NASA Technical Memorandum 101327
ICOMP-88-17

Convergence Acceleration for Vector Sequences and Applications to Computational Fluid Dynamics

Avram Sidi
Institute for Computational Mechanics in Propulsion
Lewis Research Center
Cleveland, Ohio

and

Mark L. Celestina
Sverdrup Technology, Inc.
NASA Lewis Research Center Group
Cleveland, Ohio

August 1988

CONVERGENCE ACCELERATION FOR VECTOR SEQUENCES AND APPLICATIONS TO COMPUTATIONAL FLUID DYNAMICS

Avram Sidi*
Institute for Computational Mechanics in Propulsion
Lewis Research Center
Cleveland, Ohio 44135

and

Mark L. Celestina
Sverdrup Technology, Inc.
NASA Lewis Research Center Group
Cleveland, Ohio 44135

SUMMARY

Some recent developments in acceleration of convergence methods for vector sequences are reviewed. The methods considered are the minimal polynomial extrapolation, the reduced rank extrapolation, and the modified minimal polynomial extrapolation. The vector sequences to be accelerated are those that are obtained from the iterative solution of linear or nonlinear systems of equations. The convergence and stability properties of these methods as well as different ways of numerical implementation are discussed in detail. Based on the convergence and stability results, strategies that are useful in practical applications are suggested. Two applications to computational fluid mechanics involving the three-dimensional Euler equations for ducted and external flows are considered. The numerical results demonstrate the usefulness of the methods in accelerating the convergence of the time-marching techniques in the solution of steady state problems.

1. INTRODUCTION

Let R^d denote the d-dimensional Cartesian space and let Ω be a subset of R^d. Denote the boundary of Ω by $\partial\Omega$. Let $u(t)$ be the solution to the initial boundary value problem (IBVP)

$$\frac{du}{dt} + \phi(u) = 0 \quad \text{in} \quad \Omega,$$

$$\Lambda(u) = 0 \quad \text{on} \quad \partial\Omega \quad \text{(boundary condition)}, \tag{1.1}$$

$$M(u) = 0 \quad \text{at} \quad t = 0 \quad \text{(initial condition)}.$$

Here ϕ, Λ, and M are linear or nonlinear operators (differential, integral, etc.), and t denotes time.

*Work funded under Space Act Agreement C99066G; present address: Computer Science Dept., Technion - Israel Institute of Technology, Haifa, Israel.

Assume that the IBVP in equation (1.1) has a time-independent (steady state) solution that we shall denote u^*. Then u^* is the solution to the boundary value problem (BVP)

$$\phi(u^*) = 0 \quad \text{in} \quad \Omega, \qquad\qquad (1.2)$$
$$\Lambda(u^*) = 0 \quad \text{on} \quad \partial\Omega \quad \text{(boundary condition)}.$$

Since $u^* = \lim_{t\to\infty} u(t)$, one of the widely used techniques for determining u^* has been one in which the IBVP in equation (1.1) is solved by a time-marching technique. Specifically, equation (1.1) is discretized to give the iterative scheme

$$u^{n+1} = \psi(u^n), \qquad n = 0,1, \ldots, \qquad (1.3)$$

where u^n is the (discrete) approximation to $u(n\Delta t)$ and $\Delta t > 0$ is the time increment. The initial approximation u^0 needs to be provided in an appropriate way. Obviously, when the discretization scheme in equation (1.3) is stable $\lim_{n\to\infty} u^n = \tilde{u}$, where $\tilde{u}$ is the fixed point of the function $\psi(z)$ and in addition is a discrete approximation to u^*. In practice, when $||u^{n+1} - u^n||/||u^1 - u^0|| < \varepsilon$, for some prescribed $\varepsilon > 0$, u^n is taken to be an acceptable approximation to $\tilde{u}$. Note that $u^{n+1} - u^n = \psi(u^n) - u^n$, which is the residual of $\psi(z) - z$ for $z = u^n$, and $||\bullet||$ denotes some appropriate vector norm.

When the discretization scheme in equation (1.3) is stable, for n sufficiently large, u^n is close to $\tilde{u}$, hence

$$u^{n+1} = \psi(u^n) = \psi(\tilde{u}) + \psi'(\tilde{u})(u^n - \tilde{u}) + \varepsilon_n = Au^n + b + \varepsilon_n, \qquad (1.4)$$

where $A = \psi'(\tilde{u})$ is the Jacobian of ψ at $\tilde{u}$, $b = \psi(\tilde{u}) - \psi'(\tilde{u})\tilde{u}$, and ε_n satisfies $||\varepsilon_n|| \leq C||u^n - \tilde{u}||^2$ for some fixed constant C. Obviously A and b are independent of n. That is to say, the u^n, for sufficiently large n, satisfy (approximately)

$$u^{n+1} \simeq Au^n + b. \qquad (1.5)$$

Thus

$$||u^n - \tilde{u}|| = O\{\rho^n[\psi'(\tilde{u})]\} \quad \text{as} \quad n \to \infty, \qquad (1.6)$$

where, for any matrix B, $\rho(B)$ denotes its spectral radius.

Normally $\rho[\psi(\tilde{u})]$ is very close to 1, although it is strictly less than 1. In addition, in some cases it can be shown that $\rho[\psi'(\tilde{u})]$ tends to 1 as the meshsize of the discretization tends to zero. This causes the sequence u^0, u^1, u^2, . . . , to converge to $\tilde{u}$ very slowly. One simple way to overcome this problem is by use of convergence acceleration (or equivalently extrapolation) methods that do not require that changes be made in the basic iterative scheme of equation (1.3). In the next section we shall describe three such methods, namely, the minimal polynomial extrapolation (MPE) of Cabay and Jackson (ref. 3), the reduced rank extrapolation (RRE) of Eddy (ref. 5) and Mesina (ref. 12), and the modified minimal polynomial extrapolation (MMPE) of Sidi, Ford, and Smith (ref. 19). A slightly different version of RRE was earlier proposed by Kaniel and Stein (ref. 11). All three methods operate on the given vector sequence $u^0, u^1, u^2, \ldots$, only, irrespective of how this sequence is obtained (or, in the context of the discretization scheme (eq. (1.3)), irrespective of what ψ is). This is an important property of these methods as it allows them to be applied in conjunction with iterative methods for both linear and nonlinear problems with the same ease. We must add, however, that the rates of acceleration that can be achieved by using these methods depend on the sequence in consideration. Hence, in the context of equation (1.3), it is the structure of ψ that determines the rates of acceleration.

In the next section we shall give a brief description of MPE, RRE, and MMPE. These descriptions are based on the developments of the survey paper of Smith, Ford, and Sidi (ref. 20) and of ref. 19 and of Sidi (ref. 15). Further generalizations that are proposed in reference 19 will also be mentioned. The convergence and stability properties of these methods have been analyzed in references 15 and 19, Sidi and Bridger (ref. 18), and Sidi (ref. 16). Some of these results will be reviewed in section 3. In section 4 we shall present some applications to certain three-dimensional fluid mechanics problems.

Extrapolation methods have recently been used by several authors in computational fluid mechanics applications, see, for example, Hafez et al. (ref. 7), Wong and Hafez (ref. 22), Wigton, Yu, and Young (ref. 21), Jespersen and Buning (ref. 10), and Reddy and Jacocks (ref. 14). One of the methods described in the present work, namely, MPE, with some variation, is employed in references 10 and 14.

Before closing this section we mention that the survey paper (ref. 20) discusses in detail MPE and RRE, as well as the well known scalar epsilon algorithm (SEA) of Wynn (ref. 23) and its two vector versions, the vector epsilon algorithm (VEA) of Wynn (ref. 24) and the topological epsilon algorithm (TEA) of Brezinski (ref. 2). SEA and VEA have been used in accelerating the convergence of some numerical schemes in computational fluid mechanics, see, e.g., Hafez et al. (ref. 7). As explained in reference 20 and in Ford and Sidi (ref. 6), the epsilon algorithms are expensive both storagewise and timewise. They need about twice as much storage as MPE, RRE, or MMPE, and their vector operation counts are several times those of MPE, RRE, or MMPE.

2. DESCRIPTION OF VECTOR EXTRAPOLATION METHODS

We shall now give a brief description of MPE, RRE, and MMPE based on the developments in references 20, 15, and 19.

Let $x_0, x_1, x_2, \ldots$, be a vector sequence in the complex N-dimensional Euclidean space C^N. Assume that this sequence converges, and denote its limit by s. Using the vectors x_i only, MPE, RRE, and MMPE produce approximations to s, which are determined as follows:

Define

$$u_i = \Delta x_i = x_{i+1} - x_i, \quad w_i = \Delta u_i = \Delta^2 x_i, \quad i = 0,1,2,\ldots. \tag{2.1}$$

MPE - Minimal Polynomial Extrapolation

Pick a positive integer $k \leq N$ and form the matrix U by

$$U = \left[u_n \,\vdots\, u_{n+1} \,\vdots\, \cdots \,\vdots\, u_{n+k-1}\right] \tag{2.2}$$

and solve the overdetermined (and in general inconsistent) system of equations

$$Uc = -u_{n+k}, \tag{2.3}$$

where $c = (c_0, c_1, \ldots, c_{k-1})^T$, by linear least squares. With $c_0, c_1, \ldots, c_{k-1}$ determined, set $c_k = 1$, and let

$$\gamma_j = c_j \Big/ \sum_{i=0}^{k} c_i, \quad j = 0,1, \ldots, k. \tag{2.4}$$

Finally set

$$s_{n,k} = \sum_{j=0}^{k} \gamma_j x_{n+j}, \tag{2.5}$$

where $s_{n,k}$ is the desired approximation to s. Note that

$$\sum_{j=0}^{k} \gamma_j = 1 \tag{2.6}$$

as is implied by equation (2.4), i.e., $s_{n,k}$ is a weighted "average" of the vectors $x_n, x_{n+1}, \ldots, x_{n+k}$, in which the weights are not necessarily real and nonnegative.

