A New Look at the Lanczos Algorithm for Solving Symmetric Systems of Linear Equations

B. N. Parlett*
Department of Mathematics and Department of Electrical Engineering and Computer Science, University of California, Berkeley, California

Dedicated to Alston S. Householder
on the occasion of his seventy-fifth birthday.

Submitted by G. W. Stewart

ABSTRACT

Simple versions of the conjugate gradient algorithm and the Lanczos method are discussed, and some merits of the latter are described. A variant of Lanczos is proposed which maintains robust linear independence of the Lanczos vectors by keeping them in secondary storage and occasionally making use of them. The main applications are to problems in which (1) the cost of the matrix-vector product dominates other costs, (2) there is a sequence of right hand sides to be processed, and (3) the eigenvalue distribution of A is not too favorable.

1. INTRODUCTION

One of the most common intermediate tasks in scientific computation is to solve for x the equation

$$Ax = b,$$

where the real $n \times n$ matrix A and one or more n-vectors b are given. This paper concentrates entirely on the case when A is symmetric and n is large. In many, but not all applications A will be sparse, and an elegant way to exploit sparsity is to employ A solely as a linear operator which computes Av for any given vector v.

*Article written while visiting the Computer Science and Systems Division, Harwell, and the Computing Laboratory, Oxford University. Research supported by the Office of Naval Research Contract N00014-76-C-0013.

 0024-3795/80/010323+24$01.75

One of the most popular of these techniques is the method of conjugate gradients introduced by Hestenes and Stiefel [1]. In the same year Lanczos published his method of minimized iterations. Both he and Householder [2] pointed out the intimate connection between the two approaches.

For various reasons both methods languished until Reid [16] demonstrated how effective the conjugate gradient algorithm (called CG hereafter) can be when A is positive definite. Since that time the method has been studied intensively. Attractive features are that no further special properties are needed for A, no acceleration parameters have to be estimated, and only three or four n-vectors need be held in the main store in addition to the demands of the operator A. Another intriguing feature is that the strong effect of roundoff errors on actual implementations does not prevent convergence but merely delays it. Finally, the recent idea of preconditioning can cut down significantly the number of steps needed; see Meijerink and van der Vorst [9], Kershaw [6], and Jennings and Malik [3].

The case when A has eigenvalues of both signs is also important. A valuable contribution to this problem was the algorithm SYMMLQ of Paige and Saunders [13], which was discovered by viewing CG from the Lanczos point of view. The ensuing insights showed how to deal with the indefinite case in a stable manner. However, SYMMLQ is somewhat slower than CG (an extra $5n$ multiplications per step) and is not recommended for the positive definite case. In both cases, if the number of iterations performed is large, then roundoff errors must cause the residual vectors, which would be mutually orthogonal in exact arithmetic, to become linearly dependent to within working accuracy.

The proposed method is, at an abstract level, identical to both CG, SYMMLQ, and the Lanczos algorithm. In exact arithmetic they all produce the same approximate solution at each step. In this paper we explore the consequences of adhering to the following two precepts:

(1) Do not throw away the residual (or Lanczos) vector computed at each step.

(2) Force these vectors to maintain robust linear independence.

Both precepts are obeyed in the technique of full reorthogonalization utilized by Lanczos [8] in 1952; at each step the new Lanczos vector is orthogonalized, by the Gram-Schmidt process, against all the previous Lanczos vectors. This procedure is so very expensive in both storage requirements and execution time that it was abandoned rather soon after its introduction.

The economical way to obey (1) is to use the vast secondary storage device which is usually available. In order to achieve (2) we propose the technique of *selective orthogonalization*, presented by Parlett and Scott [15], which employs the old Lanczos vectors from time to time.

The new algorithm is equally applicable to definite and to indefinite systems. Its goal is to keep the number of calls on the operator A to a minimum. Its cost effectiveness depends on three independent factors: (i) the extent to which the computation of Av dominates a Lanczos step, (ii) the cost of recalling in order the full set of stored Lanczos vectors, and (iii) the number of right hand sides.

It will be some time before the algorithm can be implemented, debugged, refined, compared with other methods, and assessed. This paper presents the background and ideas from which the new techniques sprang.

The notation will follow the conventions popularized by Householder: small Greek letters for scalars, small Latin letters for column vectors, capital Latin letters for matrices. In addition we try to reserve symmetric letters for symmetric matrices. The roundoff unit is always denoted by ε.

2. THE LANCZOS ALGORITHM AND CONJUGATE GRADIENTS

This section concerns the simple Lanczos algorithm in the context of exact arithmetic. The basic equations are presented, as a change from most expositions, in so-called "top down" manner. Rather than specifying an algorithm and then discussing it, our approach is to keep the level of description as high as possible, disclosing practical details only when necessary.

Before beginning it will be convenient to rearrange the problem slightly. When processing a sequence of right hand sides it is usual to have on hand an initial approximation x_a to the true solution $A^{-1}b$; if not, then take $x_a = 0$ in what follows. The problem now is to find the *correction* x_c which must be added to x_a. This is done by computing the initial residual $r_0 \equiv b - Ax_a$ and then solving the nonsingular n-rowed equation

$$Ax_c = r_0. \tag{2.1}$$

Note that x_a and b have faded from the picture.

The Lanczos algorithm (LAN for short) may be described very simply at an abstract level. By the end of the jth step it has constructed a special orthonormal basis for the Krylov subspace $\mathcal{K}^j$ of $\mathbb{R}^n$ defined by

$$\mathcal{K}^j \equiv \mathrm{Span}(r_0, Ar_0, \ldots, A^{j-1}r_0). \tag{2.2}$$

The approximate solution of (2.1) associated with this step is the unique vector x_j in $\mathcal{K}^j$ whose residual $\hat{r}_j$ $(\equiv r_0 - Ax_j)$ is orthogonal to $\mathcal{K}^j$ [the so-called "weak" solution to (2.1) in $\mathcal{K}^j$.] In theory when $\dim \mathcal{K}^j$ reaches its maximum value (usually n), then $\hat{r}_j$ must vanish and x_j actually solves (2.1).

That brief description of LAN suffices for some purposes. In particular the conjugate gradient algorithm computes *the same vector* x_j at step j, and fortunately there are important cases where[1] $\|r_j\|$ is negligible for values of j much smaller than n. See Kaniel [5] for the theory behind the last remark.

LAN and CG part company in the way in which x_j is computed. In particular CG does not have the special Lanczos basis available but instead uses simple recurrences to generate x_j and two auxiliary vectors. The way in which CG is related to LAN is explained very clearly by Paige and Saunders [13, Secs. 2, 3], but that relationship is not needed here. Suffice it to say that CG requires four n-vectors in the fast store and no access to secondary store, unless the computation of Av demands it.

