JOURNAL OF COMPUTATIONAL PHYSICS 118, 1–8 (1995)

A Chebyshev Polynomial Interval-Searching Method ("Lanczos Economization") for Solving a Nonlinear Equation with Application to the Nonlinear Eigenvalue Problem

JOHN P. BOYD

Department of Atmospheric, Oceanic & Space Science, University of Michigan, 2455 Hayward Avenue, Ann Arbor Michigan 48109

Received October 19, 1993; revised May 4, 1994

To search a given *real* interval for roots, our algorithm is to replace $f(\lambda)$ by $f_N(\lambda)$, its N-term Chebyshev expansion on the search interval $\lambda \in [\lambda_{min}, \lambda_{max}]$, and compute the roots of this proxy. This strategy is efficient if and only if $f(\lambda)$ itself is expensive to evaluation, such as when $f(\lambda)$ is the determinant of a large matrix whose elements depend nonlinearly on λ. For such expensive functions, it is much cheaper to search for zeros of $f_N(\lambda)$, which can be evaluated in $O(N)$ operations, than to iterate or look for sign changes on a fine grid with $f(\lambda)$ itself. It is possible to systematically increase N until the Chebyshev series converges acceptably fast, without wasting previously computed values of $f(\lambda)$, by imitating the Clenshaw–Curtis quadrature. Our strategy of replacing $f(\lambda)$ by $f_N(\lambda)$ is similar to Lanczos economization of power series, which also replaces an expensive function by a Chebyshev approximation that is more rapidly evaluated. The errors induced by the Chebyshev approximation can be eliminated by a final step of one or two iterations with $f(\lambda)$ itself, using the zeros of $f_N(\lambda)$ as initial guesses. We show through numerical examples that the algorithm works well. The only sour note is that it is sometimes necessary to split the search interval into subintervals, each with a separate Chebyshev expansion, when $f(\lambda)$ varies by many orders of magnitude on the search interval.

1. INTRODUCTION

Most methods for solving $f(\lambda) = 0$ replace the function by a proxy, and then solve that. For Newton's method, the proxy is the two-term Taylor series expansion about the current iterate. For the secant method, the proxy is the linear polynomial which interpolates the last two iterates. In this note, we propose to replace $f(\lambda)$ by a proxy function which is its truncated Chebyshev expansion.

Such a strategy makes sense only if the proxy is cheaper and less expensive to solve than $f(\lambda)$ itself. This in turn implies that $f(\lambda)$ must be *expensive*, that is, require many more operations to evaluate than $f_N(\lambda)$, its N-term Chebyshev approximant. To illustrate our strategy, we shall focus on nonlinear eigenvalue problems, which shall here denote an $f(\lambda)$ which is the determinant of a matrix whose elements depend *nonlinearly* on the eigenparameter λ. Since the cost of evaluating the determinant of an M-dimensional matrix is $O(2\ M^3/3)$, it is obvious that if M is large, evaluating $f(\lambda)$ will be enormously costly in comparison to summing the Chebyshev series to compute $f_N(\lambda)$, which can be done by a three-term recurrence in $O(N)$ operations.

The coefficients of the N-term Chebyshev series can be computed by interpolation, which requires N evaluations of $f(\lambda)$ plus either a matrix multiplication or a fast Fourier transform. It is hard to specify an appropriate N *a priori*, but the algorithm does not require this. By using the Gauss–Lobatto grid for interpolation, the number of grid points can be doubled as often as necessary while reusing all previously computed values of $f(\lambda)$. The convergence of the Chebyshev coefficients can be monitored to determine when N is sufficiently large.

The remaining difficulty is that the user must specify a search interval, that is, an interval in λ on which the Chebyshev interpolant will be fitted. However, most root-finding methods, including Newton's iteration and the secant method, require a first guess for the root. It is easier to estimate an interval than a point!

The complete algorithm is summarized in Table I. In the next few sections, we shall explain the details.

