SIAM J. NUMER. ANAL.
Vol. 13, No. 5, October 1976

A POLYNOMIAL EXTRAPOLATION METHOD FOR FINDING LIMITS AND ANTILIMITS OF VECTOR SEQUENCES*

S. CABAY AND L. W. JACKSON†

Abstract. Given a sequence of vectors $\{x^{(0)}, x^{(1)}, x^{(2)}, \cdots\}$, which may be convergent or divergent and which is governed by $x^{(i+1)} = Mx^{(i)} + g$, where M and g may be unknown, an extrapolation method is derived for finding either the limit or antilimit of the sequence. The extrapolation method is intended for those problems where the numerical rank of M is small, or where M has only a few dominant eigenvalues. An error analysis is given; in particular, it is shown that the extrapolation method is stable provided that the eigenvalues of M are bounded away from 1.

The method is compared with the epsilon algorithm and the vector-epsilon algorithm. Numerical evidence is provided showing that the method obtains the same accuracy as the epsilon algorithms but with approximately half the cost.

It is also shown that the extrapolation method is competitive with the Woodbury method for solving the system $(I-M)x = g$, where M is known.

1. Introduction.

Given a sequence of vectors $\{x^{(0)}, x^{(1)}, x^{(2)}, \cdots\}$ governed by a linear iteration of the form

$$x^{(i+1)} = Mx^{(i)} + g, \qquad i \geqq 0, \tag{1.1}$$

where M and g may be unknown, an extrapolation method is derived for finding the limit or the antilimit of the sequence. Shanks [11] gives the definition of antilimit. For the purpose of this paper, it is assumed that $(I-M)^{-1}$ exists so that the limit (antilimit) is the unique solution of the equation

$$x = Mx + g. \tag{1.2}$$

The basis of our method is a formula discovered by Schmidt [10]. However, the ideas and motivation in Schmidt's paper are different from those contained herein. Except for Wynn [15] who used Schmidt's ideas for the development of the epsilon algorithm, it appears that Schmidt's formulae have not been used, probably because they call for the evaluation of a characteristic polynomial.

Let

$$\Delta \equiv x - x^{(0)} \tag{1.3}$$

and

$$\Delta^{(i)} \equiv x^{(i+1)} - x^{(i)}, \qquad i > 0. \tag{1.4}$$

The extrapolation formula to be developed is

$$\Delta = \sum_{i=0}^{s} \left(\sum_{j=i+1}^{s+1} p_j \right) \Delta^{(i)} / p(1), \tag{1.5}$$

* Received by the editors November 12, 1974, and in final revised form January 5, 1976.

† Department of Computing Science, University of Alberta, Edmonton, Alberta, Canada, T6G 2H1. This work was supported by National Research Council of Canada under grants NRC A8316 and NRC A8315.

where

$$p(y)=\sum_{j=0}^{s+1} p_j y^j \quad \text{with } p_{s+1}=1 \tag{1.6}$$

is the monic minimal polynomial that annihilates $\Delta^{(0)}$. Of course, such a formula would be useless if it were necessary to compute the coefficients of p accurately or if the degree $s+1$ of p were large.

The error analysis given in this paper shows that it is not necessary to compute the coefficients of p accurately. In fact the degree of the calculated polynomial need not equal the degree of the minimal polynomial. All that is necessary is that a certain residual be small; that is, the calculated polynomial should almost annihilate $\Delta^{(0)}$. Such a polynomial can be found cheaply for problems where M has small numerical rank or when M has only a few dominant eigenvalues.

The extrapolation method is intended primarily for those problems where the iterates $x^{(0)}, x^{(1)}, x^{(2)}, \cdots$ only (not M and g) are known. The computational cost of methods requiring the construction of M and g from the iterates is prohibitive. Algorithms not requiring this construction include the epsilon, the $q-d$, the η and the g algorithms. Since comparative studies of these various algorithms already appear in the literature (see, for example, [16]), and furthermore, since the epsilon algorithms have received wider attention, in this paper the epsilon algorithms are the ones chosen for comparison with the polynomial extrapolation method.

For problems in which M and g are known, Woodbury's method is a popular and attractive alternative to the polynomial extrapolation method. For this reason in § 9 comparisons are made with Woodbury's method.

2. Motivation and derivation of method.

Equations (1.1), (1.2), (1.3) and (1.4) yield

$$\Delta^{(i+1)}=M\Delta^{(i)} \tag{2.1}$$

and

$$(I-M)\Delta=\Delta^{(0)}. \tag{2.2}$$

Let

$$S_p(y)\equiv\sum_{i=0}^{s}\left(\sum_{j=i+1}^{s+1} p_j\right) y^i. \tag{2.3}$$

By computing $(I-M)S_p(M)$, one obtains the identity

$$(I-M)S_p(M)=p(I)-p(M). \tag{2.4}$$

If $p(M)$ annihilates $\Delta^{(0)}$, then

$$p(I)\Delta^{(0)}=(I-M)S_p(M)\Delta^{(0)}, \tag{2.5}$$

which, using (2.1) and (2.2), yields

$$p(1)\Delta=S_p(M)\Delta^{(0)}=\sum_{i=0}^{s}\left(\sum_{j=i+1}^{s+1} p_j\right)\Delta^{(i)}. \tag{2.6}$$

Observe that $p(1)\neq 0$, since otherwise M has an eigenvalue at unity and thus $(I-M)^{-1}$ does not exist. Therefore Δ, and consequently the solution x to (1.2), can be found after $s+2$ iterations of (1.1) provided that the polynomial (1.6) annihilating $\Delta^{(0)}$ is available. Of course, this conclusion is independent of the convergence or divergence of the iteration (1.1), making the extrapolation useful for finding antilimits as well as limits.

3. Existence of almost-annihilating polynomials.

In practice the quantities $\Delta^{(i)}$ are not available. Instead a sequence of vectors $\Delta^{*(i)}$ $(i=0, 1, 2, \cdots)$, is found such that

$$\Delta^{*(0)}=\Delta^{(0)}+\delta^{(0)}, \tag{3.1a}$$

$$\Delta^{*(i+1)}=M\Delta^{*(i)}+\delta^{(i+1)}, \tag{3.1b}$$

where $\|\delta^{(0)}\|/\|\Delta^{*(0)}\|<\varepsilon_1$, $\|\delta^{(i+1)}\|/(\|M\|\cdot\|\Delta^{*(i)}\|)<\varepsilon_1$ and ε_1 is small.

No attempt shall be made to produce a good approximation to the minimal polynomial $p(\cdot)$. Rather, we seek an almost-annihilating polynomial

$$\alpha(y)=\sum_{i=0}^{s^*+1}\alpha_i y^i, \qquad \alpha(1)\neq 0, \quad \alpha_{s^*+1}=1, \tag{3.2}$$

such that

$$\sum_{i=0}^{s^*+1}\alpha_i\Delta^{*(i)}=\delta, \tag{3.3}$$

where δ is relatively small. In this section, some conditions are given under which almost-annihilating polynomials $\alpha(\cdot)$ exist.

If M has rank s, then there exist p_j such that

$$\sum_{j=0}^{s+1}p_j\Delta^{(j)}=0. \tag{3.4}$$

From (3.1) it quickly follows that

$$\Delta^{*(j)}=\Delta^{(j)}+\sum_{i=0}^{j}M^i\delta^{(j-i)}, \qquad j\geqq 0, \tag{3.5}$$

and thus from (3.4) it follows immediately that $\|\sum_{j=0}^{s+1}p_j\Delta^{*(j)}\|$ is relatively small. That is, in this case, the minimum degree s^*+1 of an almost-annihilating polynomial is at most $s+1$.