It is now time to give a brief description of one step of LAN. The Lanczos basis consists of the columns of the $n \times j$ matrix $Q=(q_1, q_2, \ldots, q_j)$. It is special because the projection of A onto $\mathcal{K}^j$ is, in terms of Q_j, a *tridiagonal* matrix

$$T_j \equiv \begin{bmatrix} \alpha_1 & \beta_2 & & \\ \beta_2 & \alpha_2 & \beta_3 & \\ & \beta_3 & \cdot & \cdot \\ & & \cdot & \alpha_j \end{bmatrix} = Q_j^* A Q_j.$$

Equation (2.5) below helps to clarify more fully the relation of T_j to A. At the beginning of step j there is on hand Q_{j-1}, T_{j-1} ($q_0=0$, $T_0=0$) a vector r_{j-1}, and $\beta_j \equiv \|r_{j-1}\|$. The jth step first normalizes r_{j-1} to get q_j ($\equiv r_{j-1}/\beta_j$) and then computes Aq_j, α_j ($\equiv q_j^* A q_j$), and a new vector r_j according to the well-known formula

$$r_j \equiv Aq_j - q_j\alpha_j - q_{j-1}\beta_j. \tag{2.3}$$

Finally β_{j+1} ($\equiv \|r_j\|$) is computed and termination tests are made (more on this below). Note that $q_{j-2}, q_{j-3}, \ldots, q_1$ are not needed at step j. The key relationships between the quantities computed by LAN, in exact arithmetic, can be summarized in three equations:

$$I_j - Q_j^* Q_j = 0, \tag{2.4}$$

$$\boxed{A}\ \boxed{Q_j} - \boxed{Q_j}\ \boxed{T_j} = \boxed{\begin{matrix} 0 \\ \vdots \\ \cdot \end{matrix}} = r_j e_j^*, \tag{2.5}$$

$$Q_j^* r_j = 0. \tag{2.6}$$

Here e_j ($e_j^{(j)}$) is the jth column of the $j \times j$ identity matrix I_j.

[1] $\|x\|^2 = x^*x$, and x^* is the transpose of x.

We remark, in passing, that r_j is, to within a scalar factor, one of the two auxiliary vectors used in CG.

The next item is the termination criterion and the computation of x_m, where m is the last Lanczos step taken. First solve for f_m the m-rowed equation

$$T_m f_m = e_1^{(m)} \beta_1. \tag{2.7}$$

The elements $(\phi_1, \ldots, \phi_m)$ of f_m are the coefficients in the formation of x_m from the definition

$$x_m \equiv Q_m f_m. \tag{2.8}$$

That this is the correct formula can be verified by postmultiplying (2.5), for $j = m$, by f_m and using the fact that $r_0 = q_1 \beta_1 = Q_m e_1^{(m)} \beta_1$:

$$\begin{aligned} \hat{r}_m &\equiv A x_m - r_0 = A Q_m f_m - Q_m T_m f_m \\ &= (r_m e_m^*) f_m \\ &= r_m \phi_m. \end{aligned} \tag{2.9}$$

Thus the residual $\hat{r}_m$ associated with x_m is a multiple of r_m and, by (2.6) with $j = m$, is orthogonal to $\mathcal{K}^m$. Moreover

$$\|\hat{r}_m\| = \beta_{m+1} |\phi_m|, \tag{2.10}$$

and so, for each $j = 1, \ldots, m$, *the norm of the jth residual may be computed without forming either x_j or $\hat{r}_j$*. In fact it is not even necessary to compute f_j in order to find ϕ_j. There are various ways of updating ϕ_j from information at the previous step, but the most attractive of them is the recurrence used in the algorithm SYMMLQ by Paige and Saunders [13]. The details are not needed here, and the cost is negligible.

The pieces can now be pulled together into the three stage algorithm for solving (2.1) displayed in Table 1. Some comments on details of this algorithm are given in Sec. 7.

The algorithm LAN is not as elegant as CG (three phases instead of one), but it does offer some advantages for users with limited fast stores, and it is surprising that it was not tried in the middle 1950s.

Here are some of its merits. Note that step 1 is *invariant under translation*. In other words the same sequence of Lanczos vectors are produced from r_0 if $A - \sigma$ is used in place of A for any σ. Consequently LAN is as stable when applied to indefinite problems as it is for the positive definite case, provided only that a stable algorithm is used to solve $Tf = e_1 \beta_1$. [Of course a larger Krylov subspace may be needed to solve $(A - \sigma)x = b$ than to solve $Ax = b$, but that is a theoretical limitation which none of these methods can escape.]

TABLE 1

LAN (simple version)

0. Initialize: $q \leftarrow 0$, $r \leftarrow r_0$, $\rho \leftarrow \|r\|$, $\phi \leftarrow 1$.
1. Until residual norm $(= \rho\phi) <$ tol, repeat the following:
   take a Lanczos step,
   put the new Lanczos vector q into secondary store,
   update ϕ.
2. Solve $Tf = e_1\beta_1$ for f in a stable manner.
3. Recall the Lanczos vectors one by one and accumulate the solution Qf in r.

The jth Lanczos step

	Multiplications/divisions
$r \leftarrow r/\rho$	n
$q \leftarrow q(-\rho)$	n
$\beta_j \leftarrow \rho$	0
$q \leftarrow Ar + q$	"Aq"
swap q and r	0
$\alpha \leftarrow q^*r$	n
$r \leftarrow r - q\alpha$	n
$\alpha_j \leftarrow \alpha$	0
$\rho \leftarrow \|r\|$	n

The stable analogue of CG for indefinite problems is the interesting algorithm SYMMLQ of Paige and Saunders [13]. It avoids using secondary storage, but boosts the operation count by $5n$ at each step and requires two more n-vectors of fast storage. Consequently Paige does not recommend it for problems which are known to be positive definite. However, LAN can do the work of both CG and SYMMLQ.

Finally there is the task of treating a sequence of right hand sides. If Q_j and T_j are saved, then they can assist substantially in these subsequent calculations. As an extreme example, suppose that a new pair, x_a and r_0, are given and r_0 happens to lie in $\mathcal{K}^j$; then $A^{-1}r_0$ can be computed with almost no further calls on the big matrix A. Details are given in Sec. 6.

Rather than compare LAN with CG, this paper focuses on modifications of LAN which prevent the loss of linear independence which, in practice, may tarnish both CG and LAN.

3. THE WAY IN WHICH ORTHOGONALITY IS LOST

Roundoff errors cause the computed Lanczos vectors to lose not just orthogonality but linear independence (to working accuracy) by a certain

step which depends on A and on the roundoff unit. In practice LAN will not always terminate by the time $j = n$, and it will take at least as many steps as would be required in exact arithmetic, sometimes no more and sometimes twice as many. A similar retardation troubles CG and SYMMLQ. For problems in which the matrix-vector product Av is dominant, this blemish in the algorithm is serious.

The modifications to LAN, described in Sec. 4, spring from an understanding of the precise way in which orthogonality is lost. The honor for unraveling this process goes to Paige [10] in his doctoral thesis. His results are not well known, at least not to those who work on $Ax = b$, and full accounts of them are not readily available; [14] and [17] are examples. Consequently this section supplies the necessary details of his interesting theory. A full account is given in [18].

It is often stated that the cause of this orthogonality loss is persistent, albeit modest cancellation in the execution of the statement

$$r_j \leftarrow Aq_j - q_j\alpha_j - q_{j-1}\beta_j.$$

Now $\beta_{j+1} = \|r_j\|$ and $\|Aq_j\|^2 = \beta_j^2 + \alpha_j^2 + \beta_{j+1}^2$, since an adequate level of *local* orthogonality is maintained in the simple Lanczos algorithm. Thus cancellation can be measured by the ratio $\beta_{j+1}^2/(\beta_j^2 + \alpha_j^2 + \beta_{j+1}^2)$.

