+1}, T_m has the interesting additional property that

$$\beta_j = \pm\delta_j, \qquad j = 2, \cdots, m.$$

Furthermore, the right-hand side of the system (2.6) is equal to βe_1, where $\beta = \|r_0\|$; $e_1 = (1, 0, \cdots, 0)^T$, because $W_m^T r_0 = \beta W_m^T v_1 = \beta e_1$.

The approximate solution x_m is therefore quite easy to practically obtain, since we have

$$x_m = x_0 + V_m y_m = x_0 + \beta V_m T_m^{-1} e_1. \tag{2.20}$$

Computing the approximate solution by (2.20) requires the storage of the vectors $v_1, v_2, \cdots, v_m$, but this does not constitute a major drawback because the formation of the solution by (2.20) takes place only when the convergence has occurred, and therefore, the v_i's can be saved in secondary storage until then. This means, however, that we have to provide some means for determining whether the convergence is achieved without explicitly using the approximate solution. Fortunately, this can be done quite easily thanks to the following formula, well-known and tremendously useful in the symmetric case [12], [13], which expresses the residual norm in terms of y_m and $\|\hat{v}_{m+1}\|$,

$$\|b - Ax_m\| = \|\hat{v}_{m+1}\| |e_m^T y_m|. \tag{2.21}$$

Equality (2.21) enables us to compute the residual norm very economically and one can afford to make use of (2.21) periodically to monitor the convergence. Let us mention that it is not even necessary to actually compute y_m in order to estimate the residual norm. (See a similar point in [13] and [16].)

[1] It is not necessary to start with $w_1 = v_1$ but it is somewhat simplifying.

ALGORITHM 2.

1. Choose an initial vector x_0, compute

$$r_0 := b - Ax_0,$$

$$\beta := \|r_0\|,$$

$$v_1 := w_1 := \frac{r_0}{\beta}.$$

2. For $j = 1, 2, \cdots, s_{\max}$ do
   a. compute v_{j+1}, w_{j+1}, α_j, β_{j+1}, δ_{j+1} by formulae (2.10) to (2.15) of Algorithm 1.
   b. periodically (e.g., when $[j/5] \cdot 5 = j$) update the estimate ρ_j of the residual norm. If $\rho_j \leqq \varepsilon$ go to 3 else continue.
3. Form the approximate solution

$$x_j = x_0 + \beta V_j T_j^{-1} e_1. \tag{2.22}$$

2.3. Equivalent versions and the biconjugate gradient algorithm. We will now give some equivalent versions of the basic algorithms 1 and 2. This can be done by regarding Algorithm 1 as an algorithm for building a sequence of orthogonal polynomials. This point of view, developed in [2] and [5], is useful and simplifies the statements.

2.3.1. The Lanczos algorithm and orthogonal polynomials. Any vector x of the Krylov subspace K_m is of the form $x = p(A)v_1$, where p belongs to the space $\mathbb{P}_{m-1}$ of polynomials of degree not exceeding $m-1$. Clearly, this establishes a one-to-one relation between K_m and $\mathbb{P}_{m-1}$. Consider the bilinear form $\langle p, q\rangle = (p(A)v_1, q(A^T)w_1)$. This defines an indefinite inner product in $\mathbb{P}_{m-1}$. An important question is whether a sequence of orthogonal polynomials associated with this bilinear form can be constructed.

This question is usually answered in terms of the moment matrices M whose general terms m_{ij} are defined by $m_{ij} = \langle x^{i-1}, x^{j-1}\rangle = (A^{i+j-2}v_1, w_1)$, $1 \leqq i, j \leqq k$.

LEMMA 1. *The first m orthogonal polynomials associated with the above bilinear form exist if and only if*

$$\det(M_k) \neq 0, \qquad k = 1, 2, \cdots, m. \tag{2.23}$$

See [2] for the proof.

When (2.23) is satisfied, then there exists a sequence p_m of orthogonal polynomials which can be obtained by a three term recursion formula. It is clear that the relation (2.10) of Algorithm 1 corresponds to such a recursion, which means that Algorithm 1 constructs the sequence of orthogonal polynomials relative to the bilinear form $\langle \cdot, \cdot \rangle$. However, the Lanczos algorithm is not the only algorithm which permits us to build up the sequence of orthogonal polynomials relative to $\langle \cdot, \cdot \rangle$, as is shown next.

2.3.2. A generalization of Algorithm 1. The following algorithm generalizes Algorithm 1 into a whole class of equivalent versions.

ALGORITHM 3.

1. Choose v_1 and w_1 as in Algorithm 1.
2. For $j = 1, 2, \cdots, m$ do

$$\begin{cases} \hat{v}_{j+1} := Av_j - \alpha_j v_j - \beta_j v_{j-1}, \\ \hat{w}_{j+1} := A^T w_j - \sum_{i=1}^{j} h_{ij} w_i, \end{cases} \tag{2.24}$$

where $\alpha_j = (Av_j, w_j) - (Av_{j-1}, w_{j-1}) + h_{j-1,j-1}$ (when $j=1$ take $\beta_1 v_0 = 0$ and $\alpha_1 = (Av_1, v_1)$) and where the h_{ij}'s are arbitrary.

- Normalize v_{j+1} and w_{j+1} by using formulae (2.13), (2.14), (2.15), (2.16) of Algorithm 1.

Obviously, Algorithm 1 is a particular case of the above algorithm, with $h_{j,j} = (Av_j, v_j)$, $h_{j-1,j} = \delta_j$ and $h_{ij} = 0$ for $i < j-1$.

For every particular choice of the parameters h_{ij}, one obtains a set of vectors $V_m = [v_1, v_2, \cdots, v_m]$ and a tridiagonal matrix T_m defined by (2.19). What is surprising is that, theoretically, whatever the h_{ij}'s are, the above algorithm will always produce the same right vectors v_i, the same tridiagonal matrix T_m, and therefore the same approximate solution x_m as is stated in the next proposition.

PROPOSITION 1. *Suppose that Algorithm* 3 *is feasible for a given pair of starting vectors* v_1 *and* w_1. *Then*

(i)

$$(v_j, w_k) = \delta_{jk}, \qquad 1 \leqq k \leqq j \leqq m+1. \tag{2.25}$$

(ii) *The system* $V_m = [v_1, v_2, \cdots, v_m]$ *and the matrix* T_m *produced by Algorithm* 3 *do not depend upon the choice of the parameters* h_{ij} *used in* (2.24).

Proof. (i) We omit the proof of this part which is by induction and straightforward.

(ii) From (i), it is clear that the vector v_k satisfies $v_k = P_k(A)v_1$ and $(p_k(A)v_1, q(A^T)w_1) = 0$ for any polynomial q of degree $\leqq k-1$. So the uniqueness of the orthogonal polynomials shows that the v_k's produced by Algorithms 1 and 3 are the same apart from a multiplicative constant. Let us now show that they are exactly the same. If the vector w_k is divided by its leading coefficient in $(A^T)^{k-1}v_1$, we get the vector

$$\tilde{w}_k = (A^T)^{k-1} w_1 + \sum_{i=0}^{k-2} \xi_i (A^T)^i w_1.$$

Using a similar notation for v_k, we consider the inner product

$$(\tilde{v}_k, \tilde{w}_k) = (\tilde{v}_k, (A^T)^{k-1} w_1 + \sum_{i=1}^{k-2} \xi_i (A^T)^i w_1) = (\tilde{v}_k, (A^T)^{k-2} w_1),$$

because $\tilde{v}_{i+1}$ is orthogonal to $(A^T)^i w_1$, $i \leqq k-2$. This means that the normalizing factor that would be used to get an inner product equal to unity does not depend on the h_{ij}'s. To complete the proof, we have to show that the matrix T_m does not depend on the parameters h_{ij}. This is due to the fact that we are generating the same sequence of orthogonal polynomials by different three terms recurrences which employ the same normalization conditions. Therefore, the coefficients α_i, β_i, δ_i for these recurrences are the same.

Remark. When $W_m^T V_m$ is not the identity, the matrix $W_m^T A V_m$ will not be equal to T_m and is not a similarity transformation of A. However, the matrix $(W_m^T V_m)^{-1} W_m^T A V_m$ is equal to T_m and replaces the usual $W_m^T A V_m$ of the case when $W_m^T V_m$ is the identity.