On the other hand, if M has numerical rank s, then a perturbed matrix $M+\delta M$, where $\|\delta M\|/\|M\|$ is small, has rank s (see Noble [8]). Thus there exists p_j such that

$$\sum_{j=0}^{s+1}p_j(M+\delta M)^j\Delta^{(0)}=0, \tag{3.6}$$

which implies $\|\sum_{j=0}^{s+1}p_j\Delta^{(j)}\|$ is relatively small. If we use (3.5), it again follows that $\|\sum_{j=0}^{s+1}p_j\Delta^{*(j)}\|$ is relatively small and the minimum degree of an almost-annihilating polynomial is at most $s+1$.

Finally, if M has s dominant eigenvalues, there exist α_j such that $\|\sum_{j=0}^{s+t} \alpha_j \Delta^{(j)}\|$ is relatively small, where t is the number of iterations required to damp out the effects of the small transients (i.e., the small eigenvalues). The reader is referred to Shanks [11] for a discussion of this matter. As before, equality (3.5) can be used to show that $\|\sum_{j=0}^{s+t} \alpha_j \Delta^{*(j)}\|$ is relatively small. The minimum degree s^*+1 of an almost-annihilating polynomial is therefore at most $s+t$.

4. A practical algorithm and error analysis. The problem of calculating α is that of choosing s^* and α_i so that the norm of

$$\delta \equiv \sum_{j=0}^{s^*+1} \alpha_j \Delta^{*(j)}, \qquad \alpha_{s^*+1} = 1, \tag{4.1}$$

is small.

Two of the most promising techniques for obtaining α are:

(i) the method of Krylov (cf. Wilkinson [13, pp. 369–377]), and

(ii) singular value decomposition techniques (cf. Golub et al. [1]–[5] or Stewart [12]) in which the calculation of α and s^* may be regarded as a least squares problem.

In the first approach use is made of the fact that if $\Delta^{(0)}, \Delta^{(1)}, \cdots$ is a Krylov sequence defined by

$$\Delta^{(i)} = M\Delta^{(i-1)}, \qquad \text{given } \Delta^{(0)}, \tag{4.2}$$

then there exists an integer s^* such that $\Delta^{(0)}, \Delta^{(1)}, \cdots, \Delta^{(s^*)}$ form a linearly independent set of vectors and $\Delta^{(s^*+1)}$ is contained in the linear span of this set. These facts immediately lead to Krylov's method of finding the coefficients of the annihilating polynomial α. Krylov's method simply involves generating an approximate Krylov sequence (3.1) and then solving (4.1) with δ set to zero. Of course, s^* is not known a priori, and one should use a variant of triangular decomposition that uses the $\Delta^{*(i)}$ as each one becomes available (cf. Wilkinson [13, p. 370]).

Theoretically, it is quite easy to find s^*. However, in actual practice Krylov's method is beset with several difficulties. Often the equations become very ill-conditioned. Furthermore, it is often difficult to determine the grade of $\Delta^{(0)}$. As pointed out by Wilkinson, these two problems often cause disastrous results if one is attempting to calculate eigenvalues via the calculation of annihilating polynomials.

For our purposes, because we need not calculate eigenvalues, neither problem matters! We cannot emphasize this point too strongly. Both from a theoretical point of view and from much numerical evidence, what is important is that each $\delta^{(i)}$ be small and that the residual δ^* defined by

$$\delta^* \equiv \sum_{j=0}^{s^*+1} \beta_j \Delta^{*(j)} \tag{4.3}$$

be small. Here β is the computed approximation to α. Provided that δ^* is small, β need not be a good approximation to p. In fact, in several of our experiments, the polynomial $\beta(\cdot)$ did not resemble the minimal polynomial $p(\cdot)$ in that both the coefficients and the zeros of each polynomial were different.

In spite of the above negative aspects of Krylov's method, it appears sufficiently good for the purposes at hand. In order to see why, one must bear in mind that Gaussian elimination with partial pivoting usually produces small residuals (see Wilkinson [14]).

The second alternative, that of using least squares techniques to approximate α, has not been tested. Preliminary analysis indicates that least squares techniques will be more expensive. However, they should produce slightly smaller residuals, and a case can therefore be made for using these methods.

The following theorem gives an error estimate for the method and when properly interpreted indicates the critical calculations of the extrapolation algorithm.

THEOREM 4.1. *Let*

$$\beta(y) \equiv \sum_{i=0}^{s^*+1} \beta_i y^i, \qquad \beta(1) \neq 0, \quad \beta_{s^*+1} = 1, \tag{4.4}$$

$$c_i \equiv \sum_{j=i+1}^{s^*+1} \beta_j, \qquad i = -1, 0, \cdots, s^*, \tag{4.5}$$

and

$$\Delta^* \equiv \sum_{i=0}^{s^*} c_i \Delta^{*(i)} / \beta(1). \tag{4.6}$$

If

$$\sum_{i=0}^{s^*+1} \beta_i \Delta^{*(i)} = \delta^*, \tag{4.7}$$

then

$$\|\beta(1)(\Delta - \Delta^*)\| \leqq (s^*+3)\|\beta\|_1 \cdot \|(I-M)^{-1}\| \cdot \max\{\|\delta^{(i)}\|, \|\delta^*\|\}, \tag{4.8}$$

where

$$\|\beta\|_1 = \sum_{i=0}^{s^*+1} |\beta_i|. \tag{4.9}$$

Proof. From

$$\Delta^{*(i+1)} = M\Delta^{*(i)} + \delta^{(i+1)}$$

it follows that

$$c_i(\Delta^{*(i+1)} - \Delta^{*(i)}) = (M-I)c_i\Delta^{*(i)} + c_i\delta^{(i+1)}. \tag{4.10}$$

Using summation by parts and the definition of Δ^* yields

$$\sum_{i=0}^{s^*+1} \beta_i \Delta^{*(i)} - \beta(1)\Delta^{*(0)} = (M-I)\beta(1)\Delta^* + \sum_{i=0}^{s^*} c_i \delta^{(i+1)}. \tag{4.11}$$

Using equation (2.2), one obtains

$$\beta(1)\Delta^{(0)} = (I-M)\beta(1)\Delta. \tag{4.12}$$

Adding equations (4.11) and (4.12) with $c_{-1}=\beta(1)$ and $\Delta^{*(0)}-\Delta^{(0)}=\delta^{(0)}$ yields

$$(I-M)\beta(1)(\Delta-\Delta^*)=\delta-\sum_{i=-1}^{s^*} c_i\delta^{(i+1)}, \tag{4.13}$$

and thus

$$\begin{aligned}\|\beta(1)(\Delta-\Delta^*)\| &\leqq \|(I-M)^{-1}\| \left\|\delta^*-\sum_{i=0}^{s^*+1}\left(\sum_{j=i}^{s^*+1}\beta_j\right)\delta^{(i)}\right\| \\ &\leqq \|(I-M)^{-1}\|\left(\|\delta^*\|+\sum_{i=0}^{s^*+1}\|\beta\|_1\|\delta^{(i)}\|\right).\end{aligned} \tag{4.14}$$

Using $\beta_{s^*+1}=1$ we find that $\|\beta\|_1\geqq 1$, and hence (4.8) follows. Q.E.D.