By monitoring these ratios one discovers that they rarely drop below $\frac{1}{16}$. Indeed, if cancellation were the *main* cause of orthogonality loss, then there would be little choice but to use full reorthogonalization to remedy the situation, just as Lanczos did in 1950. No, there is another mechanism at work (roundoff) which imposes a very definite pattern on the way the Lanczos vectors tilt towards each other.

To explain what happens it is necessary to introduce quantities which are not of direct interest when solving $Ax = b$. Consider the eigenvalues and orthonormal eigenvectors of T_j:

$$T_j s_i^{(j)} = s_i^{(j)} \theta_i^{(j)}, \qquad i = 1, \ldots, j. \tag{3.3}$$

These eigenvectors change at each step, but when there is no danger of confusion the superscript j will be dropped. From these one defines the *Ritz vectors*

$$y_i^{(j)} = Q_j s_i^{(j)}, \qquad i = 1, \ldots, j, \tag{3.4}$$

which form an alternative to the Lanczos vectors as an orthonormal basis of $\mathcal{K}^j$.

The pairs $(\theta_i^{(j)}, y_i^{(j)})$, $i = 1, \ldots, j$, are approximate eigenpairs of A, but some are much better than others. The quality of the approximation is best measured by the associated residual norm. Postmultiply (2.5) by s_i to find

$$\begin{aligned} \|Ay_i - y_i\theta_i\| &= \|(AQ_j - Q_jT_j)s_i\| \\ &= \|r_j(e_j^* s_i)\| \\ &= \beta_{j+1}|s_{ji}| \boxed{\equiv \beta_{ji}}. \end{aligned} \tag{3.5}$$

As in the previous section, it happens that the norm of a certain residual can be computed without computing the residual vector itself. The quantities β_{ji}, which involve the bottom elements of the eigenvectors of T_j, play an important role in explaining the behavior of LAN in practice. Clearly the convergence of a Ritz pair $(\theta_i^{(j)}, y_i^{(j)})$ to an eigenpair of A is signaled by small values of β_{ji}.

From now on let Q_j, T_j, etc. denote the computed values of these quantities. When roundoff contaminates each arithmetic operation, the orthogonality relation (2.4) fails completely after a certain number of steps, but the other fundamental equation (2.5) is only slightly blurred whatever the value of j. In fact

$$AQ_j - Q_jT_j = r_je_j^* + F_j, \tag{3.6}$$

where F_j is the *local* error matrix and its jth column accounts for the roundoff errors which occur during step j. It is not difficult to show that, for all k,

$$\|f_k\| = \|F_je_k\| = O(\varepsilon\|A\|),$$

where ε is the unit roundoff quantity. In brief, $\|F_j\|$ remains tiny. Paige's insight can now be stated properly.

The key quantity is the angle between q_{j+1} and $\mathcal{K}^j$ or, equivalently, $\max|u^*q_{j+1}|$ over all u in $\mathcal{K}^j$. Paige shows that one of the Ritz vectors is essentially the closest vector in $\mathcal{K}^j$ to q_{j+1}. In practice the $y_i^{(j)}$ are defined by (3.4) and they are not the true Rayleigh Ritz approximations from $\mathcal{K}^j$, because Q_j is not orthonormal.

THEOREM (Paige). *Consider the simple Lanczos algorithm implemented in floating point arithmetic with unit roundoff* ε. *Then at step* j

$$y_i^{(j)*}q_{j+1} = \gamma_{ii}^{(j)}/\beta_{ji}, \qquad i = 1, \ldots, j,$$

where

$$|\gamma_{\mu\nu}^{(j)}| = O(\varepsilon\|A\|).$$

Furthermore, omitting (j),

$$y_i^* y_k(\theta_i - \theta_k) = \gamma_{ii}\left(\frac{s_{ik}}{s_{ii}}\right) - \gamma_{kk}\left(\frac{s_{ii}}{s_{ik}}\right) - (\gamma_{ik} - \gamma_{ki}).$$

The proof is buried in Paige [10]. A simpler version is given in [18]. What matters is that the matrix Γ_j is $O(\varepsilon\|A\|)$ but, incidentally, $\Gamma_j = S_j^* K_j S_j$, where K_j is the upper triangular part of the skew matrix $F_j^* Q_j - Q_j^* F_j$.

It turns out that many important questions about the Lanczos algorithm are answered by the theorem. Note first that *in practice*

$$\|Ay_i - y_i\theta_i\| = \|r_j s_{ji} + F_j s_i\| \leqslant \beta_{ji} + \|F_j\|. \tag{3.8}$$

Consequently the β_{ji} are accurate measures of the corresponding residual norms until a norm reaches the roundoff level. In what follows a Ritz pair $(\theta_i^{(j)}, y_i^{(j)})$ will be called *good* if β_{ji} is below some designated threshold such as $\sqrt{\varepsilon}\,\|A\|$. Another point to be made is that although Q_j, T_j, and r_j refer to computed quantities, the variables θ_i, s_i, and y_i are purely notional and there is no artificiality in assuming that S_j is exactly orthogonal, so that $I_j - S_j^* S_j = 0$.

A convenient measure of the *level of mutual orthogonality* among the $\{q_i\}$ is $\|I - Q_j^* Q_j\|$. Note that the $\{y_i^{(j)}\}$ have exactly the same level, since

$$\|I - Q_j^* Q_j\| = \|S_j^*(I - Q_j^* Q_j)S_j\| = \|I - Y_j^* Y_j\| \leqslant j \max|y_i^* y_k|. \tag{3.9}$$

Moreover, the orthogonality vector $Q_j^* q_{j+1}$ satisfies

$$\|Q_j^* q_{j+1}\| = \|S_j Y_j^* q_{j+1}\| = \|Y_j^* q_{j+1}\| \leqslant \sqrt{j}\ \max_i |y_i^{(j)} q_{j+1}|. \tag{3.10}$$

Thus the quantities appearing in Paige's theorem have a direct bearing on orthogonality loss among the Lanczos vectors.

The first deduction is usually summarized as: loss of orthogonality is equivalent to convergence of a Ritz pair. In other words, a value of $|y_i^* q_{j+1}|$ close to 1 corresponds to a value of β_{ji} close to $\varepsilon\|A\|$. More precisely, when $\|y_i^{(j)}\| \doteq 1$ for all i, Paige's theorem says that q_{j+1} is tilted more towards the most accurate Ritz vector than towards any other.

The second deduction concerns the $\{ y_i^{(j)}, i=1,\ldots,j\}$. Recall that $\Sigma_{i=1}^{j} s_{ji}^2 = 1$. Thus there are always some $|s_{ji}|$ greater than or equal to $1/\sqrt{j}$. Initially the $|s_{ji}|$ will remain close to their mean, say $|s_{ji}| > \frac{1}{10}\sqrt{j}$, and Paige's theorem shows that the $y_i^{(j)}$ belonging to different Ritz values retain a good level of orthogonality. Moreover the Kaniel theory of Krylov subspaces shows that it is almost impossible to have very close Ritz values for small values of j. On the contrary the $\theta_i^{(j)}$ tend to converge (as j increases) to the outer eigenvalues in A's spectrum.

