JOURNAL OF COMPUTATIONAL PHYSICS 67, 230–235 (1986)

Note

Accelerating the Convergence of an Iterative Method for Derivatives of Eigensystems

1. Introduction

Many problems in engineering and physical sciences require numerical computation of derivatives of eigenvalues and eigenvectors of parameter-dependent matrices [1, 2]. The iterative method proposed by Rudisill and Chu [2], and later refined by Andrew [1, 3], is still much slower than the direct methods of [2] and [4]. The main contribution of this paper is to show that a dramatic improvement in efficiency of the iterative method may be made by means of the ε-algorithm, making the method competitive with direct methods even for relatively full matrices, while retaining all advantages of the iterative method. Numerical tests carried out in Section 3 show the accuracy of this method. Some refinements suggested by Andrew [1, 3] are also tested numerically.

2. The Iterative Method

Let the $n \times n$ matrix A and its eigenvalues λ_i and corresponding eigenvectors x_i depend smoothly on m real parameters $P_1, \ldots, P_m$. We are concerned with computing local numerical values of $\lambda_{i,j}$ and $x_{i,j}$ ($j = 1, 2, \ldots, m$), where the subscript $_{,j}$ denotes partial differentiation with respect to P_j. Let the eigenvalues be labelled so that

$$|\lambda_1| \geqslant |\lambda_2| \geqslant \cdots \geqslant |\lambda_n| \tag{1}$$

and let the eigenvectors satisfy the normalizing condition

$$x_i^* x_i = 1 \qquad (i = 1, 2, \ldots, n), \tag{2}$$

where the asterisk denotes complex conjugate transpose.

The basic algorithm of [2] for computing $\lambda_{1,j}$ and $x_{1,j}$ involves the iterative scheme

$$\mu(k) = x_1^* A_{,j} x_1 + x_1^* (A - \lambda_1 I)\, u(k), \tag{3a}$$

$$u(k+1) = \{(A_{,j} - \mu(k) I) x_1 - A u(k)\}/\lambda_1. \tag{3b}$$

0021-9991/86 $3.00

Andrew [3] has shown that $\mu(k)$ and $u(k)$ will converge to $\lambda_{1,j}$ and $x_{1,j}$ as $k \to \infty$ when $|\lambda_1| \neq |\lambda_2|$, the rate of convergence of (3) being at least $O(|\lambda_2/\lambda_1|^k)$.

In many optimization problems (see [5] and [6]), λ_1 and λ_2 are complex conjugates. In this case (3) will not converge for most choices of $u(0)$. Andrew's analysis [1] for the asymptotic error expansion of (3) leads to the suggested initial vector

$$u(0) = [A_{,j}x_1 - (x_1^* A_{,j} x_1)x_1]/(\lambda_1 - \lambda_2) \tag{4}$$

which under the condition $\lambda_1 \neq \lambda_2$ and $|\lambda_2| > |\lambda_3|$ gives

$$\|u(k) - x_{1,j}\| = O(|\lambda_3/\lambda_1|^k), \qquad |\mu(k) - \lambda_{1,j}| = O(|\lambda_3/\lambda_1|^k) \text{ as } k \to \infty.$$

The numerical results in Section 4 support this result. However, with the use of the ε-algorithm, this proposed starting vector is not required.

3. The Epsilon Algorithm

The ε-algorithm (see [7, 8, 9]) involves a sequence of numbers $S_0, S_1, S_2, \ldots$ defined in the two-dimensional "ε-array"

$$\begin{array}{ccccc}
\varepsilon_{-1}^{(0)} & & & & \\
 & \varepsilon_0^{(0)} & & & \\
\varepsilon_{-1}^{(1)} & & \varepsilon_1^{(0)} & & \\
 & \varepsilon_0^{(1)} & & \varepsilon_2^{(0)} & \\
\varepsilon_{-1}^{(2)} & & \varepsilon_1^{(1)} & & \ddots \\
 & \varepsilon_0^{(2)} & & & \\
\varepsilon_{-1}^{(3)} & & \vdots & & \\
 & \vdots & & & \\
\vdots & & & &
\end{array} \tag{5}$$