2. COMPUTING THE CHEBYSHEV APPROXIMANT

To expand a function $f(\lambda)$ as an N-term series on a general interval $\lambda \in [a, b]$, define [1, 2]

$$c \equiv (a + b)/2; \quad d \equiv (b - a)/2 \tag{2.1}$$

$$\lambda_i \equiv c - d\cos[\pi(i - 1)/(N - 1)], \quad i = 1, 2, ..., N. \tag{2.2}$$

Let **F** denote the column vector with elements

$$F_i \equiv f(\lambda_i), \quad i = 1, 2, ..., N. \tag{2.3}$$

Let **a** denote the column vector containing the Chebyshev coefficients of f_N and let **H** be the square matrix with elements

0021-9991/95 $6.00

TABLE I

Summary of the Algorithm

1. Choose the following:
   (i) Search interval, $\lambda \in [\lambda_{\min}, \lambda_{\min}]$.[a,b]
   (ii) N, the number of grid points.[c]
2. Compute the roots of f_N, the Chebyshev interpolant of $f(\lambda)$.
3. Refine the roots by a secant or Newton iteration with $f(\lambda)$ itself.

[a] The search interval must be chosen by physical and mathematical analysis of the individual problem.

[b] If $f(\lambda)$ is badly scaled, that is, varies by many order of magnitude over the search interval, then split the interval into subintervals so that the range of f on each subinterval is well within machine precision.

[c] N may be chosen by setting $N = 1 + 2^m$ and the increasing N until the Chebyshev series displays satisfactory convergence.

$$H_{ij} \equiv (2/[N-1])\,\beta_i\,\delta_j \cos([i-1]\,\pi\,[j-1]/[N-1]), \quad i = 1, 2, \ldots N; j = 1, 2, \ldots, N, \tag{2.4}$$

where

$$\beta_i = \begin{cases} 1/2, & i = 1, N \\ 1, & i \neq 1, N \end{cases}; \quad \delta_j = \begin{cases} 1/2, & j = 1, N \\ 1, & j \neq 1, N \end{cases} \tag{2.5}$$

Then the Chebyshev coefficients of f_N, the Nth degree polynomial which interpolates $f(\lambda)$ at the λ_i, are given by

$$\mathbf{a} = \mathbf{H}\,\mathbf{F}, \tag{2.6}$$

i.e., by the multiplication of a vector by a matrix, which requires $O(2\,N^2)$ operations. (Alternatively, this matrix-vector multiple can be replaced by a *fast cosine transform* [1].)

The series

$$f_N(\lambda) = \sum_{j=0}^{N-1} a_{j+1}\,T_j(2[\lambda - (a+b)/2]/[b-a]) \tag{2.7}$$

may be summed in $O(N)$ operations by the initialization

$$y = 2\,(\lambda - (a+b)/2)/(b-a), \quad b_1 = 0, \quad b_2 = 0, \tag{2.8}$$

followed by $(N-1)$ passes through the loop

$$b_0 = 2yb_1 - b_2 + a_{N+1-j}, \quad j = 1, 2, \ldots, N-1, \tag{2.9}$$
$$b_2 = b_1, \quad b_1 = b_0.$$

Then

$$f_N = yb_1 - b_2 + a_1. \tag{2.10}$$

The choice of the search interval $[a, b]$ depends on the user's knowledge of the physics of his/her problem, and no general rules are possible. However, the selection of N can be automated.

To determine when N is sufficiently high, we can examine the Chebyshev coefficients a_j, which decrease exponentially fast with j. Many choices of convergence criterion are possible; we suggest accepting N when

$$\sum_{j=[2N/3]}^{N-1} |a_j| < \varepsilon, \tag{2.11}$$

where ε is a user-chosen tolerance and [2 N/3] denotes the integer nearest 2 N/3.

If N is increased from $N = \nu + 1$ to $2\,\nu + 1$, all the interpolation points λ_i from the previous coarse grid are also points of the refined grid. Thus, nothing is wasted by initially choosing N too small; all the computed values of $f(\lambda)$ can be reused for the higher degree interpolation, provided N is of the form $2^m + 1$.

A similar "double-N-and-recycle" strategy is the basis for the adaptive, Chebyshev polynomial-based, integration scheme known as the "Clenshaw–Curtis quadrature" [3, 4].

3. FINDING THE ROOTS OF THE CHEBYSHEV APPROXIMANT

Because the evaluation of $f_N(\lambda)$ is so cheap—even for $N = 100$, a Unix workstation should be capable of $O(10^4$–$10^5)$ evaluations per second—one can use almost any algorithm to compute the roots of $f_N(\lambda)$.