The picture changes abruptly as soon as one $|s_{ji}|$ drops sharply below all the others. Then all the other $y_\nu^{(j)}$, $\nu \neq i$, tilt towards $y_i^{(j)}$ and q_{j+1} is *tilted sharply towards* $y_i^{(j)}$ *while retaining its previous level of orthogonality to all the other* $y_\nu^{(j)}$, which also retain their previous level of mutual orthogonality.

Other strange phenomena occur as j increases. These are compatible with, but not deducible from, Paige's theorem, and they are not directly relevant to the next section. Suffice it to say that some time (say p steps) after a particular β_{ji} sinks to $O(\varepsilon\|A\|)$, there will occur $\beta_{j+p,l}$ and $\beta_{j+p,k}$ both $O(\varepsilon\|A\|)$ with

$$\theta_l^{(j+p)} = \theta_k^{(j+p)} = \theta_i^{(j)} = \lambda$$

to within $O(\varepsilon\|A\|)$, despite the fact that λ is a simple isolated eigenvalue of A with eigenvector z. It follows that

$$y_l^{(j+p)} = Q_{j+p} s_l = Q_{j+p} s_k = y_k^{(j+p)} = z,$$

despite the fact that s_l and s_k are orthogonal. Thus $Q_{j+p}(s_l - s_k) = O(\varepsilon)$ and so Q_{j+p} must have linearly dependent columns to within working accuracy.

The Lanczos algorithm can run on indefinitely, but for large j the Ritz vectors include many redundant copies of eigenvectors of A which belong to the outer eigenvalues in the spectrum.

Table 2, from Scott [17], illustrates some of the behavior described above. The example was chosen to force an early loss of independence.

4. SELECTIVE ORTHOGONALIZATION

This section describes some modifications to LAN. Recall that the jth Lanczos step produces a vector r_j $(= Aq_j - q_j\alpha_j - q_{j-1}\beta_j)$ and normalizes it to get q_{j+1}. The new version modifies r_j before normalizing it, and for purposes of description some notation is needed to distinguish r before modification from r afterwards. To facilitate subsequent analysis we reserve r_j for the final version and use r_j' for the original. Thus now

$$r_j' \equiv Aq_j - q_j\alpha_j - q_{j-1}\beta_j - f_j, \tag{4.1}$$

where f_j accounts for roundoff.

TABLE 2

$A = \text{diag}(0, 1\times10^{-3}, 2\times10^{-3}, 3\times10^{-3}, 4\times10^{-3}, 1)$

$q_1 = (1,1,1,1,1,1)^*/\sqrt{6}\,, \qquad \varepsilon = 10^{-7}$

j	α_j	β_j	$\|I - Q_j^* Q_j\|$
1	.166833	.3726035	0
2	.83333665	.0003464	2×10^{-7}
3	.0002004	.0003094	6×10^{-4}
4	.1464297	.3532944	5×10^{-2}
5	.9998344	.0001098	1×10^{0} ←

Selected information on Ritz values

j	i	θ_i	$\|y_i\|$	β_{ji}	$\|y_i^* q_{j+1}\|$
3	1	0.587×10^{-4}	0.9999745	$.225\times10^{-3}$	$.4297\times10^{-3}$
	2	0.3415×10^{-3}	1.000025	$.2228\times10^{-3}$	$.4297\times10^{-3}$
	3	1.0000	1.000001	$.48\times10^{-7}$	.923981 ←
5	1	0.157×10^{-4}	1.000496	$.486\times10^{-4}$	$.34\times10^{-2}$
	2	0.2001×10^{-3}	1.000477	$.778\times10^{-4}$	$.22\times10^{-2}$
	3	0.3845×10^{-3}	0.999509	$.486\times10^{-4}$	$.35\times10^{-2}$
	4	0.9999996	0.75043 ←	$.414\times10^{-4}$	$.70\times10^{-2}$
	5	1.000000	1.148672 ←	$.681\times10^{-5}$	$.70\times10^{-2}$

$y_1^{(3)} y_3^{(3)} = 0.2\times10^{-3}$ $\qquad |y_i^{(5)*} y_k^{(5)}| < 10^{-3},\ (i,k) \neq (4,5)$

$y_2^{(3)*} y_3^{(3)} = -0.2\times10^{-3}$

$y_1^{(3)*} y_2^{(3)} = 0.1\times10^{-6}$ $\qquad y_4^{(5)*} y_5^{(5)} = 0.5\times10^{-1}$

The idea is quite simple. For appropriate values of j and i (depending on j) the algorithm computes $y_i^{(j)}$ and orthogonalizes r_j' against it to get

$$r_j \equiv r_j' - y_i^{(j)} \xi_i^{(j)}, \tag{4.2}$$

where

$$\xi_i^{(j)} \equiv \frac{y_i^{(j)*} r_j'}{\| y_i^{(j)} \|^2}. \tag{4.3}$$

The hope is that $\max_\nu |y_\nu^* r_j|$ is significantly less than $\max_\nu |y_\nu^* r_j'|$. Will the hope be fulfilled, and if so, how much will the modification cost? The answers depend on how j and i are selected, so the rest of this section is devoted to that crucial topic.

The level of orthogonality among the q_i, $i = 1, \ldots, j$, is measured by

$$\kappa_j \equiv \|I - Q_j^* Q_j\|. \tag{4.4}$$

The modification is based on the goal of keeping $\kappa_j \leqslant \kappa$ for some given parameter κ and all j. If $\kappa = n\varepsilon$, then a very costly form of full reorthogonalization takes place;[2] there is no reason, other than simplicity, for keeping the bound on κ_j constant; both $\sqrt{j}\,\kappa$ and $j\kappa$ are valid alternatives. Arguments in [17] and [18] suggest that the choice $\kappa = \sqrt{\varepsilon}$ is best for the computation of eigenvalues. For solving $Ax = b$ the most desirable values of κ are not known.

At the end of the jth Lanczos step there are on hand T_j, r_j', and β_{j+1}' ($\equiv \|r_j'\|$). Loss of orthogonality is reflected in the size of $\|Q_j^* r_j'\|$, but we do not want to compute $Q_j^* r_j'$. Instead we note that

$$\|Q_j^* r_j'\| \leqslant \sqrt{j}\ \max_i |y_i^{(j)*} r_j'| \qquad \text{by (3.10)}$$

$$= \sqrt{j}\ \max_i \frac{|\gamma_{ii}^{(j)}|}{\beta_{ji}'} \qquad \text{by Paige's theorem.}$$

Paige has proved that $\gamma_{ii}^{(j)} = O(j\varepsilon\|A\|)$, but in practice no counterexample is known to the stronger assertion

$$|\gamma_{ii}^{(j)}| \leqslant \varepsilon\|A\|,$$

and the new algorithm uses this uniform bound to reduce the estimation of $Q_j^* r_j'$ to checking the computable numbers β_{ji}' ($= \beta_{j+1}' s_{ji}$), namely

$$\|Q_j^* r_j'\| \leqslant \frac{\sqrt{j}\ \varepsilon\|A\|}{\min_i \beta_{ji}'}. \tag{4.5}$$

The computation of all the s_{ji}, $i = 1, \ldots, j$, is an $O(j^2)$ process. Fortunately as j increases, the s_{ji} change in an orderly way, and it is only necessary to compute a few of them, thus reducing the cost of testing orthogonality loss to $O(j)$ multiplications per step. This important point is amplified below.