They are related by means of

$$\begin{gathered}\varepsilon_{-1}^{(k)} = 0; \qquad \varepsilon_0^{(k)} = S_k, \\ \varepsilon_{r+1}^{(k)} = \varepsilon_{r-1}^{(k+1)} + \frac{1}{(\varepsilon_r^{(k+1)} - \varepsilon_r^{(k)})}.\end{gathered} \tag{6}$$

It is known [7, 10] that if S_k has the form

$$S_k = S + \sum_{i=1}^{p} a_i y_i^k \qquad \text{with} \quad |y_1| > |y_2| > \cdots > |y_p|, \tag{7}$$

then

$$\varepsilon_{2p}^{(0)} = S. \tag{8}$$

In this case, we require the vector extension of the ε-algorithm as given in [8]. From Eqs. (7) and (8) of [3], it follows that if $u(0)$ is chosen to satisfy

$$x_1^* u(0) = 0, \tag{9}$$

then

$$u(k) = x_{1,j} + \sum_{r=2}^{n} \gamma_r \left(\frac{\lambda r}{\lambda_1}\right)^k \quad \text{with} \quad \left|\frac{\lambda_2}{\lambda_1}\right| > \cdots > \left|\frac{\lambda r}{\lambda_1}\right| > \cdots > \left|\frac{\lambda_n}{\lambda_1}\right|, \tag{10}$$

when inequality in (1) is strict, where the γ_r are constant vectors. Consequently for each component of the vector $u(k)$ *the error has exactly the same dependence on k as in* (7). To generate (6) for the vector case, we use the Samelson inverse of a vector. Suppose the vector

$$\mathbf{u} = (u_1, u_2, \ldots, u_n),$$

then the Samelson inverse of $\mathbf{u}$ is defined to be

$$\mathbf{u}^{-1} = (\|\mathbf{u}\|_2^2)^{-1}(\bar{u}_1, \bar{u}_2, \ldots, \bar{u}_n), \tag{11}$$

where $\bar{u}_i$ is the complex conjugate of u_i. From (7) and (10) it follows that, with exact arithmetic, the ε-algorithm when applied to the iterative scheme of Rudisill and Chu will yield the *exact* solution in the $(2n-2)$th column of the ε-table in (5). The number of iterations required to generate the exact result is $(2N-2)$, where N is the dimension of the matrix A. Note that the odd-numbered columns of (5) are only to aid computations (sometimes called the work columns) and have no significance. That is, only the even numbered columns in (5) have significance for our result. In fact the rate of convergence in the $2r$th column is $O(|\lambda_{r+2}/\lambda_1|^k)$ for each $r = 0, 1, 2, \ldots, (n-2)$. The performance of the algorithm in the presence of round-off errors is investigated in the next section. Values of $x_{1,j}$ obtained by the ε-algorithm were used to compute $\lambda_{1,j}$ using (3a).

4. Numerical Results

All computations in this section are carried out on a VAX-11/780 computer using the software package "MATLAB" which uses double precision (effectively about 16 significant decimal digits) and prints results in a floating point format of 16 digits. We define "exact" throughout this section as 15 digit floating point accuracy of the numerical solution compared with the analytic solution.

(a) Example 1. Consider the Nth-order lower *Hessenberg matrix* $A = (a_{i,j})$ where, for some (real or complex) parameter, α,

$$a_{i,j} = \alpha^{i+1-j} \qquad \text{all} \quad i \geqslant j-1. \tag{12}$$

A closed form solution for the eigenvalues and corresponding eigenvectors of A may be found in [11]. In fact

$$\lambda_m = \begin{cases} 4\alpha \cos^2 \dfrac{\pi m}{N+2} & \text{for} \quad m \leqslant \left[\dfrac{N}{2}\right] \\ 0 & \text{for} \quad m > [N/2], \end{cases} \tag{13}$$

where $[N/2]$ is the largest integer not exceeding $N/2$. By appropriate choice of α and N, we can study both ill-conditioned and well-conditioned systems. By choosing α small or N large, the eigenvalues in (13) can be made very close together and thus the case where A is close to a matrix with repeated eigenvalues may be investigated. Ill-conditioning may also occur when α is large because some elements of A will be very large compared to the eigenvalues of A. Round-off errors may be larger in this situation.