If we define the inner product associated with C^N to be

$$(y,z) = y^*z = \sum_{i=1}^{N} \overline{y^i} z^i, \tag{2.7}$$

where $y = \left(y^1, \ldots, y^N\right)^T$ and $z = \left(z^1, \ldots, z^N\right)^T$ are arbitrary vectors in C^N, then the c_i are equivalently the solution to the normal equations

$$\sum_{j=0}^{k-1} (u_{n+i}, u_{n+j})c_j = -(u_{n+i}, u_{n+k}), \quad i = 0,1, \ldots, k-1. \tag{2.8}$$

Finally, if we let U^+ be the generalized inverse of the matrix U, then the vector c is also given by

$$c = -U^+u_{n+k}. \tag{2.9}$$

RRE - Reduced Rank Extrapolation

Pick a positive integer $k \leq N$ and form the matrix W by

$$W = \left[w_n \,\vdots\, w_{n+1} \,\vdots\, \dots \,\vdots\, w_{n+k-1}\right] \tag{2.10}$$

and solve the overdetermined (and in general inconsistent) system of equations

$$Wq = -u_n, \tag{2.11}$$

where $q = \left(q_0, q_1, \dots, q_{k-1}\right)^T$, by linear least squares. With $q_0, q_1, \dots, q_{k-1}$ determined, set

$$s_{n,k} = x_n + \sum_{j=0}^{k-1} q_j u_{n+j}, \tag{2.12}$$

where $s_{n,k}$ is the desired approximation to s.

The q_i are equivalently the solution to the normal equations

$$\sum_{j=0}^{k-1} (w_{n+i}, w_{n+j})q_j = -(w_{n+i}, u_n), \quad i = 0,1, \dots, k-1. \tag{2.13}$$

Also, if W^+ is the generalized inverse of the matrix W, then the vector q is given by

$$q = -W^+u_n. \tag{2.14}$$

Interestingly enough, $s_{n,k}$ for RRE, just like $s_{n,k}$ for MPE, can be expressed as in equation (2.5), with equation (2.6) holding true. Specifically, the q_i and the corresponding γ_j for RRE are related by

$$\gamma_0 = 1 - q_0; \quad \gamma_j = q_{j-1} - q_j, \quad 1 \leq j \leq k-1; \quad \gamma_k = q_{k-1}. \tag{2.15}$$

MMPE - Modified Minimal Polynomial Extrapolation

Pick a positive integer $k \leq N$ and form the matrix U as in equation (2.2) and "solve" the overdetermined system of equations in equation (2.3) for $c = (c_0, c_1, \dots, c_{k-1})^T$, in the sense

$$QUc = -Qu_k, \tag{2.16}$$

where Q is a <u>fixed</u> $k \times N$ matrix of full rank. Obviously the matrix QU of equation (2.16) is $k \times k$. If $q_1^T, \dots, q_k^T$ are the rows of the matrix Q, then equation (2.16) can also be expressed as

$$\sum_{j=0}^{k-1} (q_i, u_{n+j})c_j = -(q_i, u_{n+k}), \quad 1 \leq i \leq k, \tag{2.17}$$

c.f. equation (2.8) for MPE. Now set $c_k = 1$ and determine the γ_j, $0 \leq j \leq k$, by equation (2.4), and $s_{n,k}$, by equation (2.5).

Determinantal representations for $s_{n,k}$ exist both for MPE, RRE, and MMPE, and they are of the form

$$s_{n,k} = \frac{D(x_n, x_{n+1}, \ldots, x_{n+k})}{D(1, 1, \ldots, 1)}, \tag{2.18}$$

where $D(\sigma_0, \sigma_1, \ldots, \sigma_k)$ is the determinant

$$D(\sigma_0, \sigma_1, \ldots, \sigma_k) = \begin{vmatrix} \sigma_0 & \sigma_1 & \cdots & \sigma_k \\ u_{0,0} & u_{0,1} & \cdots & u_{0,k} \\ \vdots & \vdots & & \vdots \\ u_{k-1,0} & u_{k-1,1} & \cdots & u_{k-1,k} \end{vmatrix} \tag{2.19}$$

and

$$u_{i,j} = \begin{cases} (u_{n+i}, u_{n+j}) & \text{for MPE,} \\ (w_{n+i}, u_{n+j}) & \text{for RRE,} \\ (q_{i+1}, u_{n+j}) & \text{for MMPE.} \end{cases} \tag{2.20}$$

When the σ_i are vectors $D(\sigma_0, \sigma_1, \ldots, \sigma_k)$ is also a vector, and is defined to be $\sum_{i=0}^{k} \sigma_i N_i$, where N_i is the cofactor of σ_i.

The approximations $s_{n,k}$ for all three methods are determined from the $k + 2$ vectors $x_n, x_{n+1}, \ldots, x_{n+k}$, as can easily be seen from their definitions.

As far as the implementation of MPE and RRE is concerned, a few alternatives exist:

(1) Solution by least squares packages: Although feasible for small scale problems, this approach may become very costly for large scale problems in which N, the dimension of the vectors, is very large and k is not small. We may even run into storage problems, as the present least squares solvers require the matrix of the equations to be solved (U for MPE and W for RRE) to be stored in core memory. One way out of this limitation would be to modify these solvers so that this matrix can be stored in a secondary storage device like a disk, for example. Additional storage for the $k + 2$ vectors $x_n, \ldots, x_{n+k+1}$ is also required.

(2) Solution of normal equations: Although the conditioning of this approach is less favorable than that for the previous one, the storage requirements for it are much lower. Actually, with appropriate programming, we can see that only $k + 2$ vectors, namely $x_n, u_n, u_{n+1}, \ldots, u_{n+k}$, need to be saved. That this is so can be shown as follows: Define $\tilde{u}_{i,j} \equiv (u_{n+i}, u_{n+j})$.

(a) For MPE the system of equations in equation (2.8) is entirely given in terms of the $\tilde{u}_{i,j}$. Once the c_i, hence γ_i have been obtained from equations (2.8) and (2.4), $s_{n,k}$ in equation (2.5) can be computed from

$$s_{n,k} = x_n + \sum_{i=0}^{k-1} \xi_i u_{n+i}, \tag{2.21}$$

where the ξ_i can be determined recursively from the γ_j through the relations

$$\xi_0 = 1 - \gamma_0; \quad \xi_j = \xi_{j-1} - \gamma_j, \quad j = 1,2, \ldots, k - 1, \tag{2.22}$$

or the relations

$$\xi_{k-1} = \gamma_k; \quad \xi_{j-1} = \gamma_j + \xi_j, \quad j = k - 1, k - 2, \ldots, 0. \tag{2.23}$$

(b) For RRE the system of equations in equation (2.13) is determined solely in terms of the $\tilde{u}_{i,j}$, since $(w_{n+i}, w_{n+j}) = \tilde{u}_{i+1,j+1} + \tilde{u}_{i,j} - \tilde{u}_{i+1,j} - \tilde{u}_{i,j+1}$. Once the q_j have been determined from equation (2.13), $s_{n,k}$ is obtained directly from equation (2.12).

(3) Recursive computation: Both MPE and RRE can be implemented by some recursive techniques that have recently been developed by Ford and Sidi (ref. 6). These techniques are based on the determinant representations of $s_{n,k}$ given in equations (2.18) to (2.20). For details we refer the reader to reference 6.

As for the implementation of MMPE, this can be done by either solving the linear $k \times k$ system in equation (2.17) for the c_i, $i = 0,1, \ldots, k-1$, and then proceeding as in equations (2.4) and (2.5), or by using the recursive techniques of reference 6.

The overhead in the implementation of MPE and RRE is largely due to the scalar products that are needed, c.f. equation (2.8) for MPE and equation (2.13) for RRE. To reduce this cost one might replace these inner products by a pseudo inner product involving only part of the components of the vectors x_j. In computational fluid mechanics problems this could be done by excluding some of the dependent variables from the inner product. In discretized continuum problems, in general, one can also do this by excluding some of the mesh points from the inner product.

The least expensive implementation in the above mentioned sense can be achieved by MMPE if the vectors q_i in equation (2.17) are picked such that (q_i, u_{n+j}) require almost no computation. This can be achieved by taking the q_i to be the fundamental basis vectors, which amounts to picking k out of the N equations in equation (2.3) for determining c_i, $i = 0,1, \ldots, k-1$.

One may, of course, consider other similar strategies, in which the vectors q_i have very few nonzero components.