At each step we define the subset of Ritz vectors, called the *threshold* vectors, which make too small an angle with r_j'. More precisely, let $\angle(u, v)$ denote the acute angle between u and v, and define

$$\gamma(j) \equiv \{\, i : \cos \angle(\, y_i^{(j)}, r_j') \geqslant \kappa/\sqrt{j}\ \}.$$

We say that $\theta_i^{(j)}$ is a *threshold* Ritz value if $i \in \gamma(j)$, and similarly for the Ritz

[2] If $\kappa > 1$, then no orthogonalizations will occur.

vectors. The modified algorithm LANSO (LAN with *selective orthogonalization*) replaces step 1 by

1. Until the residual norm $(\rho\phi) < \text{tol}$, repeat the following:
   determine $\gamma(j)$,
   for each $i \in \gamma(j)$ compute y_i and orthogonalize r against y_i,
   take a Lanczos step,
   put the new q into secondary store,
   update ϕ.

Another idea is needed to keep down the cost of LANSO: *if* $i \notin \gamma(j-1)$ *and* $i \in \gamma(j)$, *then the algorithm forces* $i \in \gamma(j+1)$. This device helps to keep subsequent Lanczos vectors orthogonal to *good* vectors (those for which $\beta_{ji} < \sqrt{j}\,\varepsilon\|A\|/\kappa$). The intuitive reason is as follows: good Ritz vectors are almost eigenvectors of A. If two successive Lanczos vectors are orthogonal to an eigenvector, then so are *all* subsequent Lanczos vectors because of the three term recurrence. In symbols, let $Az = z\lambda$, $z^* q_{j-1} = z^* q_j = 0$. Then

$$z^* q_{j+1} = \beta_{j+1}^{-1} z^* r_j,$$

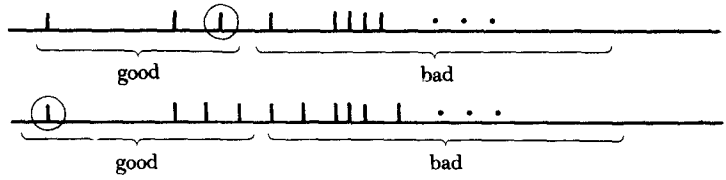
$$= \beta_{j+1}^{-1}(z^* A q_j - z^* q_j \alpha_j - z^* q_{j-1}\beta_j)$$

$$= \beta_{j+1}^{-1}(\lambda z^* q_j - z^* q_j \alpha_j - z^* q_{j-1}\beta_j) = 0.$$

The last line used the symmetry of A to get $z^* A = \lambda z^*$. In practice, because of roundoff, a good Ritz vector will not remain orthogonal to all subsequent q's, and after a while it will become a threshold vector once again. The important result is that for each i, $i \in \gamma(j)$ for *few* j, and furthermore a three term recurrence can be used to detect when an already good y_i is once again a threshold vector. The details are relegated to Sec. 5.

The situation as regards orthogonality can be described conveniently in terms of the Ritz *values* $\theta_i^{(j)}$, $i = 1, \ldots, j$. For each j they are divided into two subsets, the *good* and the *bad* ($\beta_{ji}' > \varepsilon\|A\|\sqrt{j}\,/\kappa$). By Paige's theorem r_j' is adequately orthogonal to all bad Ritz vectors, and by the Kaniell theory the bad values are in the interior of T_j's spectrum and are in the majority until j is very close to n. The good Ritz values are, in general, the outer eigenvalues, but only a subset of them are threshold values. As we have seen, the threshold Ritz values are either $\theta_i^{(j)}$ which have just become good for the first time, or else already coverged θ's whose Ritz vectors have bent too much towards the Krylov subspace and must be banished again.

The two configurations are illustrated below at the left end of the spectrum; the threshold value is circled:



Occasionally an interior Ritz value can converge before its neighbors, so the algorithm must check β'_{jk} for all bad Ritz values from time to time. However, for most j it is only necessary to compute s_{jk} for values of $\theta_k^{(j)}$ just adjacent to the good region. For many values of j ($<0.66n$), $\gamma(j)$ is empty.

Now we turn to the orthogonalization process itself. The natural sequence is as follows:

(i) Compute and store s_i (eigenvector of T_j) for each $i \in \gamma(j)$.

(ii) Fetch the Lanczos vectors back from secondary store one by one, and accumulate the $y_i = Q_j s_i$, $i \in \gamma(j)$.

(iii) For each $i \in \gamma(j)$ repeat

$$\xi_i \leftarrow \frac{r^* y_i}{\| y_i \|^2},$$

$$r \leftarrow r - y_i \xi.$$

(iv) Update bookkeeping records and store ξ_i and j together with $\theta_i^{(j)}$.

Suppose that m n-vectors are available for holding the Ritz vectors. Let $|\gamma(j)|$ be the number of elements in $\gamma(j)$. If $|\gamma(j)| \leqslant m$, the sequence given above will suffice; if not, then the whole sequence must be repeated until $\gamma(j)$ is exhausted. However, each sequence involves the recall of all the Lanczos vectors, and this cost must be weighed against the burden of keeping m n-vectors in fast store. Our present value of m is 2. It should be noted that on the second of a pair of steps at which orthogonalization occurs, the costly items (i) and (ii) are not needed.

The arithmetic operation count for each $i \in \gamma(j)$ is

(i) $4j$ (since $\theta_i^{(j)}$ is known) for s_i,
(ii) jn for y_i,
(iii) $2n$ (the algorithm assumes $\| y_i \| = 1$) for orthogonalization,
(iv) 0 for bookkeeping.

This is approximately half the cost of Lanczos's full reorthogonalization at step j.

The frequency with which orthogonalizations occur is problem dependent, but it should be noted that they occur only when actually needed to maintain the desired level of orthogonality.

5. SOME ANALYSIS

A few pieces of the algorithm will be analyzed here because the results throw light on the rather intuitive description given in the previous section.

A. *Preservation of angles*

Let $i \in \gamma(j)$. The algorithm replaces r_j' with r_j, and certainly $y_i^* r_j = O(\varepsilon \|A\|)$, by construction of r_j. However, it seems possible that the angles between r_j' and all the other Ritz vectors may have been damaged in the process. To see the connection we consider a specific step j and drop the superscript. From (4.2)

$$r_j = r_j' - \sum_{\nu \in \gamma(j)} y_\nu \xi_\nu + O(\varepsilon \|A\|), \tag{5.1}$$

whence

$$y_k^* r_j = y_k^* r_j' - \sum_\nu (y_k^* y_\nu) \xi_\nu. \tag{5.2}$$

In exact arithmetic both $(y_k^* y_\nu)$ and ξ_ν vanish, but the same effect occurs in practice provided that their *product* is tiny.