One might wonder whether it is possible to find, among all the possible choices of the parameters h_{ij}, one which makes the algorithm more efficient than Algorithm 1.

Although the result of Proposition 1 may seem powerful, it has little practical value for the solution of systems of equations as it seems that the most economical and stable version is precisely Algorithm 1.

The only practical application of the above results lies in the problem of reorthogonalization. It shows that, in fact, the orthogonality of the w_i's against the v_i *is not necessary*. We only need to have the v_i's orthogonal to the w_i's, and if there is any need to reorthogonalize, only the v_i's should be reorthogonalized against the previous w_i's. This point, however, might be more important for eigenvalue approximations than for the solution of linear equations.

2.3.3. The biconjugate gradient algorithm. The solution x_m provided by Algorithm 2 can be obtained by a conjugate gradient-like method which is a slight variation of Lanczos' method for solving linear systems [8]. The version given below can be found in [4].

ALGORITHM 4.

1. Choose an initial guess x_0 of the solution
2. Compute $r_0 = b - Ax_0$ and take $p_0^* := r_0^* := p_0 := r_0$
3. For $k = 1, 2, \cdots, m, \cdots$ compute

$$x_{k+1} := x_m + \alpha_k p_k,$$

$$r_{k+1} := r_k - \alpha_k A p_k, \tag{2.26}$$

$$r_{k+1}^* := r_k^* - \alpha_k A^T p_k^*, \tag{2.27}$$

$$p_{k+1} := r_{k+1} + \beta_k p_k, \tag{2.28}$$

$$p_{k+1}^* := r_{k+1}^* + \beta_k p_k^*, \tag{2.29}$$

with

$$\alpha_k := (r_k, r_k^*)/(Ap_k, p_k^*), \tag{2.30}$$

$$\beta_k := (r_{k+1}, r_{k+1}^*)/(r_k, r_k^*). \tag{2.31}$$

PROPOSITION 2. *The vectors r_k, r_k^* and p_k, p_k^* produced by Algorithm* 4 *are such that*:

a. $(r_k, r_j^*) = 0$ *for* $j \neq k$ (*biorthogonality property*),

b. $(Ap_k, p_j^*) = 0$ *for* $j \neq k$ (*biconjugacy property*).

See [4] for the proof.

Let us mention that all the relations that hold for the classical conjugate gradient method will hold for the biconjugate gradient method if the vectors on the right parts of the inner products are replaced by the corresponding vectors p_k^*, r_k^*, etc.

It is important not to confuse this algorithm with the bidiagonalization method [10], where one essentially solves the normal equations. The bidiagonalization methods are projection methods on the subspaces span $[r_0, (A^H A)r_0, \cdots, (A^H A)^{m-1} r_0]$, while we are dealing with an oblique projection method on the subspace $K_m =$ span $[r_0, Ar_0, \cdots, A^{m-1} r_0]$.

That Algorithm 4 is theoretically equivalent to Algorithm 2 can be simply established as follows:

The solutions obtained by both algorithms satisfy $x_k = x_0 + z_k$, where z_k is such that

$$z_k \in K_k = \text{span}\,[r_0, Ar_0, \cdots, A^{k-1} r_0],$$

$$r_k = r_0 - Az_k \perp L_k = \text{span}\,[r_0, A^H r_0, \cdots, (A^H)^{k-1} r_0].$$

Therefore, $z_k = \sum_{i=1}^{k} \eta_i A^{i-1} r_0$ for both methods and the η_i's are solutions of the linear system

$$\left(r_0 - A \sum_{i=1}^{k} \eta_i A^{i-1} r_0, (A^T)^{j-1} r_0\right) = 0, \qquad j = 1, 2, \cdots, k. \tag{2.32}$$

Assuming that the moment matrix M'_k, whose general elements m'_{ij} are $m'_{ij} = (A^{i+j-1} r_0, r_0)$, is nonsingular[2], we conclude that the vectors z_k produced by both algorithms are the same because of the uniqueness of the solution of the system (2.32).

On the practical side, Algorithm 4 presents the advantage of requiring less storage than Algorithm 2. It can be coded with six vectors of length n in core memory, while the Lanczos algorithm needs five vectors in main memory and m vectors in secondary storage (when m is large, the latter may involve substantial input/output operation times).

Furthermore, the number of arithmetic operations required is slightly in favor of Algorithm 4 because there is no tridiagonal system to solve. Note, however, that the cost of solving the tridiagonal system is negligible. Finally, because stable methods can be used to solve the tridiagonal system, Algorithm 2 is in general more stable than Algorithm 4.

2.4. Feasibility of the Lanczos algorithm and the biconjugate gradient algorithm. Thus far, we have not discussed under which conditions Algorithms 2 and 4 are feasible. The moment matrices mentioned in the previous subsection play an important role as is seen in the next proposition.

PROPOSITION 3. *Let M_k and M'_k be the $k \times k$ moment matrices whose general terms are defined by $m_{ij} = (A^{i+j-2} v_1, v_1)$ and $m'_{ij} = (A^{i+j-1} v_1, v_1)$, respectively. Then the* mth *approximate solutions x_m can be computed by Algorithm 2 if and only if*

$$\det(M_k) \neq 0, \qquad k = 1, 2, \cdots, m, \tag{2.33}$$

$$\det(M'_m) \neq 0. \tag{2.34}$$

Proof. First notice that (2.33) is equivalent to the constructibility of the Lanczos' polynomials $p_1, p_2, \cdots, p_m$, which is equivalent to the constructibility of the Lanczos vectors $v_1, v_2, \cdots, v_m$ (see § 2.2.1). Then observe that when both $\bar{V}_m = [v_1, Av_1, \cdots, A^{m-1} v_1]$ and $\bar{W}_m = [w_1, A^T w_1, \cdots, (A^T)^{m-1} w_1]$ have rank m, the tridiagonal matrix T_m can be written as

$$T_m = S_m^T M'_m U_m, \tag{2.35}$$

where S_m and U_m are $m \times m$ and nonsingular. This is because we can write $\bar{V}_m = V_m U_m$, $\bar{W}_m = W_m S_m$, $T_m = W_m^T A V_m$ and $M'_m = \bar{W}_m^T A \bar{V}_m$ for some nonsingular $m \times m$ matrices U_m and S_m.

Assuming that x_m can be obtained by Algorithm 2 means that $v_1, v_2, \cdots, v_m$ are computable and implies that (2.33) is satisfied. Since (2.33) is satisfied, we have rank $(\bar{V}_m) = \text{rank}\,(\bar{W}_m) = m$ because $M_m = \bar{W}_m^T \bar{V}_m$, and by (2.35), we see that (2.34) is true because T_m is nonsingular. This shows the "necessary" part of the proposition.

Assuming that (2.33) is true implies that we can build up the system V_m and the tridiagonal matrix T_m. If in addition (2.34) is true, then by (2.35) T_m is nonsingular, which means that we can compute x_m by (2.20). □

An important remark which can be derived immediately from the proof is that the condition (2.33) ensures that $v_1, v_2, \cdots, v_m$ and $w_1, w_2, \cdots, w_m$ can be built, while

[2] In § 2.4, we shall see that this assumption is necessary for the feasibility of Algorithm 2.

(2.34) ensures that the tridiagonal matrix T_m is nonsingular. It is therefore obvious that the proposition can be generalized as follows:

The approximations $x_{k_1}, x_{k_2}, \cdots, x_{k_m}$ can be built by Algorithm 2 if and only if $\det(M_j) \neq 0$, $j = 1, 2, \cdots, k_m$ and $\det(M'_{k_j}) \neq 0$, $j = 1, 2, \cdots, m$. For the biconjugate gradient method we have the following analogue of the above result.

PROPOSITION 4. *The first m steps of Algorithm* 4 *can be performed if and only if*

$$\det(M_k) \neq 0, \qquad k = 1, 2, \cdots, m, \tag{2.36}$$

$$\det(M'_k) \neq 0, \qquad k = 1, 2, \cdots, m. \tag{2.37}$$

Proof. It is equivalent to show that the algorithm breaks down if and only if either $\det(M_k) = 0$ or $\det(M'_k) = 0$ for some k, $1 \leqq k \leqq m$. It is clear that the algorithm will break down at step k when either of the inner products (r_k, r_k^*) or (Ap_k, p_k^*) is zero. Thus, the proposition will be proved if we show that $(r_k, r_k^*) = 0$ if and only if $\det(M_k) = 0$ and $(Ap_k, p_k^*) = 0$ if and only if $\det(M'_k) = 0$.