COROLLARY 4.1. *Let the minimal polynomial of M be*

$$p(y)=\sum_{j=0}^{s+1} p_j y^j, \qquad p(1)\neq 0. \tag{4.15}$$

If the conditions of the theorem hold true, then

$$\|p(1)\beta(1)[\Delta-\Delta^*]\|\leqq(s^*+3)\|p\|_1\cdot\|\beta\|_1\cdot\sum_{i=0}^{s}\|M\|^i\cdot\max\{\|\delta^{(i)}\|,\|\delta\|\}, \tag{4.16}$$

where

$$\|p\|_1=\sum_{i=0}^{s+1}|p_i|. \tag{4.17}$$

Proof. The proof follows immediately by using in (4.13) the fact that if $p(y)$ is the minimal polynomial of M, then

$$p(1)(I-M)^{-1}=\sum_{i=0}^{s}\left(\sum_{j=i+1}^{s+1}p_j\right)M^i. \tag{4.18}$$

5. The epsilon algorithms.

An alternative method for obtaining x from a finite number of iterates $x^{(i)}$ has been derived by Schmidt [10]. It follows from his derivation that if the degree of minimal polynomial of M annihilating $\Delta^{(0)}$ is $s+1$, then

$$x=\frac{\begin{vmatrix} x^{(0)} & x^{(1)} & \cdots & x^{(s)} \\ \Delta^{(0)} & \Delta^{(1)} & \cdots & \Delta^{(s)} \\ \vdots & \vdots & & \vdots \\ \Delta^{(s-1)} & \Delta^{(s)} & & \Delta^{(2s-1)}\end{vmatrix}}{\begin{vmatrix} 1 & 1 & & 1 \\ \Delta^{(0)} & \Delta^{(1)} & \cdots & \Delta^{(s)} \\ \vdots & \vdots & & \vdots \\ \Delta^{(s-1)} & \Delta^{(s)} & \cdots & \Delta^{(2s-1)}\end{vmatrix}}, \tag{5.1}$$

where the determinants in (5.1) are applied componentwise to the vectors $x^{(i)}$ and

$\Delta^{(i)}$. If the degree of the minimal polynomial of M annihilating $\Delta^{(0)}$ is not $s+1$, but instead if s is the smallest integer for which there exists α_i such that

$$\sum_{i=0}^{s+1} \alpha_i \Delta^{(i)} = \delta, \tag{5.2}$$

where $\|\delta\|$ is small, then

$$\begin{vmatrix} \Delta^{(0)} & \Delta^{(1)} & \cdots & \Delta^{(s)} \\ \Delta^{(1)} & \Delta^{(2)} & \cdots & \Delta^{(s+1)} \\ \vdots & \vdots & & \vdots \\ \Delta^{(s)} & \Delta^{(s+1)} & \cdots & \Delta^{(2s)} \end{vmatrix} \tag{5.3}$$

is small for all components of $\Delta^{(i)}$. It is then Schmidt's claim that s is the smallest integer for which right-hand side of (5.1) is a good approximation to x. (The numerical experiments of §§ 7 and 8 tend to support this claim.)

Wynn [15] has shown that the right-hand side of (5.1) can be computed recursively by the epsilon algorithm as follows: Let $\varepsilon_{-1}^{(i)} = 0$ and $\varepsilon_0^{(i)} = x^{(i)}$ $(i = 0, 1, 2, \cdots)$, and define $\varepsilon_{j+1}^{(i)}$ $(j = 0, 1, 2, \cdots)$ componentwise by

$$\varepsilon_{j+1}^{(i)} = \varepsilon_{j-1}^{(i+1)} + [\varepsilon_j^{(i+1)} - \varepsilon_j^{(i)}]^{-1}, \qquad i = 0, 1, 2, \cdots. \tag{5.4}$$

Wynn proves that $\varepsilon_{2s}^{(0)}$ is equal to the right-hand side of (5.1) for any sequence $x^{(0)}$, $x^{(1)}, \cdots, x^{(2s)}$. Thus if s is the smallest integer for which (5.2) holds true, then Schmidt's claim implies that $\varepsilon_{2s}^{(0)}$ is the first $\varepsilon_j^{(0)}$, $j = -1, 0, 1, \cdots$, which is good approximation to x. That is, the epsilon algorithm requires the computation of $2s$ iterates $x^{(1)}, \cdots, x^{(2s)}$ in order to achieve convergence.

In the next three sections, comparisons are also made of results obtained by the polynomial extrapolation formula (4.9) with those obtained by Wynn's vector-epsilon algorithm [17]. Let $v_{-1}^{(i)} = 0$ and $v_0^{(i)} = x^{(i)}$ $(i = 0, 1, 2, \cdots)$. The vector-epsilon algorithm computes for each $j = 0, 1, 2, \cdots$ the sequence of vectors $\{v_j^{(i)} : i = 0, 1, 2, \cdots\}$ defined by

$$v_{j+1}^{(i)} = v_{j-1}^{(i+1)} + \|v_j^{(i+1)} - v_j^{(i)}\|_2^{-2} \cdot [v_j^{(i+1)} - v_j^{(i)}]. \tag{5.5}$$

Numerical evidence suggests that, as in the case of the epsilon algorithm, if s is the smallest positive integer for which (5.2) holds true, then $v_{2s}^{(0)}$ is the first $v_j^{(0)}$, $j = -1, 0, 1, \cdots$, which is a good approximation to x.

6. Experimental objectives.

In this and the following two sections experimental results are tabulated and discussed. The polynomial extrapolation method and the epsilon algorithms are applied to the iterates $\{x_1^{(0)}, x_1^{(1)}, x_1^{(2)}, \cdots\}$ obtained from the block Gauss–Seidel iteration

$$A_{11}x_1^{(i+1)} = b_1 - A_{12}x_2^{(i)}, \tag{6.1a}$$

$$A_{22}x_2^{(i+1)} = b_2 - A_{21}x_1^{(i+1)}, \qquad i = 0, 1, 2, \cdots, \tag{6.1b}$$

where $x_1^{(0)} = x_2^{(0)} = 0$ and A_{ij} are matrices of even order and M is defined in terms of auxiliary matrices D, F, Q, $\bar{\Lambda}$, $\underline{\Lambda}$. Using the following definitions

$$A_{11} = A_{22} = F, \tag{6.2}$$

$$A_{12} = FP\bar{\Lambda}, \tag{6.3}$$

$$A_{21} = F\underline{\Lambda}P^{-1}, \tag{6.4}$$

$$P = QDQ^{-1}, \tag{6.5}$$

$$\begin{pmatrix} b_1 \\ b_2 \end{pmatrix} = \begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix} \begin{pmatrix} \underline{1} \\ \underline{1} \end{pmatrix} \tag{6.6}$$

with $\underline{1}^T = (1, 1, \cdots, 1)$, simple algebraic manipulations yield

$$x_1^{(i+1)} = Mx_1^{(i)} + g, \tag{6.7}$$

where

$$M = P\Lambda P^{-1}, \tag{6.8}$$

$$g = (P\bar{\Lambda} - M)\underline{1} \tag{6.9}$$

and

$$\Lambda = \bar{\Lambda}\underline{\Lambda}. \tag{6.10}$$

Furthermore,

$$\|(I-M)^{-1}\| \leqq \|P\| \cdot \|P^{-1}\| \cdot \|(I-\Lambda)^{-1}\| \tag{6.11}$$

and

$$\|M\| \leqq \|P\| \cdot \|P^{-1}\| \cdot \|\Lambda\|. \tag{6.12}$$

The reason for using the auxiliary matrices and cataloguing equations (6.7), (6.8), (6.9), (6.10), (6.11) and (6.12) is that the asymptotic convergence properties and the size of $p(1)$ may be controlled by appropriate choices of $\underline{\Lambda}$ and $\bar{\Lambda}$.

Throughout our experiments $\underline{\Lambda}$ and $\bar{\Lambda}$ are chosen so that

$$\Lambda = \begin{pmatrix} \Lambda_1 & & & & 0 \\ & \Lambda_2 & & & \\ & & \cdot & & \\ & & & \cdot & \\ & & & & \cdot \\ 0 & & & & \Lambda_{n/2} \end{pmatrix}, \tag{6.13}$$

where

$$\Lambda_j = c_j \begin{pmatrix} \cos\theta_j & -\sin\theta_j \\ \sin\theta_j & \cos\theta_j \end{pmatrix}, \qquad 0 \leqq \theta_j \leqq \pi. \tag{6.14}$$

Clearly the eigenvalues of M are then $c_j e^{\pm i\theta_j}$ $(j = 1, 2, \cdots, n/2)$, and hence both the size of the eigenvalues and the rank of M may be controlled by suitable choices of c_j.