The factor ξ_ν is not small. For simplicity we make the reasonable assumption that $\|y_i\| = 1 + O(\varepsilon)$ for all i. Then, from (4.3),

$$\begin{aligned} |\xi_\nu| &= |y_\nu^* r_j'| \\ &= \cos \angle (y_\nu, r_j') \beta_{j+1}', \qquad \text{since} \quad \beta_{j+1}' = \|r_j'\|, \\ &\geqslant \beta_{j+1}' \kappa / \sqrt{j}, \qquad \text{by definition of } \gamma(j) \text{ in Sec. 4.} \end{aligned} \tag{5.3}$$

The other factor $(y_k^* y_\nu)$ is bounded by κ_j, since

$$\|I - Y_j^* Y_j\| = \|S_j (I - Q_j^* Q_j) S_j^*\| = \|I - Q_j^* Q_j\| = \kappa_j. \tag{5.4}$$

Moreover, by the second assertion of Paige's theorem, this bound on $y_k^* y_\nu$ is realistic whenever $\nu \in \gamma(j)$ and y_k is a bad Ritz vector, because then the ratios $|s_{jk}/s_{j\nu}|$ are maximal. Consequently some elements in the sum of (5.2) will probably exceed

$$\kappa_j\left(\frac{\beta'_{j+1}\kappa}{\sqrt{j}}\right) = \beta'_{j+1}\kappa_j\left(\frac{\kappa}{\sqrt{j}}\right). \tag{5.5}$$

Does this matter? Note that for some values of k we have $|s_{jk}| \geqslant 1/\sqrt{j}$ and so the first term on the right in (5.2) is less than $\sqrt{j}\,\varepsilon\|A\|$ for these k. If $\kappa^2 > \varepsilon$, then the sum will dominate the first term in (5.2), and the orthogonalization process will degrade these particular angles $\angle(y_k, r'_j)$. Parlett and Scott [15] take a strict attitude and use $\kappa = \sqrt{\varepsilon}$.

At the other extreme the large values among the $|y^*_k r'_j|$, $k \notin \gamma(j)$, are already close to $\beta'_{j+1}\kappa_j$ and are unharmed by the orthogonalization, since $\kappa < 1$.

B. *The equivalent perturbation*

Selective orthogonalization necessitates a change in the governing equation (3.6). To derive the new one we begin from

$$AQ_j - Q_j T_j = r'_j e_j^* + F_j \tag{5.6}$$

and use (5.1) to eliminate the overwritten vector r'_j, finding

$$AQ_j - Q_j T_j - \left(\sum_{\nu \in \gamma(j)} \xi_\nu^{(j)} y_\nu\right) e_j^* = r_j e_j^* + F_j. \tag{5.7}$$

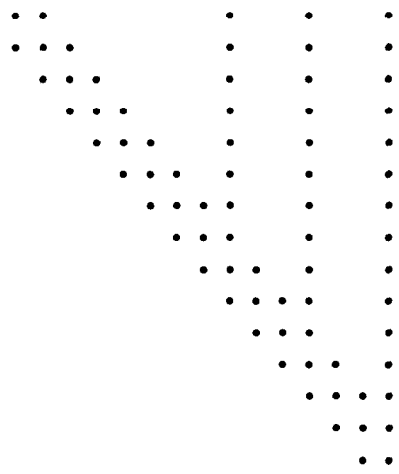
The small device of bringing the new term to the left side of the equation permits a simple interpretation of the effect of orthogonalization. Substitute $y_\nu = Q_j s_\nu$ in (5.7) to obtain the new equation

$$AQ_j - Q_j T'_j = r_j e_j^* + F_j, \tag{5.8}$$

where

$$T'_j = T_j + \left(\sum_{\nu \in \gamma(j)} \xi_\nu^{(j)} s_\nu^{(j)}\right) e_j^*. \tag{5.9}$$

The effect of selective orthogonalization is simply to replace T_j by T_j'. After a sequence of orthogonalizations T_j' has the form indicated below:



It may be verified that for $i \in \gamma(j)$, the eigenpairs (θ_i, s_i) of T_j coincide with corresponding eigenpairs of T_j' to working accuracy, whereas the pairs corresponding to bad Ritz values θ_k are significantly changed.

C. *Solving* $T'f' = e_1\beta_1$

Step 2 of LAN requires the solution of $T_m f = e_1\beta_1$, where m is the final value of j. However, in the light of (5.9) it would be better to replace f by the solution f' of $T_m' f' = e_1\beta_1$. This can be done without imposing serious extra demands for storage, because the vectors $s_\nu^{(j)}$ which characterize the rank one perturbations of T_j are easily recomputed when needed. All that is needed is to store $\xi_\nu^{(j)}$ and j, along with θ_ν, $\nu \in \gamma(j)$, at each orthogonalization.

Here is the reconstruction of f' from f in a typical situation. Let

$$T_m = \begin{bmatrix} T_j & \vdots & \\ \cdots & \bullet & \cdots \\ & \vdots & \hat{T}_j \end{bmatrix}, \qquad \bullet = \beta_{j+1}, \quad T_j s_i = s_i \theta_i. \tag{5.10}$$

Recall that $\xi_i^{(j)} = y_i^* r_j'$ and, by Paige's theorem,

$$\xi_i^{(j)}\beta_{j+1}' = y_i^* r_j' \beta_{j+1}' = \gamma_i^{(j)} s_{ji}.$$

Thus

$$|\xi_i^{(j)}\beta'_{j+1}s_{ji}| = |\gamma_{ii}^{(j)}| \leqslant \varepsilon\|A\|. \tag{5.11}$$

Now define an m-vector $\bar{s}_i$ by

$$\bar{s}_i \equiv \begin{pmatrix} s_i^{(j)} \\ 0 \end{pmatrix}. \tag{5.12}$$

Then

$$\begin{aligned} T_m\bar{s}_i\xi_i &= \bar{s}_i\xi_i\theta_i + e_{j+1}^{(m)}(\beta'_{j+1}s_{ji}\xi_i) \\ &= \bar{s}_i\xi_i\theta_i + O(\varepsilon\|A\|), \qquad \text{by (5.11)}, \end{aligned}$$

and we may say that $\bar{s}_i\xi_i$ is an approximate eigenvector of T_m. The perturbed matrix is

$$T'_m = T_m + \bar{s}_i e_j^{(m)*}\xi_i^{(j)}. \tag{5.13}$$

Let $f = (\phi_1, \ldots, \phi_m)^*$ satisfy $T_m f = e_1^{(m)}\beta_1$; then it can be verified that with

$$f' \equiv f - \bar{s}_i\eta, \qquad \eta \equiv \frac{\xi_i^{(j)}\phi_j}{\theta_i + \xi_i^{(j)}s_{ji}}, \tag{5.14}$$

we have

$$T'_m f' = e_1^{(m)}\beta_1 + O(\varepsilon\|A\|). \tag{5.15}$$

No requirement is made in the derivation of (5.15) that T_j be tridiagonal. It could be replaced by any matrix for which (θ_i, s_i) was an eigenpair. It follows that f may be modified, as in (5.14), for a *sequence* of orthogonalizations provided only that these are made in the natural order of increasing j.

D. The return of banished Ritz vectors

Let $y_i^{(j)}$ be a good Ritz vector:

$$Ay_i - y_i\theta_i = u_i, \qquad \|u_i\| = O(\varepsilon\|A\|). \tag{5.16}$$

Roundoff error will introduce small components of y_i into all Lanczos

vectors after orthogonalization, and eventually these will be amplified unacceptably. Let $|y_i^* q_j| \leqslant \tau_j$, suppressing the dependence on i. Then premultiplying (5.6) by y_i^* yields

$$\begin{aligned} y_i^* r_j' &= y_i^* A q_j - y_i^* q_j \alpha_j - y_i^* q_{j-1} \beta_j + y_i^* f_j \\ &= (\theta_i - \alpha_j) y_i^* q_j - \beta_j y_i^* q_{j-1} + u_i^* q_j + y_i^* f_j, \end{aligned} \tag{5.17}$$

where we have invoked (5.16). Hence

$$|y_i^* q_{j+1}'| \leqslant \frac{|\theta_i - \alpha_j| \tau_j + \beta_j \tau_{j-1} + O(\varepsilon \|A\|)}{\beta_{j+1}'}.$$

The algorithm drops the $O(\varepsilon \|A\|)$ and uses

$$\tau_{j+1} \equiv \frac{|\theta_i - \alpha_j| \tau_j + \beta_j \tau_{j-1}}{\beta_{j+1}'}. \tag{5.18}$$

This recurrence is updated, for each good i, at each step. Whenever τ_{j+1} exceeds $\kappa/\sqrt{j}$, then i is put into $\gamma(j)$. After orthogonalization τ is set back to ε.