Suppose that $(r_k, r_k^*) = 0$. Then, by Proposition 2, $(r_k, r_j^*) = 0$, $j = 1, 2, \cdots, k$, which implies that r_k is orthogonal to the subspace $L_m = \operatorname{span}[w_1, A^T w_1, \cdots, (A^T)^{k-1} w_1]$. Using the notation of the proof of Proposition 3 and writing r_k as $r_k = \bar{V}_k s_k$, we see that $\bar{W}_k^T \bar{V}_k s_k = 0$ with $s_k \neq 0$, and therefore $\det(M_k) = 0$. If $\det(M_k) = 0$ then we must have $(r_k, r_k^*) = 0$; otherwise, the sequence $r_1, r_2, \cdots, r_k$ would correspond to an orthogonal sequence of polynomials, and this would contradict Lemma 1. The rest of the proposition can be proved in a similar way. □

3. Other oblique projection methods. The purpose of this section is to attempt to derive some other oblique projection methods. It will first be seen that the incomplete orthogonalization method without correction presented in [16] is nothing but an oblique projection method. Then, based upon an analogue of Proposition 1, we shall describe a particular class of the oblique projection methods for the solution of linear systems.

3.1. The incomplete orthogonalization method (I.O.M.). Among the methods proposed in [16], the incomplete orthogonalization method without correction was found to be the most attractive. A simple description of the method is the following:

ALGORITHM 5.

a. Choose two integers p and m, and construct a system of vectors $v_1, v_2, \cdots, v_m$ by:

1. $v_1 := r_0/(\beta := \|r_0\|)$, with $r_0 = b - Ax_0$.
2. For $j = 1, \cdots, m$,

$$\hat{v}_{j+1} := Av_j - \sum_{i=i_0}^{j} h_{ij} v_i, \tag{3.1}$$

where $i_0 = \max\{1, j-p+1\}$,

$$h_{ij} = (Av_j, v_i), \tag{3.2}$$

$$v_{j+1} := \hat{v}_{j+1}/(h_{j+1,j} := \|\hat{v}_{j+1}\|). \tag{3.3}$$

b. Take as the approximate solution

$$x_m \equiv x_0 + \beta V_m H_m^{-1} e_1, \tag{3.4}$$

where $V_m \equiv [v_1, \cdots, v_m]$ and where H_m is the Hessenberg matrix whose nonzero elements are the h_{ij} computed by (3.2) and (3.3). Note that $\hat{v}_{j+1}$ is obtained by orthogonalizing Av_j against the previous p vectors.

The above method was founded upon the fact that if we compare the solution (3.4) with that provided by Arnoldi's method (an orthogonal projection method upon the Krylov subspace K_m), we would find that the difference between them is negligible provided that the system $[v_1, \cdots, v_m]$ remains nearly orthogonal [16], a fact which is often observed.

We now would like to give an interpretation of the method in terms of oblique projection methods. More precisely, we shall exhibit a system of left vectors $w_1, \cdots, w_m$ such that the I.O.M. algorithm will amount to performing an (oblique) projection method onto $K_m = \text{span}[v_1, Av_1, \cdots, A^{m-1}v_1]$ and orthogonal to $L_m = \text{span}[W_m]$.

Consider the system of vectors w_i obtained from $v_1, v_2, \cdots, v_m, v_{m+1}$ as follows:

$$w_i = v_i - (v_i, v_{m+1})v_{m+1}, \qquad i = 1, 2, \cdots, m. \tag{3.5}$$

Each of the vectors w_i is orthogonal to v_{m+1}, so that if we set $W_m \equiv [w_1, \cdots, w_m]$ we get

$$W_m^T v_{m+1} = 0. \tag{3.6}$$

We can then state the next result.

PROPOSITION 5. *Let $V_m = [v_1, \cdots, v_m]$ be the system obtained from Algorithm 5, and let $W_m = [w_1, \cdots, w_m]$ be defined by (3.5). Then the approximate solution provided by the incomplete orthogonalization method is equal to that obtained by the oblique projection method using $K_m = \text{span}[V_m]$ as the right space and $L_m = \text{span}[W_m]$ as the left space.*

Proof. From (3.1), it can be shown that

$$AV_m = V_m H_m + h_{m+1,m} v_{m+1} e_m^T,$$

which gives, on multiplying both sides by W_m^T, $W_m^T A V_m = W_m^T V_m H_m + h_{m+1,m} W_m^T v_{m+1} e_m^T$. Using (3.6) and assuming that $W_m^T V_m$ is nonsingular, we get

$$(W_m^T V_m)^{-1} W_m^T A V_m = H_m. \tag{3.7}$$

From (2.6), it is seen that the solution x'_m obtained by the oblique projection method, using as left space span $[W_m]$ and right space span $[V_m]$, is given by

$$x'_m = x_0 + V_m (W_m^T V_m)^{-1} W_m^T A V_m W_m^T r_0,$$

and since $r_0 = \beta v_1 = \beta V_m e_1$, we have

$$x'_m = x_0 + \beta V_m (W_m^T A V_m)^{-1} W_m^T V_m e_1,$$

which in view of (3.7) gives

$$x'_m = x_0 + \beta V_m H_m^{-1} e_1.$$

But this is just the solution (3.4) provided by the I.O.M. method, and the proof is complete. □

Note that in the case when v_{m+1} is orthogonal to $v_1, v_2, \cdots, v_m$, the w_i's coincide with the v_i's, which means that the method becomes an orthogonal projection method. In that case it would give theoretically the same result as Arnoldi's method [16]. It is precisely the aim of the incomplete orthogonalization method with corrections described in [16] to attempt to orthogonalize v_{m+1} against all previous vectors $v_1, v_2, \cdots, v_m$ by finding scalars h_{im}, $i = 1, \cdots, m$ such that

$$Av_m - \sum_{i=1}^{m} h_{im} v_i \perp v_i, \qquad i = 1, \cdots, m.$$

This, however, is difficult to achieve in practice because the previous v_i's, $i = 1, \cdots, m$ no longer form an orthogonal system, and therefore, the coefficients h_{im} can be found only by solving a least squares problem.

3.2. A particular class of oblique projection methods for linear systems.

3.2.1. Generalized Hessenberg processes. The results of the previous section can be extended to yield a whole class of oblique projection methods. Suppose that we start with $v_1 = r_0/\beta$, where $\beta = \|r_0\|$, and that we build a sequence of vectors $v_1, v_2, \cdots, v_m$, by the general formula

$$h_{j+1,j}v_{j+1} = Av_j - \sum_{i=1}^{j} h_{ij}v_i, \tag{3.8}$$

where the h_{ij}, $i = 1, 2, \cdots, j+1$, are determined such as to make the vector v_{j+1} satisfy certain conditions such as, for example, $(v_{j+1}, v_i) = \delta_{ij}$, $i = 1, \cdots, j+1$ (which gives the method of Arnoldi). Such processes, called the generalized Hessenberg processes by Wilkinson [19], have in common the equation

$$AV_m = V_m H_m + h_{m+1,m}v_{m+1}e_m^T, \tag{3.9}$$

where V_m and H_m are defined as before. Let us then consider the solution x_m obtained by applying the formula (3.4) in the same way as in the incomplete orthogonalization method. Such an approximate solution will have a residual vector proportional to the last vector v_{m+1} obtained from (3.8), because, from relation (3.9), we can show that

$$b - Ax_m = h_{m+1,m}e_m^T y_m v_{m+1}. \tag{3.10}$$

Thus, by requiring that the vector v_j satisfy certain conditions when building the sequence $\{v_j\}$ by (3.8), we implicitly require that the same conditions be satisfied by the jth residual vector.

Let us now show that the generalized Hessenberg processes belong to the class of oblique projection methods.