Finally, we would like to mention some new extrapolation methods that have been proposed in reference 19 and are akin to MPE and RRE. These methods differ from MPE and RRE in the way the overdetermined systems of equations (2.3) and (2.11) are solved. The standard linear least squares method minimizes the ℓ_2-norm of $Uc + u_{n+k}$ for MPE and of $Wq + u_n$ for RRE, where this norm is defined by $||z|| = \sqrt{(z,z)}$ with (y,z) as defined in equation (2.7). We can modify this norm to read $||z||_M = \sqrt{(z,Mz)}$ for some hermitean positive definite matrix M. Going further, we can choose to minimize the (weighted) ℓp-norms of $Uc + u_{n+k}$ and $Wq + u_n$. In particular, the ℓ_1- and ℓ_∞-norms give rise to problems that can be solved using linear programming techniques.

3. CONVERGENCE AND STABILITY OF VECTOR EXTRAPOLATION METHODS

We now state without proof some results concerning the convergence and stability properties of MPE, RRE, and MMPE. These results are helpful in understanding how these methods work, and how and when they can be used in an effective manner. All our results are stated for those sequences that arise from iterative solutions of linear systems of equations.

Let s be the unique solution to the linear system

$$x = Ax + b, \tag{3.1}$$

where A is a constant $N \times N$ matrix, and b is a constant vector in C^N. Given x_0, an initial approximation to s, generate the vectors x_1, x_2, . . . , by the iterative method

$$x_{j+1} = Ax_j + b, \quad j = 0,1, \ldots . \tag{3.2}$$

Let $\lambda_1, \lambda_2, \ldots, \lambda_r$ be the distinct eigenvalues of the matrix A, and let $v_1, v_2, \ldots, v_r$ be corresponding eigenvectors. Obviously $r \leq N$. Assume for simplicity that A is diagonalizable so that the initial error $x_0 - s$ can be expressed in the form

$$x_0 - s = \sum_{i=1}^{r} \alpha_i v_i \tag{3.3}$$

for some scalars α_i. Consequently

$$x_n - s = \sum_{i=1}^{r} \alpha_i v_i \lambda_i^n, \quad n = 1,2, \ldots . \tag{3.4}$$

The assumption that equation (3.1) has a unique solution implies that $\lambda_i \neq 1$ for all i. Without loss of generality we can further assume that (1) $\lambda_i \neq 0$ for all i, since a zero eigenvalue does not contribute to equation (3.4) for $n \geq 1$, and (2) $\alpha_i \neq 0$ for all i, since if $\alpha_i = 0$ for $i = i'$, then we can discard it from equation (3.4) and rename the eigenvalues.

Let us now order the λ_i such that

$$|\lambda_1| \geq |\lambda_2| \geq |\lambda_3| \geq \cdots . \tag{3.5}$$

We now state a convergence theorem for MPE, RRE, and MMPE, whose proof can be found in references 15 and 19.

Theorem 3.1. Assume

$$|\lambda_k| > |\lambda_{k+1}|. \tag{3.6}$$

Then, for both MPE and RRE, the approximations $s_{n,k}$ to s satisfy

$$s_{n,k} - s = \Gamma(n)|\lambda_{k+1}|^n, \tag{3.7}$$

where the vector $\Gamma(n)$ is such that

$$||\Gamma(n)|| = O(1) \quad \text{as} \quad n \to \infty, \tag{3.8}$$

and is proportional to $\prod_{i=1}^{k} (\lambda_i - 1)^{-1}$. Furthermore, if we denote the γ_i in equations (2.5) by $\gamma_i^{(n,k)}$ to stress their dependence on n and k, then

$$\sum_{i=0}^{k} \gamma_i^{(n,k)} \lambda^i = \prod_{i=1}^{k} \frac{\lambda - \lambda_i}{1 - \lambda_i} + O\left(\left|\frac{\lambda_{k+1}}{\lambda_k}\right|^n\right) \quad \text{as} \quad n \to \infty. \tag{3.9}$$

Equations (3.7) to (3.9) hold also for MMPE provided

$$\begin{vmatrix} (q_1,v_1) & \cdots & (q_1,v_k) \\ \vdots & & \vdots \\ (q_k,v_1) & \cdots & (q_k,v_k) \end{vmatrix} \neq 0. \tag{3.10}$$

Note that the results stated in Theorem 3.1 concern the strategy in which first a large number of iterations have been performed through equation (3.2) and then MPE or RRE or MMPE had been applied with fixed k. As such, Theorem 3.1 has the following important implications:

(1) If MPE or RRE or MMPE is applied to the vector sequence $x_0, x_1, \ldots$, beginning with x_n, for large n, then provided (3.6) holds, the error in $s_{n,k}$, the desired approximation, behaves like $|\lambda_{k+1}|^n$. In view of the fact that the error in x_{n+k+1} behaves like $|\lambda_1|^n$ for n large, and $|\lambda_1| > |\lambda_{k+1}|$, all three methods accelerate the convergence of the sequence $x_0, x_1, x_2, \ldots$, when the latter converges. Furthermore, even when this sequence does not converge, i.e., $|\lambda_1| \geq 1$, $s_{n,k}$ converges to s as n becomes large, provided $|\lambda_{k+1}| < 1$.

In some applications the matrix A may have several distinct dominant eigenvalues that are all equal to the spectral radius $\rho(A)$ in modulus. In order to achieve acceleration in such cases we need to take k at least equal to the number of these eigenvalues.

(2) If some of the λ_i are very close to 1 as is the case in many problems of interest, then the quality of $s_{n,k}$ may deteriorate, as the error in $s_{n,k}$ is proportional to $\prod_{i=1}^{k} (\lambda_i - 1)^{-1}$ for n large. In fact, the computation of $s_{n,k}$ may become numerically unstable in this case as is seen from equation (3.9). Specifically, the coefficients of the polynomial $\prod_{i=1}^{k} [(\lambda - \lambda_i)/(1 - \lambda_i)]$ are large when some of $\lambda_1, \ldots, \lambda_k$ are very close to 1 in the complex plane. Thus the $\gamma_i^{(n,k)}$, being approximations to these coefficients, are large too, although their sum equals 1, c.f. equation (2.6). This means that errors (round-off, in general) in the vectors x_i are magnified severely in the computation of $s_{n,k}$, resulting in instability. One suitable way of overcoming this problem is to apply MPE, RRE, or MMPE to the sequence $y_j = x_{pj}$, $j = 0,1, \ldots,$ p being an integer greater than 1. With this equation (3.4) is now replaced by

$$y_n - s = \sum_{i=1}^{r} \alpha_i v_i \mu_i^n , \quad n = 1,2, \ldots , \tag{3.4)'}$$

where $\mu_i = \lambda_i^p$, $i = 1,2,\ldots$. By picking p large enough we can cause $\mu_1, \ldots, \mu_k$ to be far from 1 in the complex plane, thus stabilizing the convergence acceleration methods. It is interesting to note that equation (3.2) becomes

$$y_{j+1} = A^{pj} y_j + \sum_{i=0}^{p-1} A^i b, \quad j = 0,1, \ldots . \tag{3.2)'}$$

(3) The already computed γ_j, $0 \le j \le k$, can be used for estimating $\lambda_1, \lambda_2, \ldots, \lambda_k$, the largest eigenvalues of A. In fact, the zeros of the polynomial $\sum_{j=0}^{k} \gamma_j \lambda^j$ are the estimates of interest. If we denote these zeros by $\lambda_1(n), \ldots, \lambda_k(n)$, then with appropriate ordering

$$\lambda_j(n) - \lambda_j = O\left(\left|\frac{\lambda_{k+1}}{\lambda_j}\right|^n\right) \quad \text{as} \quad n \to \infty, \quad 1 \le j \le k, \tag{3.11}$$

see reference 17.

We note that the results of Theorem 3.1 and equation (3.11) do not hold as they are, in general, when the matrix A is not diagonalizable, although similar but slightly complicated results for this case exist. For the precise statements and proofs of these results see reference 18.

As has already been mentioned, Theorem 3.1 above explains the convergence and stability properties of MPE, RRE, and MMPE when k is held fixed and n is increasing. Another obvious use of these methods is one in which n is kept fixed and k is increasing. An error analysis for this use, pertaining to MPE and RRE, has been given in reference 16. Part of this analysis is summarized in the following theorem. For simplicity we fix $n = 0$ and consider the sequence $s_k \equiv s_{0,k}$, $k = 0,1,2, \ldots$, with $s_0 \equiv s_{0,0} = x_0$.