By choosing Q and D in (6.5) to be an orthogonal and a diagonal matrix, respectively, the size of $\|P\|\,\|P^{-1}\|$ may be controlled, and hence a bound on $\|(I-M)^{-1}\|$ and $\|M\|$ is easily obtained. The reason, of course, for controlling $\|(I-M)^{-1}\|$ and $\|M\|$ is that these quantities appear in the error bounds (4.8) and (4.16).

In addition to bounding $\|(I-M)^{-1}\|$ we wish to control the size of the residuals appearing in (4.8) and (4.16). The rationale for controlling the residuals is provided by the following analysis.

Let

$$A_{11}x_1^{*(i+1)}+A_{12}x_1^{*(i)}-b_1=e_1^{(i+1)}, \tag{6.15a}$$

$$A_{21}x_1^{*(i)}+A_{22}x_2^{*(i)}-b_2=e_2^{(i)}. \tag{6.15b}$$

It follows that

$$x_1^{*(i+1)}=Mx_1^{*(i)}+g+A_{11}^{-1}(e_1^{(i+1)}-A_{12}A_{22}^{-1}e_2^{(i)}), \tag{6.16}$$

and therefore

$$\Delta^{*(i)}=M\Delta^{*(i-1)})+A_{11}^{-1}(e_1^{(i+1)}-e_1^{(i)}-A_{12}A_{22}^{-1}(e_2^{(i)}-e_2^{(i-1)})), \tag{6.17}$$

where it is assumed roundoff error in the calculation of $\Delta^{*(i)}$ is zero. Thus

$$\delta^{(i)}=F^{-1}(e_1^{(i+1)}-e_1^{(i)}-FP\bar{\Lambda}F^{-1}(e_2^{(i)}-e_2^{(i-1)})), \tag{6.18}$$

and hence

$$\|\delta^{(i)}\|\leqq 2\|F^{-1}\|(1+\|P\|\,\|\bar{\Lambda}\|)e, \tag{6.19}$$

where

$$\|e_j^{(i)}\|\leqq e. \tag{6.20}$$

By varying the parameter e for fixed F, P and $\bar{\Lambda}$, it is possible to study the effects of the size of the residuals on the error of the extrapolated results.

7. Numerical results: $n=10$.

In this section results are tabulated for the iteration (6.1) with $n=10$. The auxiliary matrices are defined by $\underline{\Lambda}=I$,

$$\Lambda=\bar{\Lambda}=\begin{pmatrix}\Lambda_1 & & & & 0\\ & \Lambda_2 & & & \\ & & \Lambda_3 & & \\ & & & \Lambda_4 & \\ 0 & & & & \Lambda_5\end{pmatrix} \tag{7.1}$$

with

$$\Lambda_i=c_i\begin{pmatrix}\cos\theta_i & -\sin\theta_i\\ \sin\theta_i & \cos\theta_i\end{pmatrix} \tag{7.2}$$

and $\theta_i=\pi(i-1)/4$,

$$F_{ij}=\begin{cases}4 & \text{if } i=j,\\ -1 & \text{if } |i-j|=1,\\ 0 & \text{otherwise,}\end{cases} \tag{7.3}$$

and

$$Q_{ij} = \sqrt{2/\pi}\sin(\pi ij/11). \tag{7.4}$$

The different choices of c_i are tabulated as parameters in the first five columns of Tables 7.1 and 7.2. Two different choices of the D matrix have been used. The choice of D in Table 7.1 is

$$D_{ij} = i\delta_{ij}, \tag{7.5}$$

where δ_{ij} is the Kronecker delta, whereas the choice of D in Table 7.2 is

$$D_{ij} = 2^{i-1}\delta_{ij}. \tag{7.6}$$

For both tables, the residuals $e_j^{(i)}$ are controlled so that $\|e_j^{(i)}\|_\infty \leqq e = 8.9\times 10^{-6}$. In the two tables, s, where $s+1$ is the degree of the minimal polynomial of M, is tabulated in column six. The remaining columns labeled $\|\underline{1} - x_1^{*(s+2)}\|$, $\|\underline{1} - x_1^{*(2s)}\|$, $\|\underline{1} - x^*\|$, $\|\underline{1} - \varepsilon^*\|$ and $\|\underline{1} - v^*\|$ give, respectively, in the $\|\cdot\|_\infty$ sense the errors in the

TABLE 7.1

c_1	c_2	c_3	c_4	c_5	s	$\|\underline{1}-x_1^{*(s+2)}\|$	$\|\underline{1}-x_1^{*(2s)}\|$	$\|\underline{1}-x^*\|$	$\|\underline{1}-\varepsilon^*\|$	$\|\underline{1}-v^*\|$
0.1	0	0	0	0	1	$3.5_{10}-3$	$3.6_{10}-2$	$7.7_{10}-7$	$8.9_{10}-7$	$8.9_{10}-7$
0.3	0.1	0	0	0	3	$8.6_{10}-3$	$2.8_{10}-3$	$5.3_{10}-6$	$3.2_{10}-6$	$2.5_{10}-6$
0.5	0.3	0.1	0	0	5	$2.7_{10}-2$	$3.3_{10}-3$	$7.6_{10}-6$	$3.4_{10}-5$	$1.1_{10}-5$
0.7	0.5	0.3	0.1	0	7	$1.4_{10}-1$	$2.4_{10}-2$	$8.3_{10}-5$	$7.7_{10}-4$	$9.7_{10}-5$
0.9	0.7	0.5	0.3	0.1	8	$1.3_{10}+0$	$6.5_{10}-1$	$2.6_{10}-4$	$7.9_{10}-2$	$4.9_{10}-4$
0.99	0.9	0.5	0.3	0.1	8	$5.2_{10}+0$	$4.9_{10}+0$	$1.1_{10}-2$	$6.5_{10}+1$	$2.7_{10}-1$
1.1	0.9	0.7	0.5	0.3	8	$9.9_{10}+0$	$1.7_{10}+1$	$6.0_{10}-4$	$1.6_{10}-1$	$1.2_{10}-3$
1.3	1.1	0.9	0.7	0.5	8	$5.4_{10}+1$	$2.3_{10}+2$	$3.4_{10}-4$	$1.1_{10}+0$	$8.2_{10}-4$
1.5	1.3	1.1	0.9	0.7	8	$2.3_{10}+2$	$2.7_{10}+3$	$3.2_{10}-4$	$6.9_{10}-2$	$1.3_{10}-3$
1.7	1.5	1.3	1.1	0.9	8	$8.7_{10}+2$	$1.1_{10}+4$	$1.8_{10}-4$	$5.8_{10}-2$	$1.6_{10}-3$
1.9	1.7	1.5	1.3	1.1	8	$2.8_{10}+3$	$9.6_{10}+4$	$2.5_{10}-4$	$1.8_{10}-1$	$1.6_{10}-3$
2.1	1.9	1.7	1.5	1.3	8	$8.2_{10}+3$	$5.5_{10}+5$	$3.1_{10}-4$	$6.2_{10}-2$	$1.3_{10}-2$