6. PROCESSING A SEQUENCE OF RIGHT HAND SIDES

Consider the case when the first Lanczos sequence (or run) stops at step j with a negligible residual. In the fast store are the tridiagonal T_j and the vector r_j which would have been normalized to become q_{j+1} at the next step. In secondary store is Q_j. At this point a new right hand side b and a new initial approximation x_a (possibly 0) are presented. As before, the goal is to solve for the correction x_c the equation

$$A x_c = r_0 \ (\equiv b - A x_a). \tag{6.1}$$

The vector b fades from the scene, but $\|b\|$ is kept to test the negligibility of the residuals from subsequent approximations. Now we begin by describing the approximate solution of (6.1) in the context of exact arithmetic.

The first move is to compute β_{j+1} $(\equiv \|r_j\|)$ and q_{j+1} $(= r_j/\beta_{j+1})$. Then Q_j is brought back from secondary storage in order to calculate the orthogonal decomposition

$$r_0 = Q_j w_j + q_{j+1} \omega_{j+1} + \bar{r}_0, \tag{6.2}$$

where $w_j = Q_j^* r_0$ and $Q_{j+1}^* \bar{r}_0 = 0$. It is convenient to consider the contributions to x_c from the three terms in (6.2) one by one.

(1) $Q_j w$. It is only necessary to solve $T_j f^{(1)} = w_j$ for $f^{(1)}$. The contribution from Span Q_j is

$$x_1 \equiv Q_j f^{(1)}. \tag{6.3}$$

There is no need to compute x_1 at this point, but its residual is

$$\begin{aligned} r_0^{(1)} &\equiv r_0 - Ax_1 \\ &= \left(\bar{r}_0 + q_{j+1}\omega_{j+1} + Q_j w_j\right) - \left(Q_j T_j f^{(1)} + q_{j+1}\beta_{j+1} e_j^* f^{(1)}\right) \\ &= \bar{r}_0 + q_{j+1}(\omega_{j+1} - \beta_{j+1}\phi_j), \qquad \phi_j \equiv e_j^* f^{(1)}. \end{aligned} \tag{6.4}$$

No calls on A are needed in the formation of x_1. Note that the effective component of q_{j+1} is changed from ω_{j+1} to $\chi_{j+1} = \omega_{j+1} - \beta_{j+1}\phi_j$.

(2) $q_{j+1}\chi_{j+1}$. If $|\chi_{j+1}|$ is not negligible, then the next contribution x_2 can be calculated by simply continuing the previous Lanczos sequence after setting the residual norm to $|\chi_{j+1}|$. However, there is one new feature, because the algorithm must include the component of $\bar{r}_0$ in the direction of each new Lanczos vector. The extra calculations at i $(i > j)$ are

$$\begin{aligned} &\omega_i = q_i^* \bar{r}_0, \\ &\bar{r}_0 \leftarrow \bar{r}_0 - q_i \omega_i, \\ &\text{update the residual norm} \quad \chi_i. \end{aligned} \tag{6.5}$$

The sequence continues until χ_i is negligible; when $i = k$ say. The approximate solution from Span Q_{k-1} is

$$x_2 = Q_{k-1} f^{(2)}, \tag{6.6}$$

where $T_{k-1} f^{(2)} = w_{k-1}$, and this supersedes (6.3). Again there is no need to accumulate x_2, but its residual is the latest $\bar{r}_0$ which satisfies $Q_k^* \bar{r}_0 = 0$.

(3) $\bar{r}_0$, $\bar{\beta}_{k+1} = \|\bar{r}_0\|$. If $\bar{\beta}_{k+1}$ is not negligible relative to $\|b\|$, then a new Lanczos run is needed with starting vector $q_{k+1} = \bar{r}_0 / \bar{\beta}_{k+1}$. The new feature here is that, although $Q_k^* q_{k+1} = 0$,

$$Q_k^* A q_{k+1} \neq 0, \qquad \text{in general.} \tag{6.7}$$

Thus the new Krylov subspace need not be orthogonal to the previous one. However, by the fundamental property of Krylov subspaces,

$$\text{if} \quad Q_k^* v = 0, \ v \text{ in } \mathcal{K}^n, \quad \text{then} \quad Q_{k-1}^* A v = 0. \tag{6.8}$$

Thus the Krylov subspace generated by q_{k+1} will be orthogonal to Span Q_{k-1} *but not to* q_k. In order to keep *all* the q's mutually orthogonal it suffices to modify the basic Lanczos step to be

$$\begin{aligned} r_i &= Aq_i - q_i\alpha_i - q_{i-1}\beta_i - q_k\gamma_i, \qquad i > k+1, \\ r_{k+1} &= Aq_{k+1} - q_{k+1}\alpha_{k+1} - q_k\gamma_{k+1}, \end{aligned} \tag{6.9}$$

where $\gamma_i = q_k^*(Aq_i)$. Note that, by definition, $\beta_{k+1} = 0$. However $\bar{\beta}_{k+1}$ $(= \|\tilde{r}_0\|)$ may be stored in β_{k+1} for convenience. Strictly speaking $\text{Span}(q_{k+1}, q_{k+2}, \ldots, q_i)$ is not a Krylov subspace, but it is the projection of one onto the subspace orthogonal to q_k.

The most serious burden incurred by the modification is the need to keep q_k in the fast store, for $i > k$. The new elements γ_i can be put into a vector g. The new recurrence (6.9) changes the basic equation (2.5) into

$$AQ_i - Q_i \left[\begin{array}{c|c} T_1 & e_k g^* \\ \hline g e_k^* & T_2 \end{array}\right] = r_i e_i^*, \tag{6.10}$$

where T_1 $[k \times k]$ and T_2 $[(i-k) \times (i-k)]$ are symmetric tridiagonal matrices and $g^* = (\gamma_{k+1}, \ldots, \gamma_i)$. Let us denote the rank two modification of the tridiagonal by $\hat{T}_i$.

Suppose that by step $i = l$ the associated residual is negligible and the Lanczos iteration finally stops. The final component of x_c is given by

$$x_3 = Q_l f^{(3)}, \tag{6.11}$$

where $f^{(3)}$ satisfies

$$\hat{T}_l f^{(3)} = e_{k+1}\bar{\beta}_{k+1}. \tag{6.12}$$

Recall that $f^{(2)}$, in (6.6), is a k-vector, whereas $f^{(3)}$ is an l-vector. To compute the approximation to x_c produced by neglecting residuals as mentioned above it is only necessary to form

$$f = \begin{pmatrix} f^{(2)} \\ 0 \end{pmatrix} + f^{(3)} \tag{6.13}$$

and bring the columns of Q_l back to main memory to accumulate $Q_l f$. The neglected residual is

$$q_k \beta_k (e_k^* f^{(2)}) + r_l (e^* f^{(3)}). \tag{6.14}$$

The number of calls on A is $l-j$, and there will be cases when this number is small. However, when the new b is orthogonal to $\operatorname{Span} Q_j$ it is more efficient to switch off the mechanism which keeps $\operatorname{Span}(q_{k+1}, \ldots, q_l)$ orthogonal to $\operatorname{Span} Q_j$ and then ignore Q_j.