Another option is to convert the Chebyshev series to an ordinary polynomial, that is, a sum of terms of the form $b_j\,\lambda^j$. One can then call a polynomial root-finder. The danger is that this conversion is highly ill-conditioned; the coefficients b_j of the powers of λ may grow rapidly with j, even though the Chebyshev coefficients are decreasing exponentially with j. The advantage of the Chebyshev-to-powers-of-λ conversion is the ready availability of robust library software to solve polynomial equations. Since the decimal places which are lost to the ill-conditioned conversion can usually be retrieved by one or two Newton's iterations with the Chebyshev form, conversion-to-powers is a fairly attractive option.

If we rescale λ so that $\lambda \in [a, b]$ is mapped into $\boldsymbol{\lambda} \in [-1, 1]$ and then let $\mathbf{b}$ be the column vector storing the coefficients of the polynomial form

$$f_N(\boldsymbol{\lambda}) = \sum_{j=0}^{N-1} b_{j+1}\,\boldsymbol{\lambda}^j \tag{3.1}$$

and define $\mathbf{Q}$ as the square matrix whose elements are

$$Q_{11} = 1; \quad Q_{i1} = 0, \quad i \neq 1;$$

$$Q_{ij} = \begin{cases} \dfrac{(j-1)\left(\dfrac{i+j}{2}-2\right)!2^i(-1)^{(j-i)/2}}{4\left(\dfrac{j-i}{2}\right)!(i-1)!}, \\ 0. \qquad i, j \text{ both odd or both even and } i \leq j \end{cases}$$

Then

$$\mathbf{b} = \mathbf{Qa}. \tag{3.3}$$

It is usually easy to compute the roots of the truncated Chebyshev expansion directly, however. For the examples below, we sampled $f_N(\lambda)$ on a fine grid to isolate the roots to within small subintervals and then we applied the secant iteration. However, because $f_N(\lambda)$ is inexpensive to evaluate, we have many good choices for computing its roots.

4. NUMERICAL EXAMPLES

Both examples are nonlinear eigenvalue problems; the goal is to find zeros of the determinant of an $M \times M$ matrix $\mathbf{A}$. The first problem is from Ruhe [5], where the 8×8 matrix is defined by

$$\mathbf{A} = (e^{\lambda} - 1)\,\mathbf{B}_1 + \lambda^2\,\mathbf{B}_2 - \mathbf{B}_0, \tag{4.1}$$

where $\mathbf{B}_0$ is identity matrix multiplied by 100 and

$$(\mathbf{B}_1)_{ij} \equiv (9 - \max(i,j))\,ij \tag{4.2}$$

$$(\mathbf{B}_2)_{ij} \equiv 8\delta_{ij} + 1/(i+j). \tag{4.3}$$

The first step, for any nonlinear root-finding problem, is to investigate the individual example as thoroughly as possible through analytical and qualitative methods. As Acton [6] puts it, "If there exists any one reliable algorithm for finding the roots of transcendental equations it is yet to be found. We have a variety of medicines that work with varying degrees of potency ..., but the state of the art still precludes the confident writing of computational prescriptions without having looked over the patient rather closely."

For the problem (4.1)–(4.3), Ruhe [5] supplies some helpful theory. For M-dimensional matrices of the form of $\mathbf{A}(\lambda)$, the following has been proven:

(i) There are exactly two M eigenvalues.

(ii) All eigenvalues are real.

(iii) n eigenvalues are positive while n are negative.

(iv) all are finite.

Since $M = 8$ for (4.1), it follows that our search is finished when we have found 16 roots.