TABLE 7.2

c_1	c_2	c_3	c_4	c_5	s	$\|\underline{1}-x_1^{*(s+2)}\|$	$\|\underline{1}-x_1^{*(2s)}\|$	$\|\underline{1}-x^*\|$	$\|\underline{1}-\varepsilon^*\|$	$\|\underline{1}-v^*\|$
0.1	0	0	0	0	1	$4.5_{10}-2$	$4.7_{10}-1$	$2.6_{10}-6$	$1.7_{10}-6$	$5.7_{10}-6$
0.3	0.1	0	0	0	3	$1.1_{10}-1$	$3.3_{10}-2$	$1.3_{10}-4$	$9.3_{10}-5$	$4.3_{10}-5$
0.5	0.3	0.1	0	0	5	$3.1_{10}-1$	$3.6_{10}-2$	$7.2_{10}-3$	$2.5_{10}-1$	$1.7_{10}-2$
0.7	0.5	0.3	0.1	0	7	$2.1_{10}+0$	$3.2_{10}-1$	$8.2_{10}-3$	$1.5_{10}-1$	$4.6_{10}-2$
0.9	0.7	0.5	0.3	0.1	8	$2.1_{10}+1$	$9.0_{10}+0$	$6.8_{10}-2$	$6.4_{10}-1$	$3.7_{10}-1$
0.99	0.9	0.5	0.3	0.1	8	$8.7_{10}+1$	$8.3_{10}+1$	$7.1_{10}-1$	$8.3_{10}+1$	$6.4_{10}+0$
1.1	0.9	0.7	0.5	0.3	8	$1.9_{10}+2$	$2.4_{10}+2$	$5.9_{10}-2$	$2.7_{10}+1$	$3.1_{10}+0$
1.3	1.1	0.9	0.7	0.5	8	$1.2_{10}+3$	$3.8_{10}+3$	$2.8_{10}-2$	$1.0_{10}+0$	$3.8_{10}-1$
1.5	1.3	1.1	0.9	0.7	8	$5.6_{10}+3$	$4.4_{10}+4$	$3.2_{10}-2$	$1.8_{10}+0$	$1.2_{10}-1$
1.7	1.5	1.3	1.1	0.9	8	$2.2_{10}+4$	$2.4_{10}+5$	$6.5_{10}-2$	$5.1_{10}+0$	$4.7_{10}-2$
1.9	1.7	1.5	1.3	1.1	8	$7.2_{10}+4$	$1.2_{10}+6$	$1.9_{10}-1$	$2.8_{10}+0$	$2.4_{10}-1$
2.1	1.9	1.7	1.5	1.3	8	$2.1_{10}+5$	$8.9_{10}+6$	$6.5_{10}-2$	$2.6_{10}+1$	$5.5_{10}-1$

results obtained by using the block Gauss–Seidel iteration after $s+2$ and $2s$ iterations, the error from the polynomial extrapolation method applied after $s+2$ iterations, the error from the epsilon algorithm applied after $2s$ iterations, and finally the error from the vector-epsilon algorithm applied after $2s$ iterations. It should be noted that, with only two exceptions,

$$\|\underline{1}-x^*\| \leqq \|\underline{1}-v^*\| \leqq \|\underline{1}-\varepsilon^*\|.$$

Furthermore, the amount of work in obtaining x^* is approximately half of that in obtaining either v^* or ε^*.

The results of these tables are in accordance with the error bounds obtained in § 4. Note that:

1. the results of Table 7.1 are better than those of Table 7.2 solely because of the choice of P, which makes $\|(I-M)^{-1}\|$ and $\|\delta^{(i)}\|$ larger in Table 7.2, and
2. the results deteriorate as the eigenvalues of M approach 1 (note the case $c_1 = 0.99$), the reason being that $\|(I-M)^{-1}\|$ becomes large and/or $\beta(1)$ is near zero.

It is worth noting that for $c_1 = .99$ the epsilon and vector-epsilon algorithms are not as effective as the polynomial extrapolation. In all other tests that have been conducted, it has been found the epsilon algorithm is very ineffective if the eigenvalues of M are close to one, whereas the vector-epsilon algorithm often gives slightly better results than does the polynomial extrapolation method. Before drawing conclusions, however, one should remember that the vector-epsilon algorithm requires approximately twice as many iterates as does the polynomial extrapolation method.

At this point it should be remarked that the rank of M has not been used to decide when to apply the extrapolation formula. Indeed, if δ_k represents the residual defined by

$$(7.7) \qquad \sum_{i=0}^{k-1} \beta_i \Delta^{*(i)} = \delta_k/\|\beta\|, \qquad \|\beta\| = \max_{0 \leqq i \leqq k-1} \{|\beta_i|\},$$

where β is a tentative annihilating polynomial obtained at the kth iteration, then $\|\delta_k\|/\|\beta\|$ shows a marked decrease when finally $k = s+2$. The bound (4.8) on the error of the polynomial extrapolation formula then suggests that this error should also suddenly decrease when $k = s+2$. Table 7.3, which was used to obtain the second last row in Table 7.2, illustrates this convergence phenomenon. Observe the sudden drop in $\|\delta_k\|/\|\beta\|$ and in $\|\underline{1}-\mathrm{x}^*\|$ when $k = s+2 = 10$. This convergence phenomenon also occurs for the epsilon algorithms. The errors $\|\underline{1}-\varepsilon^*\|$ and $\|\underline{1}-v^*\|$ suddenly decrease when $k = 2s = 16$.

It should be noted that in the example of Table 7.3, the nonzero eigenvalues of M are of approximately the same magnitude. For such examples the drop in the errors when $k = s+2$ for the polynomial extrapolation method and when $k = 2s$ for the epsilon algorithms is particularly sudden. When the nonzero eigenvalues of M differ widely in magnitude, these drops are less spectacular. Nevertheless, for all M, the approximate solutions obtained by the polynomial extrapolation method stabilize for $k \geqq s+2$ and those obtained by the epsilon algorithms stabilize for $k \geqq 2s$.

TABLE 7.3

k	$\lVert 1-x^{*(k)}\rVert$	$\lVert\delta_k\rVert/\lVert\beta\rVert$	$\lVert \underline{1}-x^*\rVert$	$\lVert \underline{1}-\varepsilon^*\rVert$	$\lVert \underline{1}-\varepsilon^*\rVert$
0	$1.0_{10}+0$				
1	$7.7_{10}+2$				
2	$7.6_{10}+2$	$7.3_{10}+2$	$7.2_{10}+2$	$3.1_{10}+3$	$3.4_{10}+2$
3	$7.9_{10}+2$	$2.0_{10}+2$	$7.1_{10}+2$		
4	$6.5_{10}+2$	$8.5_{10}+1$	$7.7_{10}+2$	$3.8_{10}+3$	$6.1_{10}+2$
5	$5.1_{10}+2$	$5.8_{10}+1$	$7.2_{10}+2$		
6	$3.1_{10}+3$	$2.6_{10}+1$	$7.8_{10}+3$	$1.5_{10}+4$	$8.7_{10}+2$
7	$6.3_{10}+3$	$2.1_{10}+1$	$1.8_{10}+3$		
8	$6.1_{10}+3$	$4.6_{10}+0$	$7.1_{10}+3$	$8.0_{10}+3$	$7.5_{10}+2$
9	$3.8_{10}+4$	$3.6_{10}+0$	$4.9_{10}+3$		
10	$7.2_{10}+4$	$1.4_{10}-3$	$1.9_{10}-1$	$2.0_{10}+3$	$5.9_{10}+2$
11	$9.9_{10}+4$	$2.9_{10}-3$	$1.3_{10}-1$		
12	$1.1_{10}+5$	$6.0_{10}-4$	$1.0_{10}-1$	$5.7_{10}+2$	$7.3_{10}+2$
13	$7.9_{10}+4$				
14	$1.1_{10}+5$			$3.9_{10}+2$	$6.3_{10}+2$
15	$1.7_{10}+5$				
16	$1.2_{10}+6$			$2.8_{10}+0$	$2.4_{10}-1$
17	$3.7_{10}+6$				
18	$7.7_{10}+6$			$1.2_{10}+0$	$7.8_{10}-1$
19	$1.3_{10}+7$				
20	$1.7_{10}+7$			$2.3_{10}+0$	$4.1_{10}-1$

Note that in Table 7.3, results for the epsilon algorithms are not reported for k odd because such values would be meaningless (see Wynn [15]). Furthermore, for $k \geqq 13$, results are not given for the polynomial extrapolation method, since for such k the system

$$\sum_{i=0}^{k-2} \beta_i \Delta^{*(i)} = -\Delta^{*(k-1)} \tag{7.8}$$

which determines the coefficients β_i has $k-12$ degrees of freedom.