Suppose that $W_m = [w_1, w_2, \cdots, w_m]$ is any system of vectors such that

$$W_m^T v_{m+1} = 0, \tag{3.11}$$

$$\det(W_m^T V_m) \neq 0. \tag{3.12}$$

(Note that W_m is not unique.) Then it can be shown by the same proof as that of Proposition 5 that x_m is exactly the solution that would be obtained with an oblique projection method using as right space the space $K_m = \text{span}[v_1, \cdots, A^{m-1}v_1]$ and as left space the space $G_m = \text{span}[W_m]$.

Clearly, the methods of Lanczos and the I.O.M. are particular cases. In the Lanczos method, the h_{ij}'s, $i = 1, \cdots, j$, are chosen such that v_{j+1} is orthogonal to all the left space $L_m = \text{span}[w_1, A^T w_1, \cdots, (A^T)^{m-1}w_1]$, and it turns out that this can be realized by the elegant Algorithm 1 of Lanczos in which $h_{ij} = 0$ for $i < j-1$. In the I.O.M., the coefficients h_{ij} are taken such as to make v_{j+1} orthogonal to the p previous v_i's. Some other applications are described next.

3.2.2. ORTHOMIN and the conjugate residual method. Suppose that the coefficients h_{ij} in (3.8) are determined such as to make at each step j the vector v_{j+1} orthogonal to the vectors $Av_1, Av_2, \cdots, Av_j$. The h_{ij}'s can then be obtained by solving the $j \times j$ system

$$\sum_{i=1}^{j} (v_i, Av_k)h_{ij} = (Av_j, Av_k), \qquad k = 1, 2, \cdots, j. \tag{3.13}$$

Notice that $(v_i, Av_k) = 0$ for $k < i$, so that the above system is triangular. This is nothing but an oblique projection method, with $K_m = \text{span}[v_1, Av_1, \cdots, A^{m-1}v_1]$ as right space and $L_m = AK_m$ as left space. It can be shown that the solution obtained by this method minimizes $\|b - Ax\|$ over the affine subspace $x_0 + K_m$.

This method was presented in a simplified version by Vinsome [18]. It was also analyzed by Axelsson [1] and by Eisenstat, et al. [3], who give some results on the convergence theory. The simplified version, called ORTHOMIN by Vinsome, produces directly x_m as a sequence of the form $x_m = x_{m-1} + \alpha_m p_m$, where p_m is the direction of search.

ALGORITHM 6 (ORTHOMIN or generalized conjugate residual method).

1. *Start* x_0 initial vector. Compute $r_0 = b - Ax_0$, take $p_0 = r_0$.
2. *Iterate*

$$x_{k+1} = x_k + \alpha_k p_k, \qquad \alpha_k = \frac{(r_k, Ap_k)}{(Ap_k, Ap_k)},$$

$$r_{k+1} = r_k - \alpha_k A p_k,$$

$$p_{k+1} = r_{k+1} - \sum_{i=1}^{k} \beta_{ik} p_i, \qquad \beta_{ik} = \frac{(Ar_{k+1}, Ap_i)}{(Ap_i, Ap_i)}.$$

The coefficient α_k is chosen such that the residual r_{k+1} is orthogonal to Ar_k, while the β_{ik}'s are such that Ap_{k+1} is orthogonal to Ap_i, $i \leqq k$. Under these conditions, it can be shown that $(r_{k+1}, Ar_i) = 0$, $i \leqq k$ (which is equivalent to the condition $(v_{k+1}, Av_i) = 0$, $i \leqq k$ of the previous version), and hence, the residuals are "conjugate." Notice that since A is nonsymmetric, the conjugacy holds only on one side because it is not true that $(v_{k+1}, Av_i) = 0$ for $i > k+2$. It would be more appropriate to say that residuals are "semiconjugate."

The amount of work and the storage required in Algorithm 6 is prohibitive, and unless the algorithm is used iteratively with periodic restarting, it would be of little practical value. Vinsome then suggested performing an Incomplete Orthogonalization to generate the p_k's. The idea is similar to that of I.O.M. and consists of truncating the sum defining p_{k+1} in Algorithm 6 as follows:

$$p_{k+1} = r_{k+1} - \sum_{i=k-p+1}^{k} \beta_{ik} p_i.$$

Obviously, this is still an oblique projection method. If we compare Algorithm 6 with the I.O.M., we will find that while the amount of work is similar, the storage is in favor of the latter. However, ORTHOMIN is certainly easier to study theoretically because of the minimum residual property. Numerical tests will compare the two methods in the last section.

3.2.3. The modified Hessenberg process. In the method of Hessenberg for reducing a matrix to Hessenberg form [19], the h_{ij}'s in (3.8) are chosen such that v_{j+1} has zero components in its j first positions. The h_{ij}'s are found by solving a $j \times j$ triangular system. Therefore, $K_m = \text{span}[v_1, Av_2, \cdots, A^{m-1}v_1]$ and $L_m = \text{span}[e_1, e_2, \cdots, e_m]$. A natural simplification similar to the ideas used in I.O.M. and ORTHOMIN (p) is to save the previous p vectors only, replace (3.8) by

$$h_{j+1,j} v_{j+1} = Av_j - \sum_{i=j-p+1}^{j} h_{ij} v_i$$

and define h_{ij} to make $p+1$ components of the vector v_{j+1} equal to zero. An important question is how to choose the positions in which the zeros must appear. Some experiments have motivated us to prefer the following choice: Eliminate the components having the largest modulus among the vectors $v_j, v_{j-1}, \cdots, v_{j-p+1}$. This is only one choice among many others. The modified Hessenberg process described here has the advantage of not requiring any inner products.

4. Some practical considerations. In practice, all of the methods described in this paper can be considerably improved by the use of preconditioning techniques. This is important not only for a better rate of convergence but also because in general the resulting matrix will be closer to a symmetric matrix than the original one. The preconditioning technique proposed by Meijerink and van der Vorst in [9] obtains an incomplete LU factorization of an (unsymmetric) M-matrix $A = \tilde{L}\tilde{U} + E$. Then the original problem $Ax = b$ is replaced by $\tilde{U}^{-1}\tilde{L}^{-1}Ax = \tilde{U}^{-1}\tilde{L}^{-1}b$ or $\tilde{L}^{-1}A\tilde{W}^{-1}(Wx) = \tilde{L}^{-1}b$. The strongly implicit procedure due to Stone [17] has been applied quite successfully in conjunction with the incomplete orthogonalization method; see [16]. Some progress has recently been achieved and the problem of preconditioning a general unsymmetric matrix is being thoroughly investigated.

Another important practical question in regard with the Lanczos algorithm concerns the problem of breakdown. What should we do in case the inner product $(\hat{v}_k, \hat{w}_k)$ is zero or small? The easy alternative of restarting the algorithm is always possible but may not be satisfactory as restarting often means a slowing down of convergence. Parlett and Taylor [14] suggest an elegant way of dealing with the most common types of breakdowns. Their idea is based on the fact that the vectors v_{k+2} and w_{k+2} can still be defined in a certain way even when v_{k+1}, w_{k+1} are not defined.

Block generalizations of the algorithms described in the previous sections are straightforward to derive and can be useful for solving systems with several right-hand sides. An important advantage of storing the vectors v_i, is when solving a system with several successive right hand sides that are not much different from each other. As suggested in [13], a projection process on the previous system $\{v_i\}$ can then give a quite satisfactory result for the new system. This is also relevant when solving a system $A'x' = b'$ where A' and b' are slight modifications of A and b. There might be cases, however, where it is not desirable to use secondary storage because, for example, the I/O operations are too expensive on the computer used. How can we adapt our algorithms to cope with this restriction? The answer to this question is made clear by observing that all the methods described in this paper have two different versions; a version using explicitly the v_i's which necessitates auxiliary storage, and a "simplified" version which does not. For example, a simplified version for the I.O.M. can be found without difficulty by writing the condition that the residual r_k is orthogonal to the previous p residuals. We should point out, however, that the simplified versions will be less stable in general than the versions using the v_i's unless we derive algorithms based implicitly on the LQ factorization of the banded Hessenberg matrices H_m, similar to the well-known SYMMLQ algorithm for the symmetric undefinite case [11].