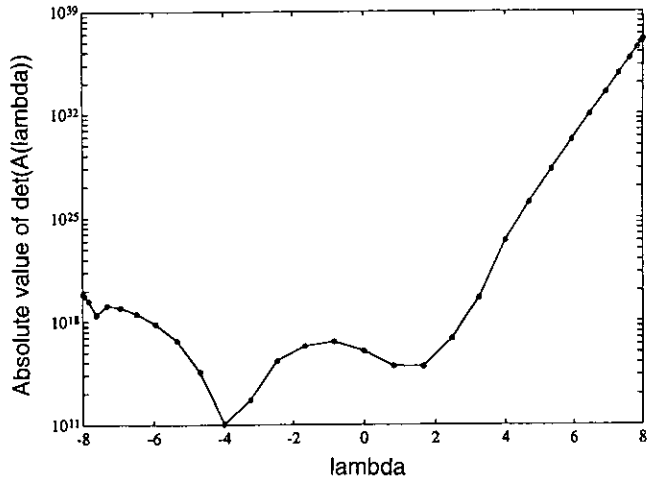

FIG. 1. $|\det(\mathbf{A}(\lambda))|$ at 31 Chebyshev collocation points on $\lambda \in [-8, 8]$. Data values are shown as circles, which are connected by straight lines.

To estimate $\lambda_{\min}$ and $\lambda_{\max}$, note that all of the component matrices $\mathbf{B}_j$ are individually nonsingular. It follows that when λ is so large or small that one of these three matrices dominates the others, $\mathbf{A}(\lambda)$ must be nonsingular, where "dominates" means "has larger norm."

For large negative λ, $\exp(\lambda) \ll 1$, so

$$\mathbf{A}(\lambda) \approx \lambda^2\,(\mathbf{B}_2 - \lambda^{-2}\,(\mathbf{B}_0 + \mathbf{B}_1)), \quad \lambda \ll -1. \tag{4.4}$$

By comparing norms, we find that B_2 has larger norm than $(\mathbf{B}_0 + \mathbf{B}_1)/\lambda^2$ for all $\lambda < -8$. Similarly, for large positive λ, $\exp(\lambda)\ \mathbf{B}_1$ dominates the other terms for $\lambda > 8$ (roughly).

The physics of the problem then determines the parameters of the search: To test for roots on $\lambda \in [-8, 8]$ until we have identified 16 eigenvalues. Since the usual rule-of-thumb [1] is that roughly π polynomials per wavelength are the absolute minimum to resolve a quasi-sinusoidal disturbance—$\pi/2$ points/root—it follows that, to fit the determinant over the whole search interval (with its 16 roots), the minimum $N = 31$, or roughly two collocation points per root.

Figure 1 illustrates $|\det(\mathbf{A}(\lambda))|$ at the collocation points. We note that (i) the determinant is badly scaled in the sense that it varies in magnitude over the interval by more than the floating point range on our machine (roughly 10^{16}) and (ii) the linear growth for $\lambda > 4$, which is roughly $\exp(8\,\lambda)$, suggests that the $\exp(\lambda)\ \mathbf{B}_1$ is dominating the rest of the matrix for smaller λ than we estimated *a priori*. (Note that from the definition of the determinant, one can show that growth of matrix elements in each row or column as $\exp(\lambda)$ implies that the determinant will grow as $\exp(M\,\lambda)$ for sufficiently large λ, where M is the matrix dimension.)

The first observation implies that the search interval must be split into subintervals so that the range of $f(\lambda)$ (=$\det(A(\lambda))$ here) within each subinterval is smaller than the reciprocal of

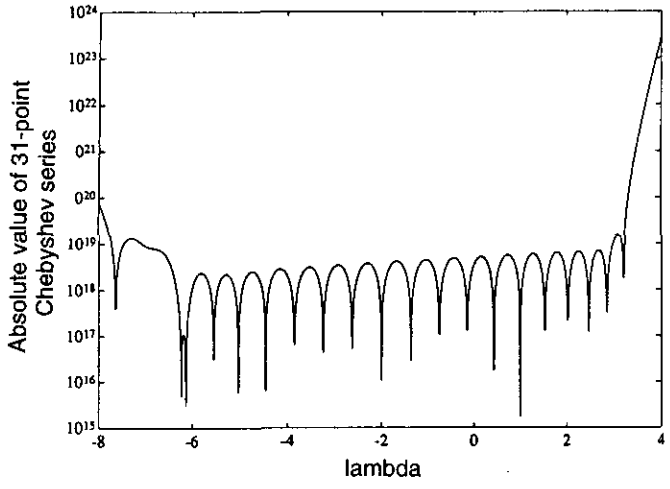

FIG. 2. Absolute value of the 31-point Chebyshev approximation to $\det(\mathbf{A}(\lambda))$ on $\lambda \in [-8, 4]$.