Theorem 3.2 Let the matrix $C = I - A$ be such that $C_h = (C + C^*)/2$, the hermitian part of C, is positive definite. (This implies that all eigenvalues of C have positive real parts.) Then for some appropriate norms $||\bullet||''$

$$||s_k - s||'' \leq L\Gamma_k \, ||s_0 - s||'', \tag{3.12}$$

where L is a constant independent of k, dependent on the norm and the extrapolation method, and Γ_k is the minimum of $||Q_k(C)||$, the natural matrix ℓ_2-norm of $Q_k(C)$, over all polynomials $Q_k(z)$ of degree at most k satisfying $Q_k(0) = 1$. Let now $F(d,c,a)$ be the smallest ellipse that contains all the eigenvalues of C and is itself contained in the right half of the complex plane, such that its center is at d, its foci are at $d \pm c$, and its semimajor axis length is a. Then Γ_k in (3.12) can be bounded as

$$\Gamma_k \leq \beta k^m \eta^k, \tag{3.13}$$

where β is a constant independent of k, m is a fixed nonnegative integer, and η is given by

$$\begin{gathered} \eta = \exp(\phi - \mathrm{Re}w) < 1 \\ \text{with} \quad \phi = \cosh^{-1}(a/|c|) \quad \text{and} \quad w = \cosh^{-1}(d/c). \end{gathered} \tag{3.14}$$

(When the matrix C(or A) is diagonalizable $m = 0$.) Consequently, we can replace (3.12) by

$$||s_k - s||'' \leq Tk^m\eta^k \, ||s_0 - s||'', \; T \equiv L\beta. \tag{3.15}$$

Connections with Krylov Subspace Methods

It is furthermore shown in reference 16 that MPE and RRE are Krylov subspace methods and that they are equivalent to the Arnoldi method and the method of generalized conjugate residuals respectively, when the latter are being used for solving the equations $Cx = b$ with the matrix C as in Theorem 3.2. Recall that when C is hermitian the Arnoldi method reduces to the method of conjugate gradients and the method of generalized conjugate residuals reduces to the method of conjugate residuals. In this sense then MPE is a generalization of the Arnoldi method, which in turn, is a generalization of the method of conjugate gradients. Similarly, RRE is a generalization of the method of generalized conjugate residuals, which in turn, is a generalization of the method of conjugate residuals. For details and the relevant literature see reference 16. Although MPE and RRE can be applied in a very straightforward way to the vector sequence obtained by a fixed point iteration technique, for nonlinear as well as linear problems, the other methods require some kind of local linearization for nonlinear problems that may increase the cost of acceleration depending on how the linearization is performed.

From what has been said in the previous section it is obvious that we would not be interested in increasing k indefinitely as this would require an increasing amount of storage. From Theorem 3.2 however, it is easy to see that the following strategy, which has been designated "cycling" in reference 20 is useful in solving a system of the form $x = F(x)$:

Step 1. Pick $s_k^{(0)} \equiv x_0$ arbitrarily and set $q = 1$.

Step 2. Generate $x_1, \ldots, x_{k+1}$ by $x_{j+1} = F(x_j)$, $j = 0,1, \ldots, k$.

Step 3. Use MPE, RRE, or MMPE to compute $s_k^{(q)}$ from $x_0, x_1, \ldots, x_{k+1}$.

Step 4. If $s_k^{(q)}$ is a satisfactory approximation, stop; otherwise, replace x_0 by $s_k^{(q)}$, and q by $q+1$, and go to Step 2.

If $F(x)$ is a linear function of x, then Theorem 3.2 can be used to prove that

$$\left\| s_k^{(q)} - s \right\|'' \leq \left(L\Gamma_k \right)^q \left\| s_k^{(0)} - s \right\|'' . \qquad (3.16)$$

This implies that taking k sufficiently large we can cause $L\Gamma_k \leq Tk^{m}\eta^k < 1$. This, by (3.16), guarantees the convergence of cycling. (3.16) for this case also suggests that cycling is a linearly converging process.

In case $\lambda_1, \ldots, \lambda_k$ are very close to 1 we can modify Steps 2 and 3 to read

Step 2'. Generate $y_0, y_1, \ldots, y_{k+1}$ by $x_{j+1} = F(x_j)$, $j = 0,1, \ldots, p(k+1) - 1$, and $y_j = x_{pj}$, $j = 0,1, \ldots, k+1$.

Step 3'. Use MPE, RRE, or MMPE to compute $s_k^{(q)}$ from $y_0, y_1, \ldots, y_{k+1}$.

This helps to stabilize the extrapolation process.

4. APPLICATIONS

We shall now mention some applications to computational fluid mechanics problems. In this respect the following remarks are important.

(1) Practically no acceleration is achieved for problems that use explicit schemes. The addition of even a small amount of implicitness, such as the implicit residual averaging (ref. 8), makes the scheme quite suitable for acceleration. We note that the implicit residual averaging is a simple device that acts like a filter on the approximate solution produced by the explicit scheme, and can be added to the scheme without making any changes in it.

(2) The vectors that are processed by the extrapolation methods must include the boundary values as a rule. This is especially so for problems in which the boundary values are time-dependent. Going back to the discussion of our introduction the discrete operator ψ in equation (1.3) also contains the boundary values when these are time-dependent. Thus the boundary values influence the γ_j in the extrapolation process, and are influenced by the γ_j. (Time-independent boundary conditions, however, neither influence, nor are influenced by, the γ_j in the extrapolation methods.)

(3) As they are described in the previous sections, the extrapolation methods operate on the vectors $x_0, x_1, \ldots,$ in a global manner, i.e., the γ's in equation (2.5) are the same for all components of the vectors x_j. In some applications we may want to apply these methods to different portions of the vectors x_j separately, and this may produce more acceleration in some problems. For computational fluid mechanics this may amount to accelerating each of the dependent variables separately.

Example 1. Ducted Flow Over an Axisymmetric Center Body

The duct is cylindrical in shape and the center body is the hub of that found on the General Electric unducted fan. The flow is assumed to be inviscid, and thus the Euler equations are solved. In addition, the flow is assumed to be axisymmetric.

This problem was solved by using two three-dimensional codes that employed the same mesh.

The first code uses an explicit finite volume four-stage Runge Kutta scheme patterned after Jameson, Schmidt, and Turkel (ref. 9). The dependent variables for this code are ρu, ρv, ρw, ρ, ρe_0, where u, v, and w are the axial, radial, and circumferential components of the velocity in cylindrical coordinates, ρ is the density, and e_0 is the total internal energy per unit volume. The scheme also contains artificial viscosity. It is supplemented by local time-stepping and by implicit residual averaging in the axial and radial directions. For a proper description of this scheme see reference 4.

The second code uses an implicit finite volume flux-vector splitting scheme. The dependent variables for this scheme are ρu, ρv, ρw, ρ, and ρe_0 as in the first code, except that now u, v, and w stand for the components of the velocity in Cartesian coordinates. For a proper description of the scheme used in this code see reference 1.

The mesh used by both codes consists of 56 points in the axial direction and 7 points in the radial direction. The first code also requires a minimum of 5 points in the circumferential direction as dictated by the fourth order difference employed in representing the artificial viscosity. We chose to have 7 equally spaced points in the circumferential direction their spacing being $2\pi/56$ rads. If periodicity is imposed in the numerical solution of an axisymmetric problem, then there should be no variation in the circumferential direction. This was achieved to machine accuracy in the iterative scheme used in the first code. The mesh employed was generated algebraically as described in reference 13, and it is shown in figure 1.

Both codes were run at a Mach number of 0.60, where the critical Mach number for this problem is approximately 0.62.

In figures 2(a) to 3(b) there are six curves numbered 1, 2, . . . , 6. Curve number 1 is for the ℓ_2-norm of the error associated with all five dependent variables. Curves number 2, 3, . . . , 6, are for the ℓ_2-norms of the errors associated with ρ, ρu, ρv, ρw, and ρe_0 respectively.

Figure 2(a) shows the results obtained by using the first code without extrapolation. As is seen the residual history gets into the linear regime after 1000 iterations. Figure 2 shows the results obtained by the first code with extrapolation being applied only once to the sequence $x_n, x_{n+p}, x_{n+2p}, \ldots, x_{n+(k+1)p}$ with $n = 700$, $p = 10$, and $k = 20$. The approximation obtained by the extrapolation method is then used as the new x_0 and new vectors are obtained from it by iteration. We see that with only one extrapolation the ℓ_2-norm of the error has dropped from about $10^{-4.3}$ to about $10^{-6.8}$, resulting in a gain of about 2.5 orders of magnitude. The jump in the ℓ_2-norm of the residual vector for ρw, to circumferential momentum per unit volume, is a result of $\sum_{j=0}^{k} |\gamma_j|$ being relatively large. This can be explained as follows: Theoretically ρw is supposed to be zero. Because we are using a three-dimensional code we can achieve at best machine zero for ρw, and this is the case for the first code. Thus the residual vectors associated with ρw are at best machine zero. By equation (2.5) then ρw and hence its residual are of order $\left(\sum_{j=0}^{k} |\gamma_j|\right) \times \varepsilon$, where ε is the norm of the residual for ρw. Figure 2(c) shows the results obtained from the first code with extrapolation in the cycling mode starting at the 100th iteration, with $p = 10$ and $k = 20$ again. We again observe a substantial amount of acceleration. However, the amount of acceleration now is smaller than that we observe in figure 2(b). The reason for this may be that we are applying the extrapolation method much before the residual history reaches the linear regime. The jumps in the ℓ_2-norm of the residual vector associated with ρw can be explained as above.

Figure 3(a) shows the results obtained by using the second code without extrapolation. Figure 3(b) shows the results obtained by the second code with extrapolation in the cycling mode starting at the 10th iteration with $p = 1$ and $k = 10$. We see from this figure that with the second code (involving implicit flux-vector splitting) cycling, even when started very early in the iteration process, is very effective in this case. A reduction of approximately 9 orders of magnitude is achieved without extrapolation in 1000 iterations, whereas with extrapolation this takes only 500 iterations. Also machine zero is reached in about 1400 iterations without extrapolation and in about 800 with extrapolation.