Table I summarizes the numerical testing carried out for computing $\lambda_{1,j}$ and $x_{1,j}$ for various different values of α and N. Good accuracy was obtained with significantly less than $(2N-2)$ iterations. This is because from (13), $\lambda_{[N/2]+1} =$

TABLE I

		Number of iterations required to satisfy (14) with TOL $= 10^{-6}$		Number of iterations required to yield at least t-figure accuracy with the ε-algorithm	
N	α	$u(0)=0.0$	$u(0)=[A_{,j}x_1-(x_1^*A_{,j}x_1)x_1]/(\lambda_1-\lambda_2)$	$t=10$	$t=14$
	0.0001	16	16	5	5
4	1.0	15	4	5	5
	20.0	15	14	5	5
	0.05	47	47	13	13
8	1.0	43	17	13	13
	10.0	44	43	9	13
	0.01	87	87	21	27
12	1.0	82	32	21	25
	5.0	82	81	17	21
	0.1	136	136	31	45
16	1.0	129	51	27	43
	3.0	129	128	21	43
	0.05	194	194	45	—
20	1.0	185	73	39	—
	2.5	185	183	39	—
	0.25	288	289	57	—
25	1.0	265	105	49	—
	2.0	265	263	49	—

$\lambda_{[N/2]+2} = \cdots = \lambda_N = 0$ and hence $O[\lambda_{[N/2]+1}/\lambda_1] = 0$ for all N. This means that with exact arithmetic the $(N-2)$th column (if N is even) or the $(N-3)$th column (if N is odd) should yield the exact solution in the first row. In practice round-off errors limit accuracy especially when N is large or the problem is ill-conditioned. However, accuracy improved quite rapidly with a few extra iterations and for later columns improved rapidly down the column.

For the well-conditioned systems, the rate of convergence improved from $O(|\lambda_2/\lambda_1|^k)$ to $O(|\lambda_3/\lambda_1|^k)$ when (4) was used instead of $u(0) = 0.0$. This is as predicted by the theory of [1]. For the ill-conditioned case, this choice of $u(0)$ offered no special advantage, possibly because λ_2 and λ_3 are very close in our ill-conditioned test examples.

In Table I, "t-figure" accuracy means that the error $\delta x_{1,j}$ in the computed value of $x_{1,j}$ satisfied $\|\delta x_{1,j}\|_\infty < 10^{-t}\, \|x_{1,j}\|_\infty$ and the error $\delta\lambda_{1,j}$ in the computed value of $\lambda_{1,j}$ also satisfied $|\delta\lambda_{1,j}| < 10^{-t}\, |\lambda_{1,j}|$. For $N \geqslant 20$ the ε-algorithm had not achieved 14 significant figure accuracy when iteration was arbitrarily terminated at $k = 61$, though more than 10 figure accuracy was achieved. For calculations without the ε-algorithm, iteration was continued until, for a parameter TOL,

$$|\mu(k) - \mu(k-1)| < \mathrm{TOL} \qquad \text{and} \qquad \|u(k) - u(k-1))\|_2 < \mathrm{TOL}. \tag{14}$$

(b) Example 2. In the next numerical example, we consider a case where the dominant eigenvalues are a complex conjugate pair and we combine the method of origin shift with the ε-algorithm.