5. Convergence properties. In this section, the difficult problem of the convergence of the approximate solution x_m toward the exact solution x^* will be considered. It is important to clarify what is meant by convergence. First, if we assume that the w_i's, $i = 1, 2, \cdots, n$, are linearly independent, then the approximate solution x_m will converge to x^* in at most n steps. This is because if we write the condition (2.2) in the form $W_n^T(b - Ax_n) = 0$, we obtain on multiplying by $(W_n^T)^{-1}$, $x_n = A^{-1}b = x^*$. Therefore, the sequence x_m is a finite sequence, and by studying the convergence of

x_m, we shall mean deriving some properties which will ensure that x_m may be a good approximation to x^* even for m much smaller than the dimension n of the problem. The analysis proposed here is essentially the same as that given in our previous paper [16], and we shall only emphasize those results that present nontrivial differences.

Let P_m be the orthogonal projector onto the subspace K_m. We shall study the convergence in terms of the distance $\varepsilon_m = \|(I - P_m)z^*\|$, where z^* is the exact solution of the problem (2.5) and where $\|\cdot\|$ denotes the Euclidean norm. This distance between z^* and the subspace K_m has been fully studied in [16], and some bounds for it have been established, showing that in general ε_m is a quantity which decays rapidly.

We will need an interpretation of the oblique projection method in terms of operator equations. Let us define the projector Q_m onto K_m and orthogonal to L_m by

$$Q_m x \in K_m \text{ and } x - Q_m x \perp L_m.$$

The vector $Q_m x$ is uniquely defined only under the assumption that no vector of K_m is orthogonal to L_m, which is equivalent to the assumption (2.6). Let us then define the operator[3] $A_m = Q_m A P_m$ and make the assumptions (2.6), (2.7) of § 2.1. We then have:

LEMMA 2. *The problem*

$$z \in K_m, \tag{5.1}$$

$$r_0 - A_m z = 0 \tag{5.2}$$

has as its unique solution the approximate solution z_m provided by the oblique projection method using K_m as right space and L_m as left space.

Proof. It is sufficient to translate problems (5.1), (5.2) into matrix notation. Since $z \in K_m$, it can be written as

$$z = V_m y. \tag{5.3}$$

Furthermore, r_0 and z belong to K_m, and therefore, $P_m z = z$ and $Q_m r_0 = r_0$. The matrix representation of Q_m in the canonical basis is $V_m(W^T V_m)^{-1} W_m^T$, and so (5.1), (5.2) give

$$V_m(W_m^T V_m)^{-1} W_m^T r_0 - V_m(W_m^T V_m)^{-1} W_m^T A V_m y = 0,$$

which yields

$$y = (W_m^T A V_m)^{-1} W_m^T r_0. \tag{5.4}$$

This means that the problem (5.1), (5.2) has a unique solution. A comparison between (5.3), (5.4) on the one hand and (2.8), (2.9) on the other hand shows that the solution is just that obtained by the projection method. □

We shall refer to problem (5.1), (5.2) as the approximate problem. What the lemma shows is that the projection method described in § 2.1 amounts to replacing the problem (2.5) by the approximate problem. Our next task is naturally to relate the solutions of the two problems. A simple way to relate z^* is to give a bound for either the residual of z_m for problem (2.5) or for the residual of z^* for problem (5.2). The latter case is considered in the next proposition.

PROPOSITION 6. *Let $\gamma_m = \|Q_m A(I - P_m)\|$. Then*

$$\|r_0 - A_m z^*\| \leqq \gamma_m \varepsilon_m. \tag{5.5}$$

[3] Note that here A denotes at the same time a matrix and its associated linear operator.

Proof. We have

$$\begin{aligned}\|r_0 - A_m x^*\| &= \|Q_m(r_0 - AP_m z^*)\| = \|Q_m(Az^* - AP_m z^*)\| \\ &= \|Q_m A(I-P_m)z^*\| = \|Q_m A(I-P_m)(I-P_m)z^*\| \\ &\leqq \gamma_m \varepsilon_m.\end{aligned}$$

□

COROLLARY 1. *Let γ_m be defined as above, and let $\kappa_m = \|(A_m|K_m)^{-1}\|$. Then*

$$\|z_m - z^*\| \leqq (1+\gamma_m^2\kappa_m^2)^{1/2}\varepsilon_m. \tag{5.6}$$

For the proof see the analogous result in [16].

The number $\gamma_m\kappa_m$ plays the role of a condition number for the approximate problem. The corollary therefore means that the error made in approximating z^* by z_m (which is the same as the error $x^* - x_m$) will be of the same order as ε_m provided that the approximate problem is not too badly conditioned. We believe that there is no simple way of bounding either κ_m or γ_m because Q_m is an oblique projector. Thus, γ_m can be bounded as $\gamma_m \leqq \|Q_m\|\|A\|$, where $\|Q_m\|$ is not known, except in the orthogonal projection case, where we have $\|Q_m\| = 1$.

Note that we do not have at our disposal optimality properties such as in the conjugate gradient method. An interesting bound for the residual of z_m for problem (2.5) can also be established by adapting a result shown by Vainikko, see [7], for orthogonal projection methods.

PROPOSITION 7. *Assume that $\tau_m = \min_{x\in AK_m, \|x\|=1} \|Q_m x\|$ is nonzero, and let $c_m = \|Q_m\|$ and $\varepsilon'_m = \min_{z\in K_m} \|r_0 - Az\|$. Then*

$$\varepsilon'_m \leqq \|r_0 - Az_m\| \leqq (1 + c_m/\tau_m)\varepsilon'_m. \tag{5.7}$$

Proof. Consider the restriction $\tilde{Q}_m$ of Q_m to the subspace AK_m. If $\tau_m \neq 0$, then $\tilde{Q}_m$ is a bijection from AK_m to $Q_m AK_m$. Furthermore, from (5.2) we get

$$r_0 = Q_m A z_m,$$

and since Az_m belongs to AK_m, we have

$$Az_m = \tilde{Q}_m^{-1} r_0 = \tilde{Q}_m^{-1} Q_m r_0.$$

Hence,

$$r_0 - Az_m = (I - \tilde{Q}_m^{-1} Q_m) r_0. \tag{5.8}$$

Let x now be any vector of AK_m. Then $(I - \tilde{Q}_m^{-1} Q_m)x = 0$, and hence, (5.8) can also be written as

$$r_0 - Az_m = (I - \tilde{Q}_m^{-1} Q_m)(r_0 - x) \quad \forall x \in AK_m.$$

Thus,

$$\|r_0 - Az_m\| \leqq \|I - \tilde{Q}_m^{-1} Q_m\| \|r_0 - x\|, \qquad x \in AK_m$$

and

$$\|r_0 - Az_m\| \leqq (1 + \|\tilde{Q}_m^{-1}\|\|Q_m\|) \min_{x\in K_m} \|r_0 - Ax\|.$$

Since $\|\tilde{Q}_m^{-1}\| = \tau_m$, this establishes the second part of (5.7). The first part is obvious. □

It is important to notice that in the case where K_m is the Krylov subspace,

$$\varepsilon'_m = \min_{\substack{p\in\mathbb{P}_{m-1} \\ p(0)=1}} \|p(A)r_0\|, \tag{5.9}$$

where $\mathbb{P}_{m-1}$ denotes the space of polynomials of degree not exceeding $m-1$. This quantity is very similar to the quantity ε_m, and the bounds for ε'_m are of the same nature as those for ε_m.

It may seem at first that inequality (5.7) is more powerful than the previous inequality (5.6) because the condition number of the approximate problem does not appear in it. This is not true, however, because the number τ_m^{-1} can be shown to satisfy

$$\tau_m^{-1} = \|A(A_m|_{K_m})^{-1}\|. \tag{5.10}$$

The inverse of $A_m|_{K_m}$ is therefore implicitly involved in the constant τ_m^{-1}, and we have $\tau_m^{-1} \leqq \|A\|\kappa_m$, where κ_m is defined in Corollary 1.

In the various methods that we have described, the left spaces L_m have quite different properties. In the Lanczos method, L_m does not have any relation a priori with the right space K_m. In I.O.M., we have $L_m \subset K_{m+1}$ but the spaces L_m do not form an increasing sequence of spaces, that is, $L_m \not\subset L_{m+1}$. It is therefore interesting to attempt to analyze the effect of the choice of the left space on the convergence properties.

Since ε_m and ε'_m are independent of the space L_m, it is clear that the effect of the choice of the left space on the bounds will be reflected on the constants γ_m in (5.5) and in c_m/τ_m in (5.7).