Before closing we also mention that we assessed the rates of convergence of both codes by estimating the largest eigenvalues of the Jacobian matrix $\psi'(\tilde{u})$. We recall that this is done by solving the polynomial equation $\sum_{j=0}^{k} \gamma_j \lambda^j = 0$, where the γ's are those associated with $s_{n,k}$ for n sufficiently large, c.f. equations (3.9) and (3.11). The modulus of the largest zero of the polynomial $\sum_{j=0}^{k} \gamma_j \lambda^j$ is an estimate for $\rho[\psi'(\tilde{u})]$, the spectral radius of $\psi'(\tilde{u})$. The information on the largest eigenvalues of $\psi'(\tilde{u})$ is then also used in deciding what values to take for p and k.

Example 2. Flow Over a Single-Blade Row Unducted Fan

The geometry consists of an eight-bladed unducted fan designed to operate at a free-stream Mach number of 0.80 and advance ratio 3.1145. The hub for this fan is the one that was considered in Example 1.

We again assume the flow to be inviscid, thus we are solving the Euler equations in this case too.

The code used in this example is the first code that was used in Example 1. However, this time the implicit residual averaging is used in all three directions.

The mesh used by the code consists of 88 points in the axial direction, 23 points in the radial direction, and 11 points in the circumferential direction across one blade pitch. Ten points lie on each blade axially, and 11 points, radially from the hub to the blade tip. A sting whose diameter is equal to the sting at the hub exit is affixed to the front of the nacelle. The mesh is generated algebraically as described in reference 13, and is shown in detail in reference 4.

The six curves shown in figures 4(a) and 4(b) and numbered 1, 2, . . . , 6, again correspond respectively to the ℓ_2-norms of the residuals associated with all five dependent variables, with ρ, ρu, ρv, ρw, and ρe_0 respectively.

Figure 4(a) shows the residuals obtained without extrapolation. As is seen the residuals drop by about 2.5 orders of magnitude in 1000 iterations. Figure 4(b) shows the residuals obtained by employing RRE in the cycling mode starting at the 100th iteration with $p = 10$ and $k = 20$. The amount of acceleration achieved by doing this is quite substantial, in that in 1000 iterations the residuals drop by 4.5 orders of magnitude.

ACKNOWLEDGMENT

We thank Prof. D.L. Whitfield of Mississippi State University for making available to us his flux-vector splitting Euler-solver that was referred to as the "second code" in Example 4.1 of this work.

REFERENCES

1. Belk, D.M.; and Whitfield, D.L.: Three-Dimensional Euler Solutions on Blocked Grids Using an Implicit Two-Pass Algorithm. AIAA Paper 87-0450, AIAA 25th Aerospace Sciences Meeting, Reno, Nevada, 1987.

2. Brezinski, C.: Generalisations de la Transformation de Shanks, de la Table de Padé et de l'epsilon-algorithme. Calcolo, vol. 12, 1975, pp. 317-360.

3. Cabay, S.; and Jackson, L.W.: A Polynomial Extrapolation Method for Finding Limits and Antilimits of Vector Sequences. SIAM J. Numer. Anal., vol. 13, 1976, pp. 734-752.

4. Celestina, M.L.; Mulac, R.A.; and Adamczyk, J.J.: A Numerical Simulation of the Inviscid Flow Through a Counterrotating Propeller. ASME J. Turbomachinery, vol. 108, 1986, pp. 187-193.

5. Eddy, R.P.: Extrapolating to the Limit of a Vector Sequence. Information Linkage Between Applied Mathematics and Industry, P.C.C. Wang, ed., Academic Press, 1979, pp. 387-396.

6. Ford, W.F.; and Sidi, A.: Recursive Algorithms for Vector Extrapolation Methods. Appl. Numer. Math., vol. 4, 1988, in press.

7. Hafez, M.; Palaniswamy, S.; Kuruvilla, G.; and Salas, M.D.: Applications of Wynn's Epsilon-Algorithm to Transonic Flow Calculations. AIAA Paper 87-1143, AIAA 8th Computational Fluid Dynamics Conference, Honolulu, Hawaii, 1987.

8. Jameson, A.; and Baker, T.J.: Solution of the Euler Equations for Complex Configurations. AIAA Paper 83-1929, AIAA 6th Computational Fluid Dynamics Conference, Danvers, Massachusetts, 1983.

9. Jameson, A.; Schmidt, W.; and Turkel, E.: Numerical Solutions of the Euler Equations by Finite Volume Methods Using Runge-Kutta Time-Stepping Schemes. AIAA Paper 81-1259, AIAA 14th Fluid and Plasma Dynamics Conference, Palo Alto, California, 1981.

10. Jespersen, D.C.; and Buning, P.G.: Accelerating an Iterative Process by Explicit Annihilation. SIAM J. Sci. Stat. Comput., vol. 6, 1985, pp. 639-651.

11. Kaniel, S.; and Stein, J.: Least-Square Acceleration of Iterative Methods for Linear Equations. J. Optim. Theory Appl., vol. 14, 1974, pp. 431-437.

12. Mesina, M.: Convergence Acceleration for the Iterative Solution of the Equations X=AX+f. Comput. Methods Appl. Mech. Eng., vol. 10, 1977, pp. 165-173.

13. Mulac, R.A.: A Multistage Mesh Generator for Solving the Average-Passage Equation System. NASA CR-179539, 1988.

14. Reddy, K.C.; and Jacocks, J.L.: A Locally Implicit Scheme for the Euler Equations. AIAA Paper 87-1144, AIAA 8th Computational Fluid Dynamics Conference, Honolulu, Hawaii, 1987.

15. Sidi, A.: Convergence and Stability Properties of Minimal Polynomial and Reduced Rank Extrapolation Algorithms. SIAM J. Numer. Anal., vol. 23, 1986, pp. 197-209.

16. Sidi, A.: Extrapolation Vs. Projection Methods for Linear Systems of Equations. J. Comput. Appl. Math., vol. 22, 1988, pp. 71-88.

17. Sidi, A.: unpublished research, in preparation.

18. Sidi, A.; and Bridger, J.: Convergence and Stability Analyses for Some Vector Extrapolation Methods in the Presence of Defective Iteration Matrices. J. Comput. Appl. Math., vol. 22, 1988, pp. 35-61.

19. Sidi, A.; Ford, W.F.; and Smith, D.A.: Acceleration of Convergence of Vector Sequences. SIAM J. Numer. Anal., vol. 23, 1986, pp. 178-196.

20. Smith, D.A.; Ford, W.F.; and Sidi, A.: Extrapolation Methods for Vector Sequences. SIAM Rev., vol. 29, 1987, pp. 199-233. (See also: Correction to "Extrapolation Methods for Vector Sequences, SIAM Rev., to appear.)

21. Wigton, L.B.; Yu, N.J.; and Young, D.P.: GMRES Acceleration of Computational Fluid Dynamics Codes. AIAA Paper 85-1494, AIAA 7th Computational Fluid Dynamics Conference, Cincinnati, Ohio, 1985.

22. Wong, Y.S.; and Hafez, M.: Conjugate Gradients Methods Applied to Transonic Finite Difference and Finite Element Calculations. AIAA Journal, vol. 20, 1982, pp. 1526-1534.

23. Wynn, P.: On a Device for Computing the $e_m(S_n)$ Transformation. Math. Tables Aids Comput., vol. 10, 1956, pp. 91-96.

24. Wynn, P.: Acceleration Techniques for Iterated Vector and Matrix Problems. Math. Comput., vol. 16, 1962, pp. 301-322.

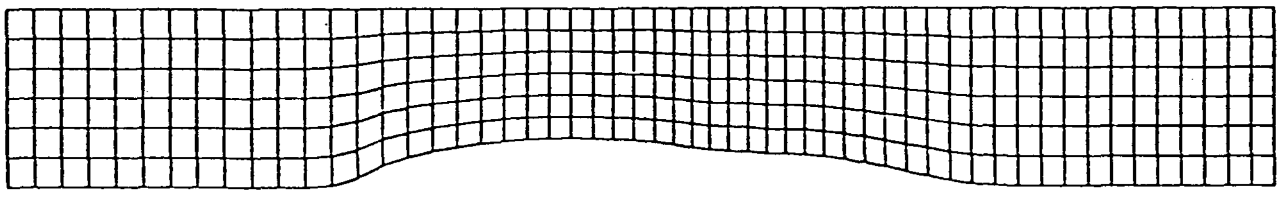

FIG. 1 AXISYMMETRIC CENTER BODY GRID FOR EXAMPLE 1.