Consider the 8×8 matrix

$$B = \begin{bmatrix} -C_1 - \sigma I & -D_1 \\ I & -\sigma I \end{bmatrix},$$

where σ is a shift parameter (σ real), C_1 and D_1 are the 4×4 matrices given in [12] and I is the 4×4 identity matrix. From [12] it follows that B has eigenvalues

$$\begin{gathered} \lambda^{(1,2)} = -\sigma \pm i, \qquad \lambda^{(3,4)} = -\sigma \pm \beta i, \qquad \lambda^{(5,6)} = 1 - \sigma - \beta \pm \beta i, \\ \lambda^{(7)} = -\sigma, \qquad \lambda^{(8)} = 1 - \sigma - \beta. \end{gathered} \tag{15}$$

Different σ in (15) give different dominant eigenvalues. With $\sigma = -1$ for example, the largest eigenvalues of (15) are $\lambda^{(3,4)}$ followed by $\lambda^{(5,6)}$ for all values of $\beta \in (1, 3)$. An eigenvector corresponding to the eigenvalues $\lambda^{(3,4)}$ is

$$\mathbf{x} = [-\beta^4, i\beta^3, \beta^2, i\beta, i\beta^3, -\beta^2, i\beta, 1]^{\mathrm{T}}.$$

Different values of $\beta \in (1, 3)$ were chosen for numerical testing with this shift parameter. The method of origin shift as suggested by Andrew [3] was found to be a useful device in accelerating the rate of convergence of the iterative scheme when the ε-algorithm is not used, the increase in convergence rate for well-conditioned problems being in good agreement with the predictions of [3]. For β near 1.0, the problem is very ill-conditioned because $\lambda^{(1)} \simeq \lambda^{(3)} \simeq \lambda^{(5)}$ and $\lambda^{(2)} \simeq \lambda^{(4)} \simeq \lambda^{(6)}$ and the optimal origin shift [3] offers no special advantage. Convergence of the even column of the ε-algorithm was also found to be much faster when the optimal origin shift was used. However, for well-conditioned problems, both the optimal origin shift and the shift $\sigma = -1$ gave the "exact" solution after $14 = 2N - 2$ iterations, as predicted by the theory.

Recently the power of the ε-algorithm has been significantly extended by various generalizations such as [13]. Also several alternative schemes (see [14, 15, 16]) have been proposed which assume a different asymptotic form for the sequence. The reason for the special appropriateness of the ε-algorithm for our problem is that the error term of (10) has *exactly* the same form as in (7). This is not true for the schemes proposed in [14, 15, 16] and our numerical experience confirms that they perform much less well in this particular case than the ε-algorithm. (See [17] for details.)

Acknowledgments

The author would like to thank Dr. A. L. Andrew for the many helpful discussions he had with him during the preparation of this paper.

References

1. A. L. Andrew, *J. Inst. Math. Appl.* **24**, 209 (1979).
2. C. S. Rudisill and Y. Y. Chu, *AIAA J.* **13**, 843 (1975).
3. A. L. Andrew, *J. Comput. Phys.* **26**, 107 (1978).
4. R. B. Nelson, *AIAA. J.* **14**, 1201 (1976).
5. J. G. Beliveau, *J. Optim. Theor. Appl.* **23**, 41 (1977).
6. S. Garg, *AIAA. J.* **11**, 1191 (1973).
7. P. Wynn, *Math. Comput.* **10**, 91 (1956).
8. P. Wynn, *Math. Comput.* **16**, 301 (1962).
9. P. Wynn, *SIAM J. Numer. Anal.* **3**, 91 (1966).
10. D. Shanks, *J. Math. Phys.* **34**, 1 (1955).
11. G. Fairweather, *SIAM Rev.* **13**, 220 (1971).
12. L. Hayes and E. Wasserstrom, *J. Inst. Math. Appl.* **17**, 5 (1976).
13. C. Brezinski, *Numer. Math.* **46**, 311 (1985).
14. D. Levin, *Int. J. Comput. Math. B* **3**, 371 (1973).
15. D. Levin and A. Sidi, *Appl. Math. Comput.* **9**, 175 (1981).
16. A. Sidi, *Math. Comput.* **33**, 315 (1979).
17. Roger C. E. Tan, La Trobe University, Department of Mathematics, Mathematics Research Paper, 86-8 (1986) (unpublished).

Received: August 8, 1985; revised: December 6, 1985

Roger C. E. Tan
Department of Mathematics,
La Trobe University,
Bundoora, Victoria 3083, Australia