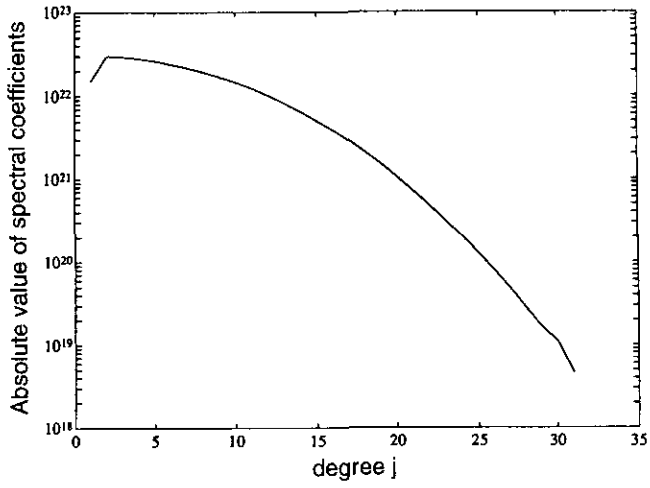

FIG. 3. $|a_j|$ versus j for the 31-point Chebyshev series for $\det(\mathbf{A}(\lambda))$ on $\lambda \in [-8, 4]$. The ratio of the largest to the smallest coefficient defines $\delta(N)$.

machine epsilon. (The smallest difference between two floating point numbers on a given computer is "machine epsilon," 2×10^{-16} here.) On a subinterval, where

$$f(\lambda) < \varepsilon f_{\max}, \tag{4.5}$$

where ε is the machine epsilon and $f_{\max}$ is the maximum of $|f(\lambda)|$ on the expansion interval, the Chebyshev series—even in the limit $N \Rightarrow \infty$—approximates $f(\lambda)$ by random noise of amplitude $O(\varepsilon f_{\max})$. If there are roots of $f(\lambda)$ within the noisy subinterval, this is disastrous.

Second, the region of exponential growth suggests that it is unlikely that there are roots on $\lambda \in [4, 8]$. So, we will return to this subinterval if and only if our search elsewhere fails to locate all 16 roots.

Figure 2 shows that the $N = 31$ approximation on $[-8, 4]$ has 20 roots, each marked by a downward spike on this logarithmic graph of the absolute value. Since there are really only 16 roots, it is obvious that N is too small. However, we could deduce this even without prior knowledge of the true number of eigenvalues. Inspection of the Chebyshev coefficients shows that a_{N-1} is $O(\delta)\,|a_0|$, where δ is $O(10^{-4})$ (Fig. 3). On subintervals, where

$$f(\lambda) < \delta(N) f_{\max}, \tag{4.6}$$

the Chebyshev series will also approximate $f(\lambda)$ by noise. In other words, an approximation accurate for root-finding over the whole of an interval $[\lambda_0, \lambda_1]$ requires that

$$|f(\lambda)|/f_{\max} > \delta(N), \varepsilon \tag{4.7}$$

on the whole of $\lambda \in [\lambda_0, \lambda_1]$, except for very small intervals about the roots themselves. The large subinterval of ripples whose amplitude is roughly $\delta(31)\, f_{\max}$ is highly suspicious.

When N is doubled, the Chebyshev approximation reproduces all 16 roots faithfully (Fig. 4 and Table II). The graph vividly illustrates the clustering of six roots between $\lambda = -4$ and $\lambda = -3.5$, but the Chebyshev series is nevertheless able to resolve them.

Figure 5 is a zoom diagram of $f_{61}(\lambda)$. Remarkably, there are only two Chebyshev interpolation points on the interval, marked by the vertical dotted lines, but the six zeros of $\det(\mathbf{A}(\lambda))$ are nonetheless faithfully mirrored by the Chebyshev approximation. Piecewise linear interpolation could only detect a single zero between neighboring grid points. The Chebyshev method is global, however, in the sense that the approximant on the small interval shown is computed by using all 61 samples of