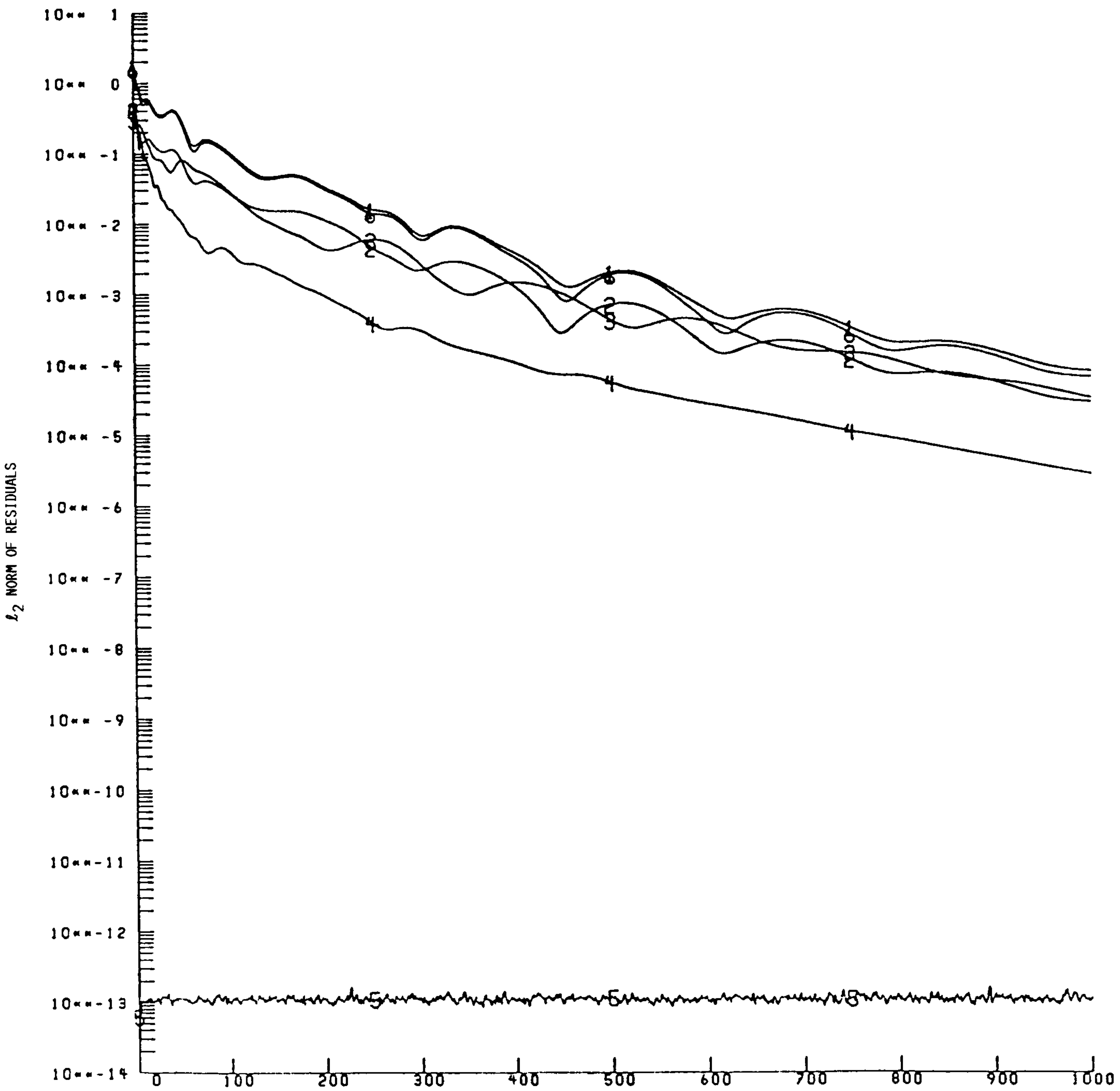

FIG. 2A RESIDUAL HISTORY FOR EXAMPLE 4.1 USING THE FIRST CODE WITH NO EXTRAPOLATION.

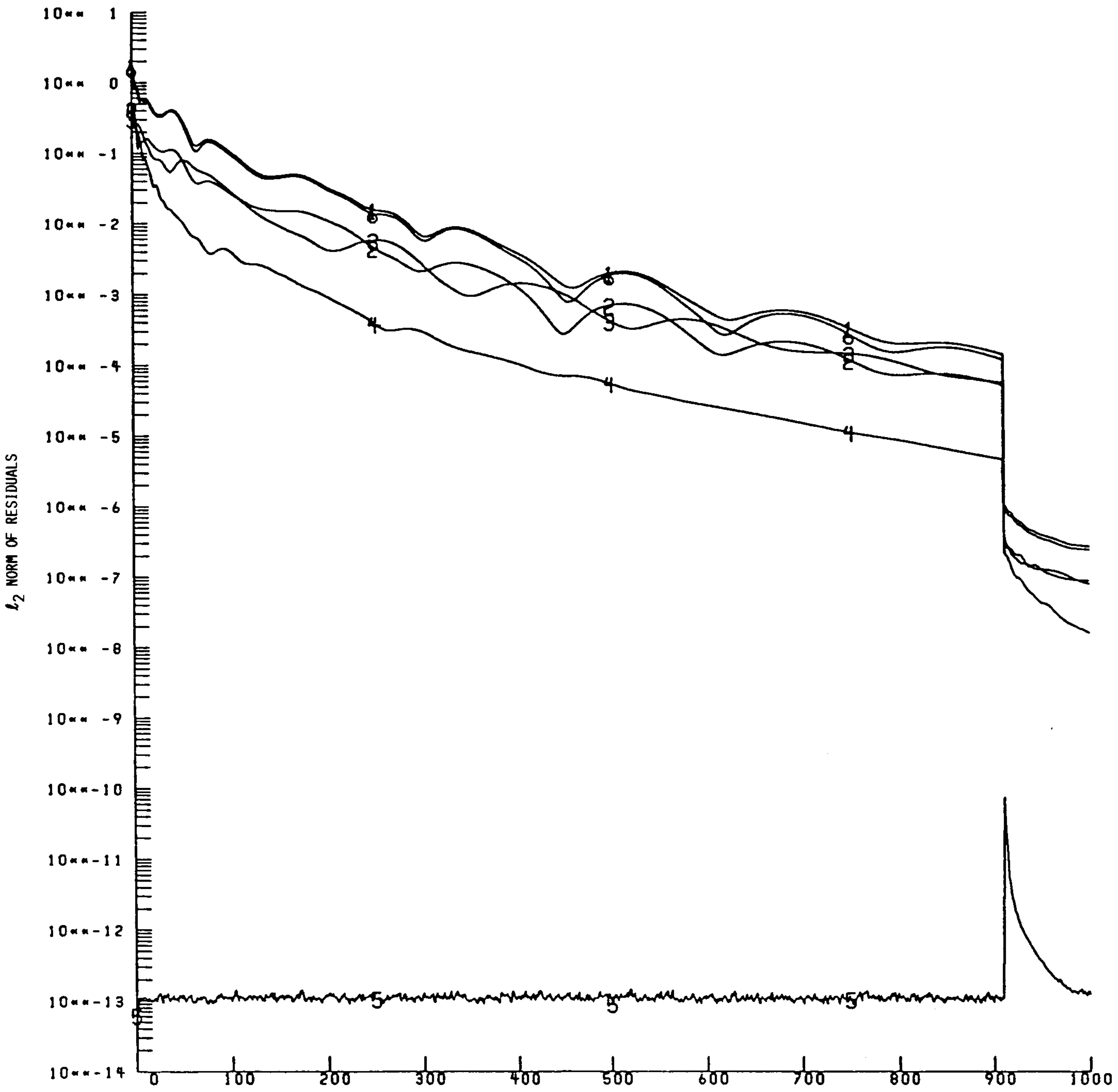

FIG. 2B RESIDUAL HISTORY FOR EXAMPLE 4.1 USING THE FIRST CODE WITH ONE APPLICATION OF MPE AT THE 700TH ITERATION AND WITH p = 10 AND k = 20.