Noticing that $\gamma_m \leqq \|Q_m\|\|A\|$ and comparing (5.7) and (5.10) shows that the determining factor in the error bounds is the norm $c_m = \|Q_m\|$.

When the quantity c_m is large, then the right side of the error bound (5.7) becomes large, showing that the approximation can be poor.

If we denote by $\theta(L_m, K_m)$ the acute angle between the subspaces L_m and K_m defined by $\cos\theta(L_m, K_m) = \min_{x\in K_m} \cos\theta(x, Q_m x)$, where $\theta(x, y)$ represents the acute angle between x and y, then we have:

PROPOSITION 8.

$$\|Q_m\| = \frac{1}{\cos\theta(L_m, K_m)}. \tag{5.11}$$

We will need the next lemma to prove (5.11).

LEMMA 3.

$$\|Q_m\| = \max_{\substack{x\in L_m \\ x\neq 0}} \|Q_m x\|/\|x\|. \tag{5.12}$$

Proof of lemma. The lemma means that in $\|Q_m\| = \max_{x\neq 0} \|Q_m x\|/\|x\|$ the maximum is reached for $x \in L_m$. This is because if x is decomposed as $x = x_1 + x_2$, where $x_1 \in L_m$, $x_2 \perp L_m$, then

$$\frac{\|Q_m x\|}{\|x\|} = \frac{\|Q_m x_1 + Q_m x_2\|}{\|x_1 + x_2\|} = \frac{\|Q_m x_1\|}{\|x_1 + x_2\|} \leqq \frac{\|Q_m x_1\|}{\|x_1\|}.$$

Proof of proposition.

$$\cos\theta(x, Q_m x) = \frac{|(x, Q_m x)|}{\|x\| \cdot \|Q_m x\|} = \frac{|(x, x) - (x, x - Q_m x)|}{\|x\|\|Q_m x\|};$$

since $x \in L_m$ and $x - Q_m x \perp L_m$, this becomes $\cos\theta(x, Q_m x) = \|x\|/\|Q_m x\|$. Inverting this equality and taking the maximum over all x in L_m gives the result. □

The angle between L_m and K_m is not computable in general. The proposition asserts that the constant $c_m = \|Q_m\|$ is large when the two subspaces L_m and K_m are almost orthogonal. It is difficult to predict a priori whether this will happen for a

particular problem. Practically, it will not be necessary because such a situation can be detected when testing for convergence. For the I.O.M., however, we have the following interesting relation.

PROPOSITION 10. *For the incomplete orthogonalization method,*

$$\|Q_m\| = \frac{1}{\sin \theta(v_{m+1}, K_m)}, \tag{5.13}$$

where

$$\theta(u, K) = \min_{x \in K} \theta(u, x).$$

Proof. First notice that $Q_m w_i = v_i$, where the w_i are defined as in Proposition 5. By Lemma 3, $\|Q_m\| = \max_{x \in L_m, x \neq 0} \|Q_m x\|/\|x\|$. Let $x = \sum \alpha_i w_i$ be any vector in L_m. (Here $\sum = \sum_{i=1}^{m} \sum_{i=1}^{m}$.) Then

$$\frac{\|Q_m x\|}{\|x\|} = \frac{\|\sum \alpha_i v_i\|}{\|\sum \alpha_i w_i\|} = \frac{\|\sum \alpha_i v_i\|}{\|\sum \alpha_i v_i - \sum \alpha_i \delta_i v_{m+1}\|},$$

where $\delta_i = (v_i, V_{m+1})$. If $\sum \alpha_i \delta_i = 0$, then $\|Q_m x\|/\|x\| = 1$, and this is not the maximum unless $\delta_i = 0$, $i = 1, \cdots, m$. So assume $\delta = \sum \alpha_i \delta_i \neq 0$, then

$$\frac{\|Q_m x\|}{\|x\|} = \frac{\left\|\sum \frac{\alpha_i}{\delta} v_i\right\|}{\left\|\sum \frac{\alpha_i}{\delta} v_i - v_{m+1}\right\|}.$$

Therefore

$$\|Q_m\| = \max_{y \in K_m} \frac{\|y\|}{\|y - v_{m+1}\|} = \frac{1}{\min\limits_{y \in K_m} \|y - v_{m+1}\|/\|y\|}.$$

Finally, it can be easily shown that the minimum in the denominator is just $\sin \theta(v_{m+1}, K_m)$. □

The meaning of (5.13) is that the parameter c_m will be large in case v_{m+1} is close to the subspace K_m. When v_{m+1} is orthogonal to K_m, on the other hand, then $\|Q_m\| = 1$, and this corresponds to Arnoldi's method. When v_{m+1} is nearly orthogonal to K_m, we will have $\|Q_m\| \simeq 1$. In practice, it is important that a sufficiently large p be taken such as to make v_{m+1} nearly orthogonal to K_m.

6. Numerical experiments. The numerical experiments described in this section have been run on the CDC CYBER 175 at the University of Illinois at Urbana-Champaign, except where otherwise indicated. The single precision has been used throughout (mantissa of 48 bits).

6.1. Comparison of I.O.M. and Lanczos. We shall first compare the incomplete orthogonalization method (see 3.1) with the Lanczos method (Algorithm 2) on the following example:

$$A = \begin{bmatrix} B & -I & & \\ -I & \ddots & \ddots & \\ & \ddots & \ddots & -I \\ & & -I & B \end{bmatrix} \quad \text{with} \quad B = \begin{bmatrix} 4 & a & & \\ b & \ddots & \ddots & \\ & \ddots & \ddots & a \\ & & b & 4 \end{bmatrix} \tag{6.1}$$

and $a = -1 + \delta$, $b = -1 - \delta$.

B is of dimension 20 and A has dimension $N = 100$. These matrices represent the 5-point discretization of the operator $-\partial^2/\partial x^2 - \partial^2/\partial y^2 + \gamma\,\partial/\partial x$ on a rectangular region.

The right-hand side b is taken to be $b = Ae$, where $e = (1, 1, \cdots, 1)^T$, such that the solution of the system is just e. The parameter δ is taken equal to 0.5 in this first example. Figure 1 compares the convergence of the I.O.M. algorithm with two values of the parameter p, $p = 2$ (upper curve) and $p = 4$ (middle curve), with Algorithm 2 (lower curve). It is seen that the convergence is faster with the Lanczos algorithm. However, each step of the Lanczos algorithm requires two matrix by vector multiplications while I.O.M. requires only one. It should be mentioned that the I.O.M. algorithm applied here is the Algorithm 5 of [16] and that it includes a restraining strategy. (Two restarts have been necessary for $p = 2$, while no restart has been needed when $p = 4$.) Figure 2 shows the same example with $\delta = 10$ treated with Algorithm 2 and I.O.M. (4). Notice the peaks presented by the Lanczos method.

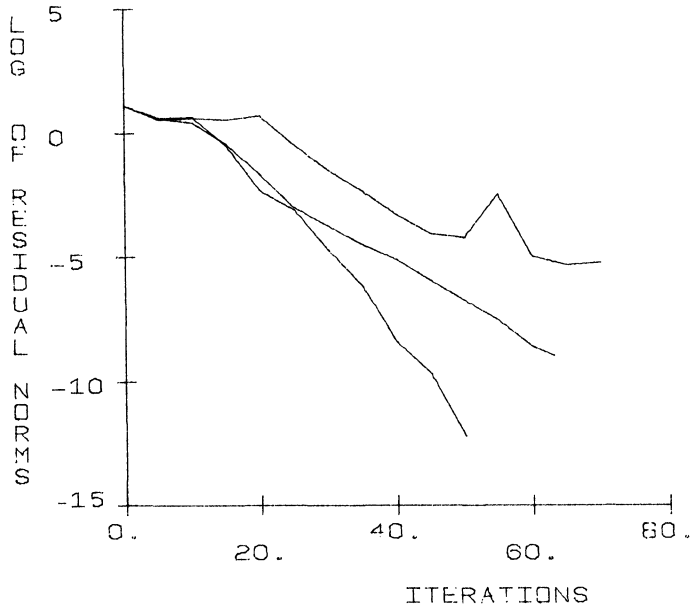

FIG. 1. I.O.M. (2) (*upper curve*), I.O.M. (4) (*middle curve*) *and Lanczos* (*lower curve*).