8. Numerical results: $n = 400$. This section discusses more realistic examples in which $s \ll n$. Let $n = 400$ in (6.1), and define

$$\bar{\Lambda} = \underline{\Lambda} = \begin{pmatrix} \underline{\Lambda}_1 & & & & \\ & \underline{\Lambda}_2 & & & \\ & & \cdot & & \\ & & & \cdot & \\ & & & & \cdot \\ & & & & & \underline{\Lambda}_{n/2} \end{pmatrix}, \tag{8.1}$$

where

$$\underline{\Lambda}_j = \underline{c}_j \begin{pmatrix} \cos \underline{\theta}_j & -\sin \underline{\theta}_j \\ \sin \underline{\theta}_j & \cos \underline{\theta}_j \end{pmatrix}, \tag{8.2}$$

$$\underline{\theta}_j = (j-1)\pi/14, \qquad 1 \leqq j \leqq 8, \tag{8.3}$$

$$\underset{\sim}{c}_j = \begin{cases} [(8-j)d_1 + (j-1)d_2]/7, & 1 \leqq j \leqq 8, \\ 0, & 8 < j, \end{cases} \tag{8.4}$$

and d_1, d_2 are parameters to be varied. Let

$$F = \begin{pmatrix} H & -I & & & \\ -I & H & -I & & \\ & & \cdot & & \\ & & \cdot & & \\ & & \cdot & & \\ & & -I & H & -I \\ & & & -I & H \end{pmatrix}, \tag{8.5}$$

where H and I are matrices of order 20 with

$$H_{ij} = \begin{cases} 4, & i = j, \\ -1, & |i-j| = 1, \\ 0 & \text{otherwise}, \end{cases} \tag{8.6}$$

and define Q in (6.5) by

$$Q_{ij} = Q_{ji} = n + 1 - i. \tag{8.7}$$

Note that Q^{-1} is given by

$$Q_{ij}^{-1} = \begin{cases} 1, & i = j = 1, \\ 2, & i = j \neq 1, \\ -1, & |i-j| = 1, \\ 0 & \text{otherwise}. \end{cases} \tag{8.8}$$

Results for different choices of the parameters d_1 and d_2 are summarized in Tables 8.1, 8.2 and 8.3. The D matrix used in the first two tables is defined by

$$D_{ij} = [1 + (i-1)/399]/\delta_{ij}, \tag{8.9}$$

whereas

$$D_{ij} = [1 + 19(i-1)/399]/\delta_{ij} \tag{8.10}$$

in Table 8.3. The conditions for obtaining the results in Tables 8.1 and 8.2 are identical with the exception that the bound e on the residuals $e_j^{(i)}$ in (6.15) is 9.6×10^{-6} in Table 8.1 and $e = 8.2 \times 10^{-4}$ in Table 8.2. In Table 8.3, $e = 9.6 \times 10^{-6}$.

Since $\Lambda = \bar{\Lambda}\underset{\sim}{\Lambda}$, the nonzero eigenvalues of M are $\underset{\sim}{c}_j^2 e^{\pm i2\theta_j}$, $1 \leqq j \leqq 8$, where it is assumed that $d_1 \neq 0$ and $d_2 \neq 0$ in (8.4). With θ_j defined by (8.3), it then follows that the degree of minimal polynomial of M is $s + 1 = 15$. The analysis of the previous sections then suggests that the polynomial extrapolation method should give good results after $s + 2 = 16$ iterations, whereas the epsilon algorithms should do so after $2s = 28$ iterations. This happens in practice. Indeed, the convergence phenomenon referred to in Table 7.3 was witnessed for each row in Tables 8.1, 8.2 and 8.3. The column labeled $\|\underset{\sim}{1} - x^*\|$ in the tables therefore refers to error in the results obtained by the polynomial extrapolation method after 16 iterations, whereas those labeled $\|\underline{1}-\varepsilon^*\|$ and $\|\underline{1}-v^*\|$ refer to errors in the epsilon algorithm and the vector-epsilon algorithm, respectively, after 28 iterations.

Examination of Tables 8.1, 8.2 and 8.3 leads to the following observations, all of which are consistent with the analysis of the previous sections.

TABLE 8.1

d_1	d_2	$\|\underline{1}-x_1^{*(16)}\|$	$\|\underline{1}-x_1^{*(28)}\|$	$\|\underline{1}-x^*\|$	$\|\underline{1}-\varepsilon^*\|$	$\|\underline{1}-v^*\|$
0.7	0.9	$3.7_{10}-2$	$2.2_{10}-3$	$5.5_{10}-5$	$1.5_{10}-2$	$5.2_{10}-5$
0.9	0.9	$7.6_{10}-2$	$5.2_{10}-3$	$1.5_{10}-4$	$4.6_{10}-2$	$1.7_{10}-4$
0.95	0.9	$4.3_{10}-1$	$1.4_{10}-1$	$7.8_{10}-4$	$6.9_{10}-1$	$4.8_{10}-4$
0.99	0.9	$1.6_{10}+0$	$1.7_{10}+0$	$8.1_{10}-2$	$2.1_{10}+1$	$1.8_{10}-1$
0.999	0.9	$2.2_{10}+0$	$3.1_{10}+0$	$3.4_{10}-2$	$2.5_{10}+0$	$2.7_{10}-2$
1.05	0.9	$9.9_{10}+0$	$7.1_{10}+1$	$4.4_{10}-3$	$4.0_{10}-1$	$5.6_{10}-3$
1.1	0.9	$4.2_{10}+1$	$1.3_{10}+3$	$7.2_{10}-3$	$1.5_{10}+1$	$5.8_{10}-3$
1.3	0.9	$9.1_{10}+3$	$5.8_{10}+7$	$1.8_{10}-2$	$2.7_{10}+0$	$1.3_{10}-1$
0.9	0.7	$5.1_{10}-2$	$2.9_{10}-3$	$1.4_{10}-4$	$6.0_{10}-1$	$1.4_{10}-4$
0.9	0.95	$2.9_{10}-1$	$7.7_{10}-2$	$2.0_{10}-4$	$1.3_{10}-1$	$1.8_{10}-4$
0.9	1.05	$8.3_{10}+0$	$4.9_{10}+1$	$2.6_{10}-4$	$1.6_{10}-1$	$3.1_{10}-4$
0.9	1.1	$4.2_{10}+1$	$1.0_{10}+3$	$2.5_{10}-3$	$3.0_{10}+0$	$1.9_{10}-3$
0.9	1.3	$1.5_{10}+4$	$5.2_{10}+7$	$7.6_{10}-2$	$1.6_{10}+0$	$3.4_{10}-1$
0.9	2.0	$5.1_{10}+10$	$7.7_{10}+19$	$5.4_{10}+0$	$1.4_{10}+0$	$1.1_{10}+0$

TABLE 8.2

d_1	d_2	$\|\underline{1}-x_1^{*(16)}\|$	$\|\underline{1}-x_1^{*(28)}\|$	$\|\underline{1}-x^*\|$	$\|\underline{1}-\varepsilon^*\|$	$\|\underline{1}-v^*\|$
0.7	0.9	$1.0_{10}-1$	$1.3_{10}-2$	$8.9_{10}-5$	$7.4_{10}-2$	$6.9_{10}-5$
0.9	0.9	$1.8_{10}-1$	$2.6_{10}-2$	$5.6_{10}-3$	$3.7_{10}-1$	$2.5_{10}-3$
0.95	0.9	$8.9_{10}-1$	$5.8_{10}-1$	$1.4_{10}-1$	$1.1_{10}+0$	$1.1_{10}-1$
1.05	0.9	$2.5_{10}+1$	$3.2_{10}+2$	$5.1_{10}-3$	$4.0_{10}+1$	$5.2_{10}-2$
1.1	0.9	$1.2_{10}+2$	$6.1_{10}+3$	$1.1_{10}-2$	$1.2_{10}+2$	$3.6_{10}-2$
1.3	0.9	$3.6_{10}+4$	$2.8_{10}+9$	$2.3_{10}-2$	$2.0_{10}+3$	$3.9_{10}-1$