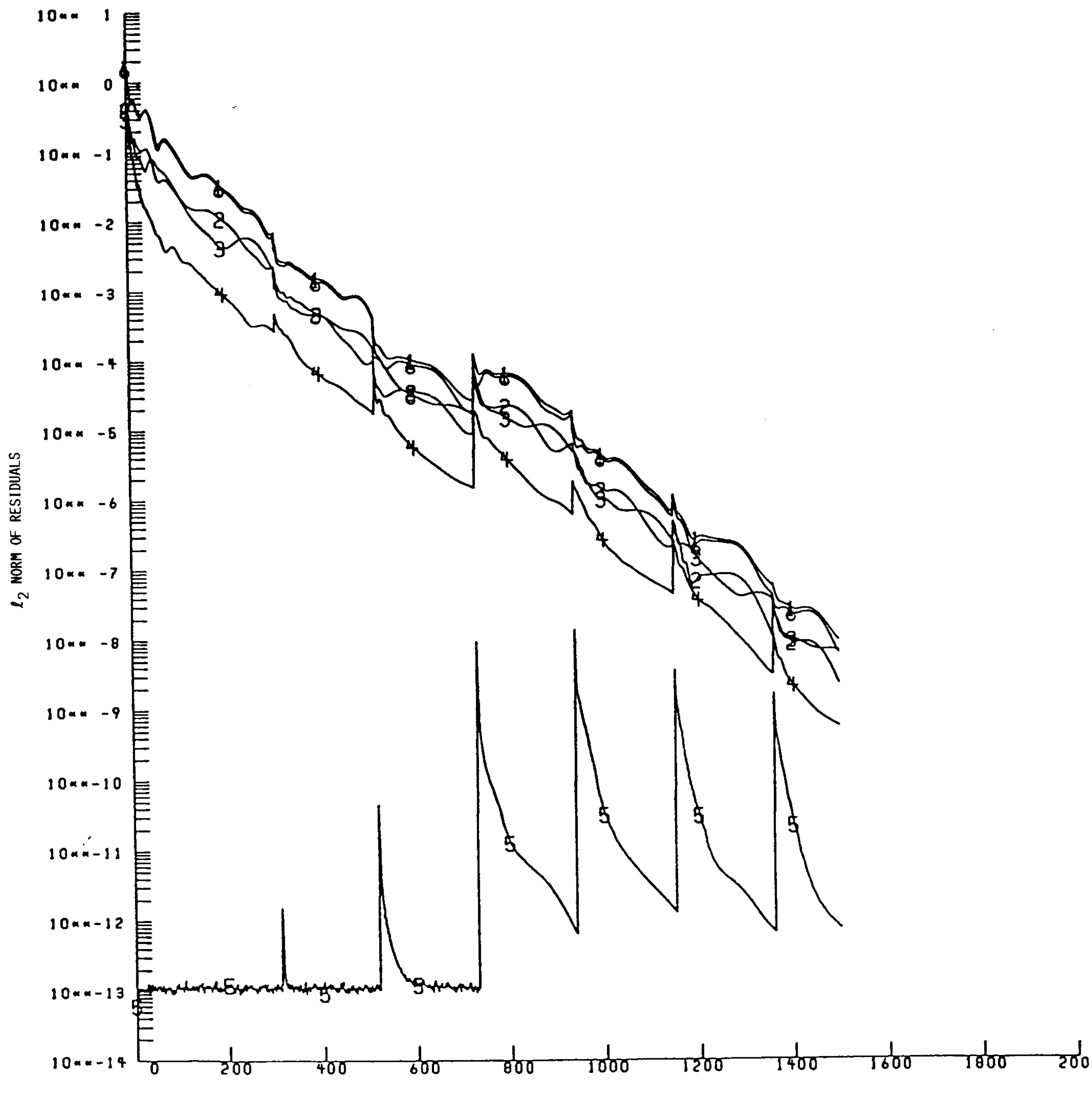

FIG. 2C RESIDUAL HISTORY FOR EXAMPLE 4.1 USING THE FIRST CODE IN CONJUNCTION WITH MPE IN THE CYCLING MODE WITH p = 10 AND k = 20. MPE IS APPLIED STARTING WITH THE 100TH ITERATION.

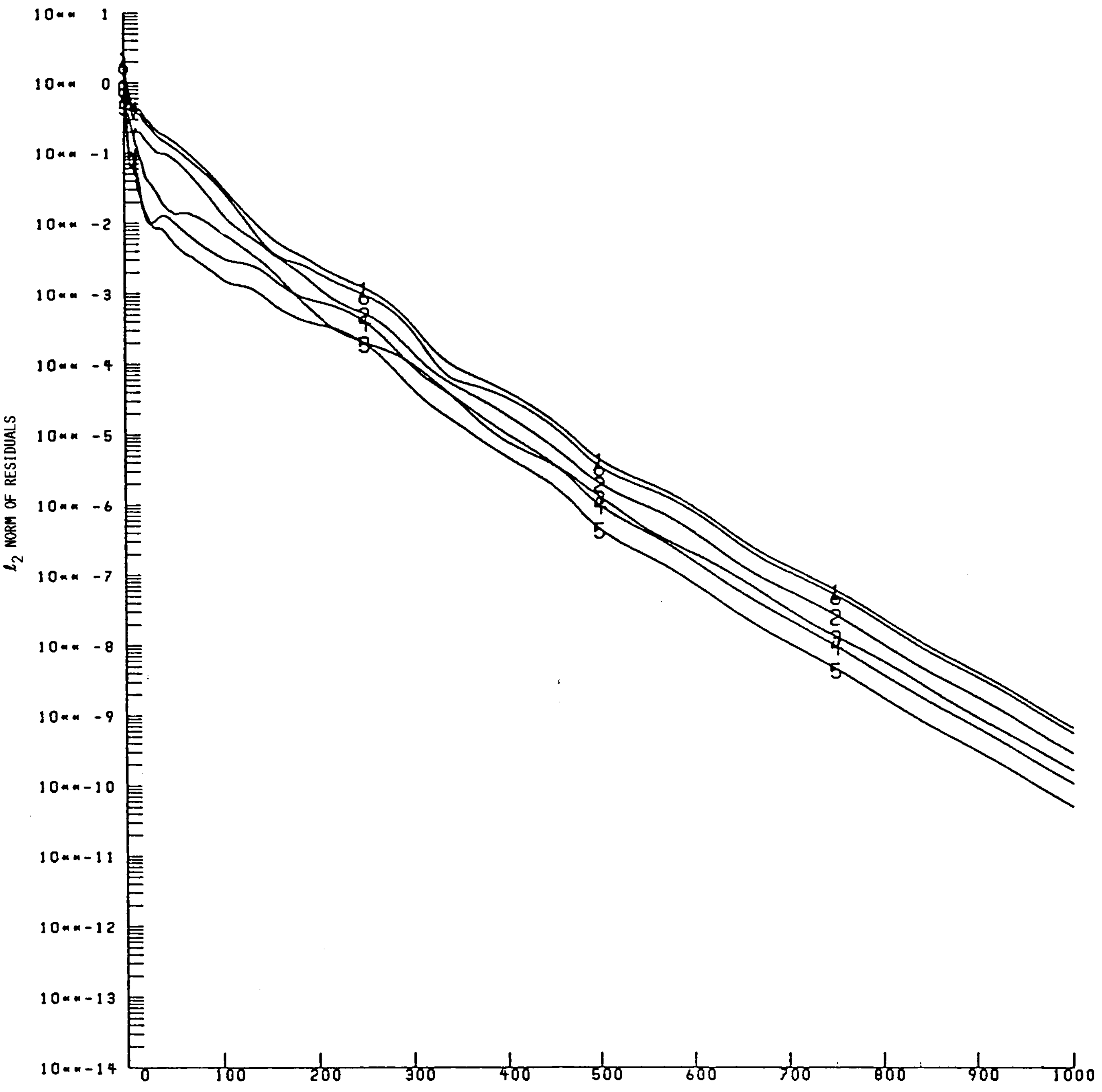

FIG. 3A RESIDUAL HISTORY FOR EXAMPLE 4.1 USING THE SECOND CODE WITH NO EXTRAPOLATION.

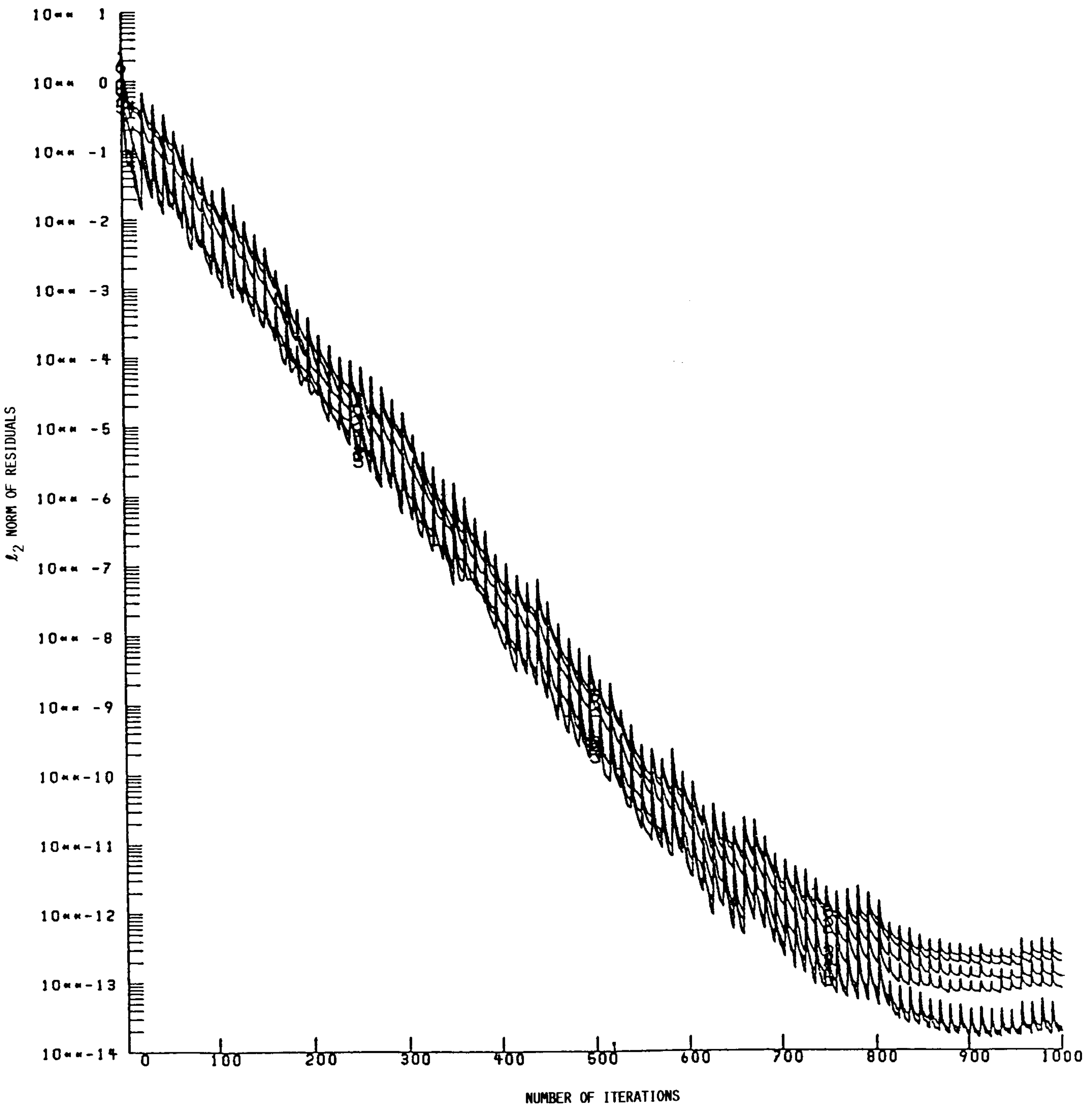

FIG. 3B RESIDUAL HISTORY FOR EXAMPLE 4.1 USING THE SECOND CODE IN CONJUNCTION WITH MPE IN THE CYCLING MODE WITH p = 1 AND k = 10. MPE IS APPLIED STARTING WITH THE 10TH ITERATION.