The Lanczos algorithm often behaves in a way similar to that of Fig. 2, especially in situations where there are large imaginary eigenvalues. It is important to note that these peaks do not seriously affect the overall convergence. When the residual norm increases rapidly at a certain step, it decreases even more rapidly in the following steps.

6.2. I.O.M., Lanczos and ORTHOMIN. Several authors have mentioned that, in the symmetric case, the conjugate residual method (or minimum residual method) and the conjugate gradient method often exhibit a similar convergence behavior. As the next experiment will show, we can make a similar remark for the I.O.M. and the ORTHOMIN-G.C.R. methods. Let A be defined as in § 6.1, with the same right-hand side and the same δ. Figure 3 shows the convergence behaviors of I.O.M. (4) (upper bound), ORTHOMIN (4) (middle curve) and the Lanczos method (lower curve) for this example.

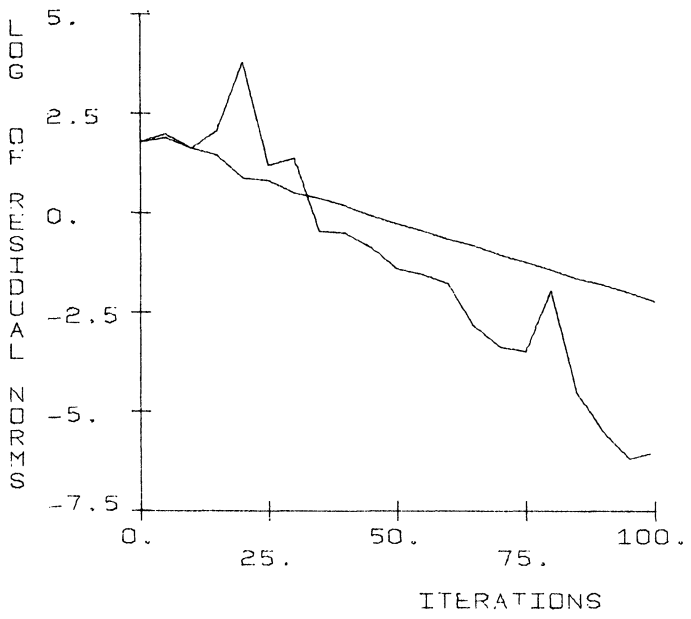

FIG. 2. *Lanczos (lower curve) and* I.O.M. (4) *(upper curve).*

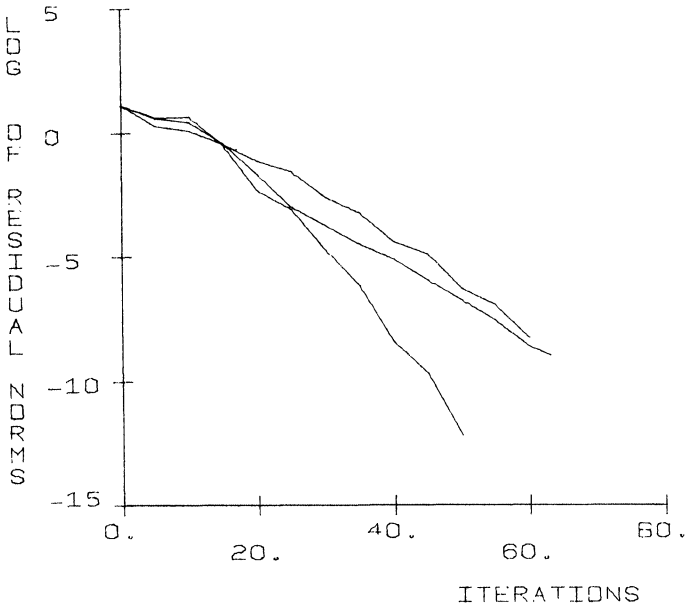

FIG. 3. I.O.M. (4) *(upper curve)*, ORTHOMIN (4) *(middle curve) and Lanczos (lower curve).*

Recall that the ORTHOMIN (p) requires twice as much memory as I.O.M. (p). This means that for this example, I.O.M. is superior if we do not take into account the fact that for the I.O.M. there are some additional I.O. operations (necessary for the preservation of the v_i's until convergence). Algorithm 2 converges much faster than I.O.M. (p) and ORTHOMIN (p) but uses two matrix by vector multiplications. However, it has the advantage that the user supplies no parameter p, which is needed by both I.O.M. (p) and ORTHOMIN (p).

6.3. Complex eigenvalues and the Lanczos method. The purpose of the following example is to show how the behavior of the Lanczos method can vary when the shape of spectrum changes. Let B be the 100×100 block-diagonal matrix with 2×2 blocks c_k defined by

$$c_k = \begin{bmatrix} a_k & b_k \\ b_k & a_k \end{bmatrix}, \qquad k = 1, 2, \cdots, 50,$$

with $a_k = (k/50)\cos\theta$, $b_k = (k/50)\sin\theta$, where θ is a parameter. The eigenvalues of B are $\lambda_k^{\pm} = (k/50)(\cos\theta \pm i\sin\theta)$, where $i = \sqrt{-1}$, $k = 1, 2, \cdots, 50$. Note that the relative distance of the origin from the spectrum does not vary with θ. When θ is small, the eigenvalues are almost real positive and B is almost symmetric. The theory indicates