TABLE 8.3

d_1	d_2	$\|\underline{1}-x_1^{*(16)}\|$	$\|\underline{1}-x_1^{*(28)}\|$	$\|\underline{1}-x^*\|$	$\|\underline{1}-\varepsilon^*\|$	$\|\underline{1}-v^*\|$
0.7	0.9	$6.8_{10}-2$	$7.7_{10}-3$	$2.9_{10}-3$	$1.7_{10}-2$	$3.5_{10}-3$
0.9	0.9	$1.7_{10}+0$	$2.3_{10}-1$	$1.9_{10}-2$	$2.0_{10}+0$	$1.1_{10}-2$
0.95	0.9	$7.7_{10}+0$	$3.7_{10}+0$	$6.3_{10}-2$	$8.4_{10}+1$	$3.7_{10}-2$
1.05	0.9	$3.6_{10}+2$	$4.8_{10}+3$	$1.6_{10}-2$	$1.3_{10}+2$	$1.4_{10}-1$
1.1	0.9	$1.5_{10}+3$	$6.8_{10}+4$	$1.2_{10}-1$	$2.9_{10}+2$	$4.3_{10}-1$
1.3	0.9	$5.8_{10}+5$	$4.2_{10}+9$	$1.4_{10}-1$	$1.7_{10}+2$	$2.5_{10}+0$

Observation 1. The eigenvalue of M closest to unity for all examples in the tables is d_1^2. Observe that as d_1 approaches 1, the results obtained by all three methods deteriorate.

Observation 2. Since $e = 8.2 \times 10^{-4}$ in Table 8.2 and $e = 9.6 \times 10^{-6}$ in Table 8.1, the bound (6.19) on the $\delta^{(i)}$'s is larger in Table 8.2 than it is in Table 8.1. Note that for all three methods the errors are larger in Table 8.2 than they are in the corresponding rows of Table 8.1.

Observation 3. Tables 8.1 and 8.3 are also generated under identical conditions with the exception that now $\|D\| \cdot \|D^{-1}\|$ is larger in Table 8.3 than it is in Table 8.1. Thus the bound for $\|(I-M)^{-1}\|$, with the same Λ, is larger in Table 8.3. Observe the corresponding larger errors in Table 8.3.

Observation 4. The epsilon algorithm performs poorly in comparison with both the polynomial extrapolation method and the vector-epsilon algorithm. The results of the latter two methods are almost identical; those of the vector-epsilon algorithm being superior.

Observation 5. The polynomial extrapolation method and the vector-epsilon algorithm significantly improve the accuracy of the last Gauss–Seidel iterate. This improvement however is deceptive for rapidly divergent sequences, since in this case, the last iterate is further away from the solution than is the initial approximation $x_1^{(0)}$. For example, in the last row of Table 8.1 for which $\rho(M) = d_2^2 = 4$, $\|\underline{1} - x_1^{(0)}\|_\infty = 1$ while $\|\underline{1} - x^*\| = 5.4$ and $\|\underline{1} - v^*\| = 1.1$. Thus in this case, and in general when $\rho(M)$ and s are large, the polynomial extrapolation method and the epsilon algorithms may produce results which are worse than the initial approximation $x_1^{(0)}$.

9. Woodbury's method.

For matrices M of rank s and for those with s nonzero eigenvalues, there exist matrices U and V of dimension $n \times s$ such that

$$M = UV^T. \tag{9.1}$$

U and V can be computed by a variety of methods available for rank factorization or for singular value decomposition of a matrix. (See Rao and Mitra [9] for a brief discussion of some of these methods and for references to others.) For a general matrix M, the cheapest methods appear to require approximately $2sn^2$ multiplications. If basis vectors for the column or row space of M are known beforehand, this cost can be reduced to approximately sn^2 multiplications. (Similar arguments to the ones above and in what follows can be made in the cases when M has numerical rank s and when M has s highly dominant eigenvalues.)

Given the decomposition (9.1), Woodbury's formula (see, for example, Householder [6]) says

$$x = (I-M)^{-1}g = (I - UT^{-1}V^T)g, \tag{9.2}$$

where

$$T = V^TU - I. \tag{9.3}$$

Since T is a matrix of order s only, (9.2) gives x cheaply (in only $O(s^2n)$ multiplications) once the decomposition (9.1) becomes available.

If instead the polynomial extrapolation method is used to find x, one must first compute $x^{(1)}, x^{(2)}, \cdots, x^{(s+2)}$ which satisfy the iteration

$$x^{(i+1)} = Mx^{(i)} + g. \tag{9.4}$$

(Note: If the annihilating polynomial of M with respect to $\Delta^{(0)}$ is of degree less than $s+1$, fewer than $s+2$ iterates are required.) For general M, the cost of computing the iterates $x^{(i)}$ is approximately $(s+2)n^2$. The extrapolation of the iterates once they become available requires an additional $O(s^2n)$ multiplications only.

Thus for general M, the cost of Woodbury's formula is dominated by the cost of the decomposition (9.1), whereas the cost of the polynomial extrapolation method is dominated by the cost of computing the iterates satisfying (9.4). In applications where the iterates $x^{(0)}, x^{(1)}, \cdots, x^{(s+2)}$ only (not M and g) are known a priori, and it is for these problems that the polynomial extrapolation method is primarily intended, the polynomial extrapolation method costs less than Woodbury's method. On the other hand, Woodbury's method should be used when U and V in (9.1) are known beforehand and the iterates are not. For general M given, but with the iterates $x^{(i)}$ and the decomposition (9.1) unknown, the cost of both methods is comparable. The Woodbury's formula can be slightly cheaper if basis vectors for the column or row space of M are known, but is almost twice as expensive if they are not.

As a means of comparing the accuracy of the two methods, the experiments of § 7 were repeated using Woodbury's formula, the results of which are summarized in Tables 9.1 and 9.2. Tables 9.1 and 9.2 are obtained under conditions identical to those of Tables 7.1 and 7.2, respectively. In these experiments,

$$M = P\Lambda P^{-1} \tag{9.5}$$

(see equations (6.5), (6.8), (7.1), (7.4), (7.5) and (7.6)). Let

$$r = \operatorname{rank}(M) = \operatorname{rank}(\Lambda). \tag{9.6}$$

The decomposition (9.1) is obtained by setting U to be the first r columns of $P\Lambda$ and V^T to be the first r rows of P^{-1}. Care was taken to insure that

$$\|U-\tilde{U}\|_\infty \leqq \|\tilde{U}\|_\infty \times 8.9\times 10^{-6}, \tag{9.7a}$$

$$\|V^T-\tilde{V}^T\|_\infty \leqq \|\tilde{V}^T\|_\infty \times 8.9\times 10^{-6}, \tag{9.7b}$$

where $\tilde{U}$ is the computed matrix U and $\tilde{V}$ is computed matrix V. Double precision computation is used in applying Woodbury's formula (9.2). Gaussian elimination with partial pivoting is used to obtain $T^{-1}(V^Tg)$. (Note: For the polynomial extrapolation method, single precision computation is used throughout.)

The tables reveal that the accuracy of both methods is approximately the same except when an eigenvalue of M is close to 1 (i.e., when $I-M$ is ill-conditioned) in which case the polynomial extrapolation method gives a more accurate result. The poor performance of Woodbury's formula in this case can be attributed to the fact that T is ill-conditioned and the small errors (9.7) which were made in computing U and V greatly affect the inverse of T. In applications where the decomposition (9.1) cannot be as accurately obtained as in these experiments (for example, when the decomposition is ill-conditioned), the performance of Woodbury's formula would deteriorate further.