Two important details are (i) the updating of the residual norm at each step from $i = k+1, \ldots, l$, and (ii) the computation of $f^{(3)}$ from (6.12). Both tasks can be accomplished by using well-known techniques concerning rank one and rank two modifications. For example, let $f^{(3)}$ be partitioned into $f_1 \oplus f_2$ so that

$$\begin{aligned} T_1 f_1 + e_k g^* f_2 &= 0, \\ g e_k^* f_1 + T_2 f_2 &= e_1 \bar{\beta}. \end{aligned} \tag{6.15}$$

On eliminating f_1 from the second equation one finds

$$(T_2 - g g^* / \delta_1) f_2 = e_1 \bar{\beta}, \tag{6.16}$$

where δ_1 ($\equiv 1/e_k^* T_1^{-1} e_k$) will be known from the first Lanczos run. To solve (6.16) *two* tridiagonal systems must be solved, namely

$$T_2 u = e_1 \bar{\beta}, \qquad T_2 v = g. \tag{6.17}$$

Then f_2 is a linear combination of u and v since

$$\begin{aligned} f_2 &= (I - v g^* / \delta_1)^{-1} T_2^{-1} e_1 \bar{\beta} \\ &= (I + v g^* / \delta) u \\ &= u + v (g^* u) / \delta, \end{aligned} \tag{6.18}$$

where

$$\delta = \delta_1 - g^* v. \tag{6.19}$$

A third tridiagonal system must be solved to recover f_1. It is not difficult to update the quantities $g^* u$ and δ in (6.18) from step to step. The tridiagonal

systems should be solved by scaled QR factorization to ensure stability in the indefinite case, but we omit the details. The point is that the modifications employed to process a second right hand side do not complicate the algorithm significantly, although they do have to be combined with the modifications induced by selective orthogonalization—the computation of Ritz vectors, for example.

It is apparent that an extra n-vector must be held in fast storage for *each* previous Krylov subspace if they are to be kept mutually orthogonal. Reid has suggested that if the new right hand sides are variations on the initial one, then it may be preferable to discard all Lanczos vectors except for the first (big) sequence, i.e. discard $q_{k+1},\ldots,q_l$. In this way the extra storage remains at one n-vector.

7. COMMENTS ON LAN

(1) The α's and β's, the nonzero elements of T, could be put into secondary storage along with the q's and brought back for step 2 of LAN. However, the modifications proposed in Secs. 4 and 6 demand T at every step, and for the sake of consistency we have ignored this feature.

(2) The algorithm for the jth Lanczos step incorporates a trick, given by Paige [12], which reduces the fast storage requirements, other than for A, to *two* n-vectors. The price to be paid is that the two vectors must be swapped in the middle of the step and the basic computation $r = Aq_j - q_j\alpha_j - q_{j-1}\beta_j$ must be done as

$$r_j = (Aq_j - q_{j-1}\beta_j) - q_j\alpha_j.$$

If the user supplied program is written to compute $(A-\sigma)q$ rather than Aq for given σ and q, then an attractive alternative for the heart of the Lanczos step is to compute $\sigma = \text{trace}(A)/n$ initially and then

$q \leftarrow (A-\sigma)r + q,$
$\alpha \leftarrow q^* r,$
swap q and r,
$r \leftarrow r - q\alpha,$
$\alpha_j \leftarrow \alpha + \sigma.$

(3) It is well known that it is not necessary to normalize the Lanczos vectors to have unit length. The resulting matrix T will not be symmetric, but the device does save n divisions (n multiplications) per step. We follow Paige and Saunders in preferring to pay the small extra cost and enjoy the simplifications which accrue from a symmetric T.

The author has enjoyed helpful discussions with David Scott, John Reid, Olof Widlund, and Iain Duff.

REFERENCES

1 M. R. Hestenes and E. Stiefel, Methods of conjugate gradients for solving linear systems, *J. Res. Nat. Bur. Standards* 49:409–436 (1952).
2 A. S. Householder, *The Theory of Matrices in Numerical Analysis*, Blaisdell, 1964, pp. 139–141.
3 A. Jennings and G. A. Malik, Partial elimination, *J. Inst. Math. Appl.* 20:307–316 (1977).
4 W. Kahan and B. N. Parlett, How far can you go with the Lanczos algorithm? in *Sparse Matrix Computations* (Bunch and Rose, Eds.), Academic, 1976.
5 S. Kaniel, Estimates for some computational techniques in linear algebra, *Math. Comp.* 20:369–378 (1966).
6 D. S. Kershaw, The incomplete Cholesky-conjugate gradient method for the iterative solution of systems of linear equations, *J. Computational Phys.* 26:43–65 (1977).
7 C. Lanczos, An iteration method for the solution of the eigenvalue problem of linear differential and integral operators, *J. Res. Nat. Bur. Standards* 45:255–282 (1950).
8 C. Lanczos, Solution of systems of linear equations by minimized iterations, *J. Res. Nat. Bur. Standards* 49:33–53 (1952).
9 J. A. Meijerink and H. A. van der Vorst, An iterative solution method for linear systems of which the coefficient matrix is a symmetric *M*-matrix, *Math. Comp.* 31:148–162 (1977).
10 C. C. Paige, The computation of eigenvalues and eigenvectors of very large sparse matrices, Ph. D. Thesis, Univ. of London, 1971.
11 C. C. Paige, Computational variants of the Lanczos method for the eigenproblem, *J. Inst. Math. Appl.* 10:373–381 (1972).
12 C. C. Paige, Bidiagonalization of matrices and solution of linear equations, Technical report No. CS295, Computer Science Department, Stanford Univ., 1972; *SIAM J. Numer. Anal.* 11, No. 1 (Mar. 1974).
13 C. C. Paige and M. A. Saunders, Solution of sparse indefinite systems of linear equations, *SIAM J. Numer. Anal.* 12:617–629 (1975).
14 C. C. Paige, Error analysis of the Lanczos algorithm for tridiagonalizing a symmetric matrix, *J. Inst. Math. Appl.* 18:341–349 (1976).
15 B. N. Parlett and D. S. Scott, The Lanczos algorithm with selective orthogonalization, *Math. Comp.*, 33:217–238 (1979).
16 J. K. Reid, On the method of conjugate gradients for the solution of large sparse systems of linear equations, in *Proceedings of a Conference on "Large Sparse Sets of Linear Equations,"* Academic, 1971.
17 D. S. Scott, Analysis of the symmetric Lanczos process, Memo UCB/ERL M78/40, Electronics Research Lab., Univ. of California, Berkeley, Jun. 1978.
18 B. N. Parlett, *The Symmetric Eigenvalue Problem*, Prentice Hall, NJ, 1980.

Received 4 December 1978