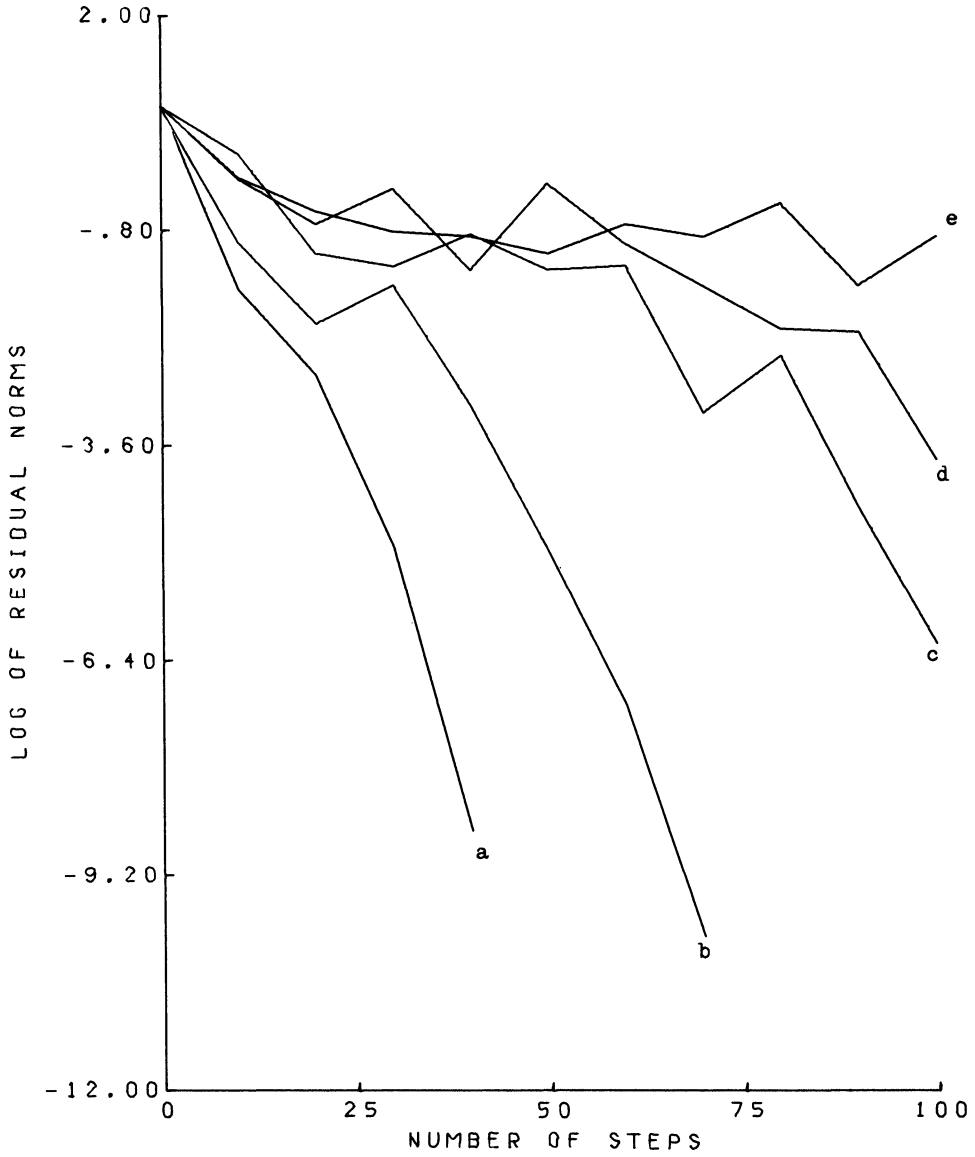

FIG. 4. *Behavior of Lanczos algorithm on different shapes of spectrum.*

that, in that case, a fast convergence can be expected because the distance $\|(I-P_m)x^*\|$ decreases rapidly to zero [16]. When θ increases, the spectrum spreads out in $\mathcal{C}$ and, in that case, the theory does not guarantee a good rate of convergence. Figure 4 shows the behavior of the Lanczos method for the following values of $\delta = \tan\theta$: $\delta = 0.0$ (curve a), $\delta = 0.1$ (curve b), $\delta = 0.4$ (curve c), $\delta = 1$ (curve d), $\delta = 10$ (curve e). The right-hand side is Ae, where $e = (1, 1, \cdots, 1)^T$, and the initial vector is a random vector. The graphs obtained agree with the theoretical predictions. Since the relative separation of the origin and the spectrum is the same, the variation of behavior is solely due to the shape of the spectrum. It is interesting to mention that an experiment with the diagonal matrix $D = \text{diag}\,(a_1, -a_1, a_2, -a_2, \cdots, a_{50}, -a_{50})$, with $a_k = k/50$, yielded a curve very similar to the curve e corresponding to $\delta = 10.0$. Note that as an exception in this section, this experiment has been performed on a VAX 11, using double precision.

6.4. Generalized Hessenberg process. Finally we will describe an experiment with a generalized Hessenberg process belonging to the class of methods outlined in § 3. Let us again take the example given in § 6.1 and consider the generalized Hessenberg process which builds a sequence of vectors v_j as follows

$$h_{j+1,j}v_{j+1} = Av_j - \sum_{i=j-p+1}^{j} h_{ij}v_i, \tag{6.2}$$

where $h_{j+1,j}$ is a normalizing factor for v_{j+1} and where the h_{ij}, $i \neq j+1$ are chosen such as to make p components of v_{j+1} equal to zero. An important question is to determine which components of v_{j+1} should be zero for more efficiency. Several tests have been made, yielding various rates of convergence, depending on the strategies adopted. It was found that for this example a good strategy consists of eliminating in (6.2) the components $j, j-1, \cdots, j-p+1$. A comparison of this strategy, when $p=2$ with I.O.M. (2) and ORTHOMIN (2) is shown in Fig. 5. It can be seen that the convergence

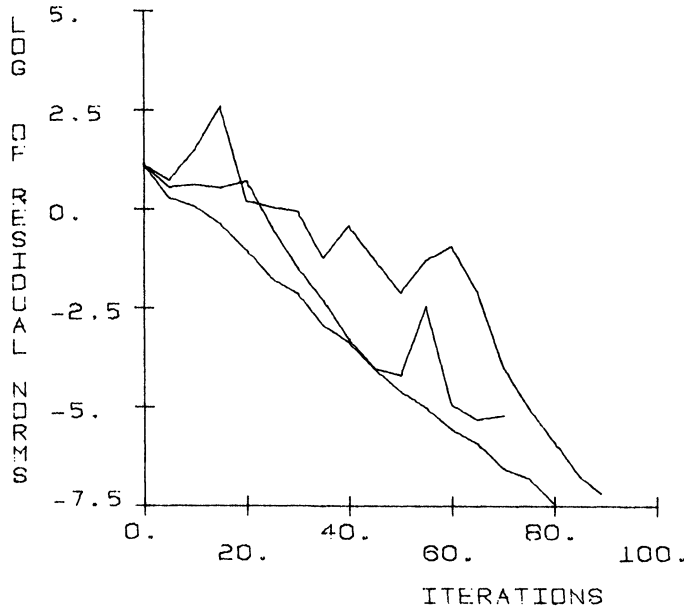

FIG. 5. *A generalized Hessenberg method (upper curve), I.O.M. (2) (middle curve) and* ORTHOMIN (2) *(lower curve).*

of the generalized Hessenberg method compares well with that of I.O.M. (2) or ORTHOMIN (2), and the fact that there are no innerproducts involved in building the v_i's makes the generalized Hessenberg method quite attractive. More general and more powerful strategies remain to be investigated. Another strategy that has appeared effective is to eliminate the components in v_{j+1} corresponding to the large components in the previous v_i's.

A few other experiments dealing with the I.O.M. methods are described in [16]. In particular, a comparison of the iterative Arnoldi algorithm and the bidiagonalization algorithm shows the superiority of orthogonalization techniques over the biadiagonalization methods. Comparisons with the competitive Chebyshev iteration are also reported. They show that I.O.M. is not only slightly faster but overall more reliable than the Chebyshev iteration. This is due to the fact that Chebyshev iteration requires that the origin not be enclosed in an ellipse which contains the spectrum.

REFERENCES

[1] O. AXELSSON, *Conjugate gradient type methods for unsymmetric and inconsistent systems of linear equations*, Linear Alg. and Appl., 29 (1980), pp. 1–16.

[2] C. BREZINSKI, *Padé-Type Approximation and General Orthogonal Polynomials*, Birkhauser-Verlag, Boston, 1980.

[3] S. C. EISENSTAT, H. ELMAN, M. H. SCHULZ AND A. SHERMAN, *Solving approximations to the convection diffusion equation*, Proc. Fifth SPE Symposium on Reservoir Simulation, Denver, Colorado, 1979, pp. 127–132.

[4] R. FLETCHER, *Conjugate gradient methods for indefinite systems*, Proc. of the Dundee Biennial Conference on Numerical Analysis, G. A. Watson, ed., Springer-Verlag, New York, 1975.

[5] W. B. GRAGG, *Matrix interpretation and applications of the continued fraction algorithm*, Rocky Mountain J. Math., 4 (1974), pp. 213–25.

[6] M. HESTENES AND E. STIEFEL, *Methods of conjugate gradients for solving linear systems*, J. Res. N.B.S., 49 (1952), pp. 409–436.

[7] M. A. KRASNOSELSKII, *Approximate Solution of Operator Equations*, Walters-Noordhoff, Groningen, 1972.

[8] C. LANCZOS, *Solution of systems of linear equations by minimized iteration*, J. Res. N.B.S. 49 (1952), pp. 33–53.

[9] J. A. MEIJERINK AND H. Z. VAN DER VORST, *An iterative solution method for linear systems of which the coefficient matrix is a symmetric M-matrix*, Math. Comp., 31 (1977), pp. 148–162.

[10] C. C. PAIGE, *Bidiagonalization of matrices and solution of linear equations*, this Journal, 11 (1974), pp. 197–209.

[11] C. C. PAIGE AND M. A. SAUNDERS, *Solution of sparse undefinite systems of linear equations*, this Journal, 12 (1975), 617–629.

[12] B. N. PARLETT, *The Symmetric Eigenvalue Problem*, Prentice-Hall, Englewood Cliffs, NJ, 1980.

[13] ———, *A new look at the Lanczos algorithm for solving symmetric systems of linear equations*, Linear Alg. and Appl., 29 (1980), pp. 323–46.

[14] B. N. PARLETT AND D. TAYLOR, *The look ahead Lanczos algorithm*, Tech. report, Center for Pure and Applied Math., University of California, to appear.

[15] Y. SAAD, *Variations on Arnoldi's method for computing eigenelements of large unsymmetric matrices*, Linear Alg. and Appl., 34 (1980), pp. 269–295.

[16] ———, *Krylov subspace methods for solving large unsymmetric linear systems*, Math. Comp., 37 (1981), pp. 105–126.

[17] H. L. STONE, *Iterative solution of implicit approximation of multidimensional partial differential equations*, this Journal, 5 (1968), pp. 530–538.

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

[19] J. H. WILKINSON, *The Algebraic Eigenvalue Problem*, Clarendon Press, Oxford, 1965.