The polynomial extrapolation method performs relatively well when $I-M$ is ill-conditioned for the following reason which is repeated for emphasis. It does not

TABLE 9.1

c_1	c_2	c_3	c_4	c_5	s	r	$\|\underline{1}-x^*\|$	$\|\underline{1}-h^*\|$
− 0.1	0	0	0	0	1	2	$7.7_{10}-7$	$4.4_{10}-6$
0.3	0.1	0	0	0	3	4	$5.3_{10}-6$	$1.8_{10}-5$
0.5	0.3	0.1	0	0	5	6	$7.6_{10}-6$	$4.7_{10}-5$
0.7	0.5	0.3	0.1	0	7	8	$8.3_{10}-5$	$1.2_{10}-4$
0.9	0.7	0.5	0.3	0.1	8	10	$2.6_{10}-4$	$5.6_{10}-4$
0.99	0.9	0.5	0.3	0.1	8	10	$1.1_{10}-2$	$8.2_{10}+0$
1.1	0.9	0.7	0.5	0.3	8	10	$6.0_{10}-4$	$6.8_{10}-4$
1.3	1.1	0.9	0.7	0.5	8	10	$3.4_{10}-4$	$1.8_{10}-4$
1.5	1.3	1.1	0.9	0.7	8	10	$3.2_{10}-4$	$1.7_{10}-4$
1.7	1.5	1.3	1.1	0.9	8	10	$1.8_{10}-4$	$1.3_{10}-4$
1.9	1.7	1.5	1.3	1.1	8	10	$2.5_{10}-4$	$1.1_{10}-4$
2.1	1.9	1.7	1.5	1.3	8	10	$3.1_{10}-4$	$1.9_{10}-5$

TABLE 9.2

c_1	c_2	c_3	c_4	c_5	s	r	$\|\underline{1}-x^*\|$	$\|\underline{1}-h^*\|$
0.1	0	0	0	0	1	2	$2.6_{10}-6$	$8.0_{10}-5$
0.3	0.1	0	0	0	3	4	$1.3_{10}-4$	$5.0_{10}-4$
0.5	0.3	0.1	0	0	5	6	$7.2_{10}-3$	$5.1_{10}-3$
0.7	0.5	0.3	0.1	0	7	8	$8.2_{10}-3$	$1.2_{10}-2$
0.9	0.7	0.5	0.3	0.1	8	10	$6.8_{10}-2$	$4.0_{10}-2$
0.99	0.9	0.5	0.3	0.1	8	10	$7.1_{10}-1$	$2.7_{10}+2$
1.1	0.9	0.7	0.5	0.3	8	10	$5.9_{10}-2$	$3.9_{10}-2$
1.3	1.1	0.9	0.7	0.5	8	10	$2.8_{10}-2$	$1.6_{10}-2$
1.5	1.3	1.1	0.9	0.7	8	10	$3.2_{10}-2$	$1.1_{10}-2$
1.7	1.5	1.3	1.1	0.9	8	10	$6.5_{10}-2$	$2.2_{10}-2$
1.9	1.7	1.5	1.3	1.1	8	10	$1.9_{10}-1$	$3.9_{10}-2$
2.1	1.9	1.7	1.5	1.3	8	10	$6.5_{10}-2$	$7.8_{10}-2$

matter if the problem of computing $\alpha_0, \alpha_1, \cdots, \alpha_{s^*+1}$ is ill-conditioned. The only requirement is that the residual δ^* defined by

$$\delta^* = \sum_{i=0}^{s^*+1} \beta_i \Delta^{*(i)},$$

where β_i is the approximation to α_i, be small and that $\beta(1)$ be bounded away from zero.

One important question regarding the comparison of the two methods which has not yet been mentioned is the following: Since the polynomial extrapolation method is extrapolating iterates, would the results shown in the tables for the method be worse if the initial approximation $x^{(0)}$ to x were worse? Since the error bound (4.8) depends implicitly on $\|x-x^{(0)}\|$, the answer to the question is yes. Yet we believe that the results are not prejudiced too greatly in favor of the polynomial extrapolation method since, in practice, one can usually predict $x^{(0)}$ so that $\|x-x^{(0)}\|$ is of the same order of magnitude as $\|x\|$ (in the tables $\|x-x^{(0)}\|=\|x\|$). Indeed, in many applications, a much better approximation to x is available.

Woodbury's formula, however, has two advantages over the polynomial extrapolation method. (i) It can be applied to multiple right-hand sides with only a little extra cost. (ii) Iterative refinement can be used to improve the computed solution also at a small extra cost. Of course, the solution obtained by the polynomial extrapolation method can also be improved by restarting the process using the last extrapolated value as the new initial approximation $x^{(0)}$ to x, but each such extrapolation is just as expensive as the previous one.

As a final remark it should be noted that the coding for the Woodbury's formula is much more complex than that for the polynomial extrapolation method.

10. Conclusions. The polynomial extrapolation method is an effective algorithm for finding limits and antilimits of vector sequences defined by (1.1). For problems of this type the polynomial extrapolation method should be used in preference to the epsilon or vector-epsilon algorithms because

1. it requires approximately half the number of iterates and
2. a rigorous justification and error bound are known.

For problems in which M is known, the polynomial extrapolation method competes favorably with Woodbury's method. If reasonable estimates of the solution vector are available and, in addition, if a basis for M is not known, we tend to favor the polynomial extrapolation method.

Acknowledgment. We are indebted to the referees for their comments and suggestions for revision on an earlier version of this paper. We especially would like to thank one of the referees for his shrewd observations on Tables 7.1 and 7.2, which led to the detection of errors in both tables.

REFERENCES

[1] P. BUSINGER AND G. GOLUB, *Linear least squares solutions by Householder transformations*, Numer. Math., 7 (1965), pp. 269–276.

[2] G. GOLUB, *Numerical methods for solving linear least-squares problems*, Ibid., 7 (1965), pp. 206–216.

[3] ———, *Least squares, singular values, and matrix approximations*, Apl. Mat., 13 (1968), pp. 44–51.

[4] G. GOLUB AND REINSCH, *Singular value decomposition and least squares solutions*, Numer. Math., 14 (1970), pp. 403–420.

[5] G. GOLUB AND W. KAHAN, *Calculating singular values of a matrix*, this Journal, 2 (1965), pp. 205–224.

[6] A. S. HOUSEHOLDER, *The Theory of Matrices in Numerical Analysis*, Blaisdell, New York, 1964.

[7] R. L. JOHNSTON, *Numerical computation of electric dipole fields in conducting media, I(U)*, DREP rep. 1967-1.

[8] B. NOBLE, *Applied Linear Algebra*, Prentice-Hall, Englewood Cliffs, N.J., 1969.

[9] C. R. RAO AND S. K. MITRA, *Generalized Inverse of Matrices and Its Applications*, John Wiley, New York, 1971.

[10] J. R. SCHMIDT, *On the numerical solution of linear simultaneous equations by an iterative method*, Philos. Mag., Ser. 7, 32 (1941), pp. 369–383.

[11] D. SHANKS, *Non-linear transformation of divergent and slowly convergent sequences*, J. Math. and Phys., 34 (1955), pp. 1–42.

[12] G. W. STEWART, *Introduction to Matrix Computations*, Academic Press, New York, 1973.

[13] J. H. WILKINSON, *The Algebraic Eigenvalue Problem*, Oxford University Press, Belfast, 1965.

[14] ———, *The solution of ill-conditioned linear equations*, Mathematical Methods for Digital Computers, vol. II, A. Ralston and H. S. Wilf, eds., John Wiley, New York, 1967.
[15] P. WYNN, *On a device for computing the $e_m(S_n)$ transformation*, Math. Tables Aids Comput., 10 (1956), pp. 91–96.
[16] ———, *On the propagation of error in certain non-linear algorithms*, Numer. Math., 1 (1959), pp. 142–149.
[17] ———, *Acceleration techniques for iterated vector and matrix problems*, Math. Comp., 16 (1962), pp. 301–322.
[18] ———, *Vector continued fractions*, Linear Algebra and Appl., 1 (1968), pp. 357–395.