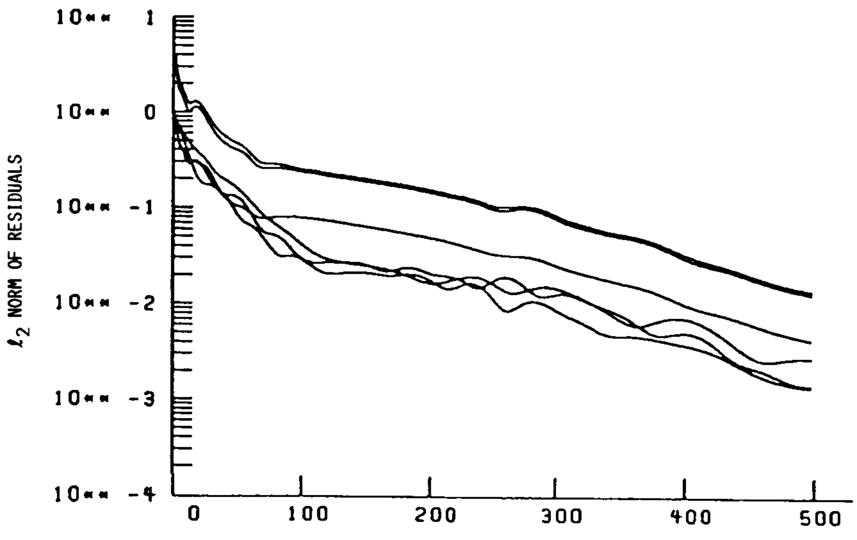

FIG. 4A RESIDUAL HISTORY FOR EXAMPLE 4.2 WITH NO EXTRAPOLATION.

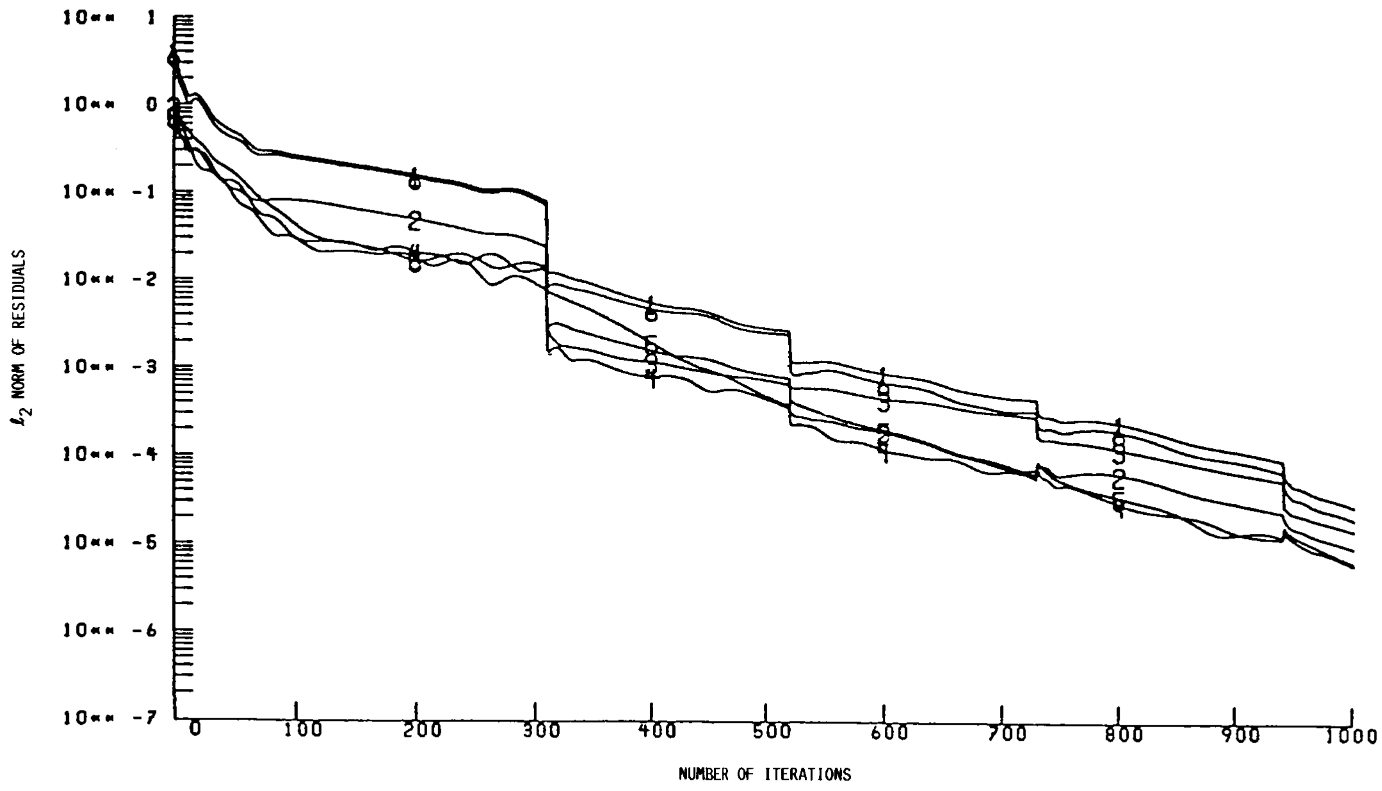

FIG.4B RESIDUAL HISTORY FOR EXAMPLE 4.2 IN CONJUNCTION WITH RRE IN THE CYCLING MODE WITH p = 10 AND k = 20. RRE IS APPLIED STARTING WITH THE 100TH ITERATION.

NASA
National Aeronautics and Space Administration

Report Documentation Page

1. Report No. NASA TM-101327 ICOMP-88-17	2. Government Accession No.	3. Recipient's Catalog No.
4. Title and Subtitle Convergence Acceleration for Vector Sequences and Applications to Computational Fluid Dynamics		5. Report Date August 1988
		6. Performing Organization Code
7. Author(s) Avram Sidi and Mark L. Celestina		8. Performing Organization Report No. E-4338
		10. Work Unit No. 505-62-21
9. Performing Organization Name and Address National Aeronautics and Space Administration Lewis Research Center Cleveland, Ohio 44135-3191		11. Contract or Grant No.
		13. Type of Report and Period Covered Technical Memorandum
12. Sponsoring Agency Name and Address National Aeronautics and Space Administration Washington, D.C. 20546-0001		14. Sponsoring Agency Code

15. Supplementary Notes

Avram Sidi, Institute for Computational Mechanics in Propulsion, NASA Lewis Research Center (work funded under Space Act Agreement C99066G); present address: Computer Science Dept., Technion—Israel Institute of Technology, Haifa, Israel. Mark L. Celestina, Sverdrup Technology, Inc., NASA Lewis Research Center Group, Cleveland, Ohio 44135.

16. Abstract

Some recent developments in acceleration of convergence methods for vector sequences are reviewed. The methods considered are the minimal polynomial extrapolation, the reduced rank extrapolation, and the modified minimal polynomial extrapolation. The vector sequences to be accelerated are those that are obtained from the iterative solution of linear or nonlinear systems of equations. The convergence and stability properties of these methods as well as different ways of numerical implementation are discussed in detail. Based on the convergence and stability results, strategies that are useful in practical applications are suggested. Two applications to computational fluid mechanics involving the three-dimensional Euler equations for ducted and external flows are considered. The numerical results demonstrate the usefulness of the methods in accelerating the convergence of the time-marching techniques in the solution of steady-state problems.

17. Key Words (Suggested by Author(s)) Convergence acceleration; Extrapolation; Iterative methods; Nonlinear equations; Computational fluid mechanics	18. Distribution Statement Unclassified – Unlimited Subject Category 64

19. Security Classif. (of this report)	20. Security Classif. (of this page)	21. No of pages	22. Price*
Unclassified	Unclassified	24	A02

NASA FORM 1626 OCT 86

*For sale by the National Technical Information Service, Springfield, Virginia 22161