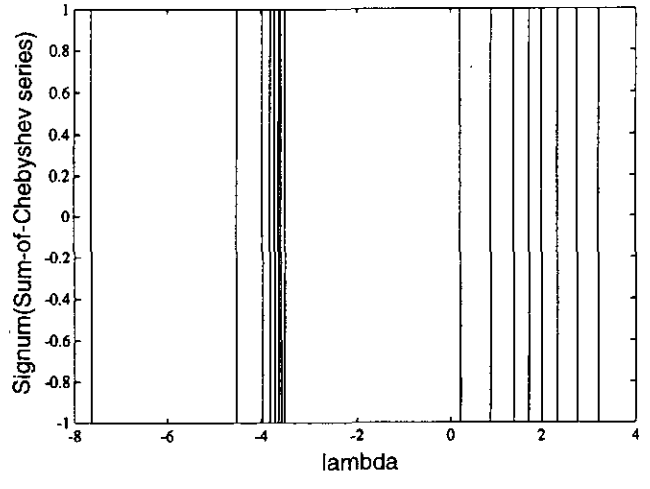

FIG. 4. $\mathrm{sign}(f_{61}(\lambda))$, sampled at intervals of 0.01. The near-vertical lines mark the roots of the Chebyshev approximation.

TABLE II

Roots of Problem 1, Exact and Numerical

m	Exact λ_m / Numerical	|Error|
1	−7.642558348	1.2E − 13
	−7.642558348	
2	−4.521556148	−1.9E − 8
	−4.511556129	
3	−3.968169057	2.4E − 5
	−3.968194	
4	−3.801274897	3.5E − 4
	−3.801274898	
5	−3.702761577	4.1E − 4
	−3.70175 to	
	−3.70295	
6	−3.627468151	2.8E − 4
	−2.62665 to	
	−3.62885	
7	−3.571755851	3.6E − 4
	−3.57085 to	
	−3.57195	
8	−3.491852633	6.1E − 5
	−3.491791	
9	0.217461384	6.8E − 10
	0.217461385	
10	0.884961520	2.6E − 9
	0.884961523	
11	1.394724184	4.8E − 10
	1.394724184	
12	1.726304141	1.5E − 9
	1.726304140	
13	2.007943631	1.2E − 10
	2.007943630	
14	2.335424784	9.3E − 11
	2.335424784	
15	2.731077006	8.4E − 12
	2.731077006	
16	3.182595890	6.7E − 14
	3.182595890	

$f(\lambda)$. Despite the rule of thumb quoted earlier—one root per two grid points—the Chebyshev method can sometimes do much better. The moral of Fig. 5 is that it is necessary to be very careful in computing the roots of the Chebyshev approximation, lest one miss roots of $f(\lambda)$ that have been correctly incorporated in its proxy, $f_N(\lambda)$.

Increasing N produces no improvement because $\delta(61) \approx \varepsilon$; that is, the last few coefficients of the computed Chebyshev series are at the roundoff threshold and increasing N would only generate noise, as illustrated in Fig. 6. Indeed, $f_{61}(\lambda)$ is so noisy in the vicinity of three of the roots ($m = 5, 6, 7$) that the secant iteration did not converge, but instead oscillated within the small interval shown in Table II. Nevertheless, if the centroid of these intervals is taken as the best estimate of the root, all of the roots of the Chebyshev series have absolute errors of no more than 0.00041.

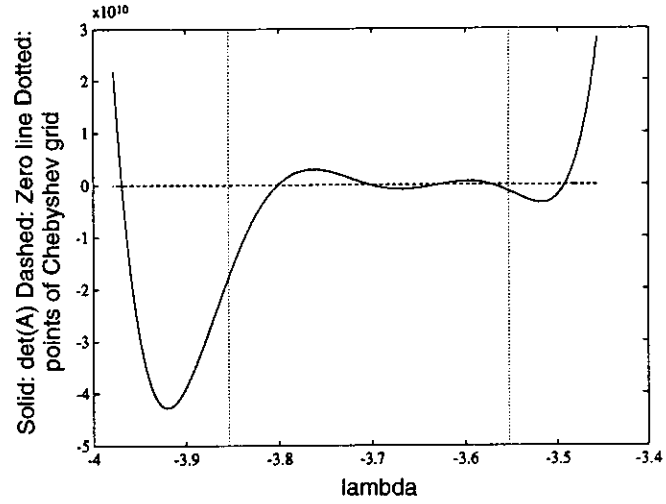

FIG. 5. Solid: $f_{61}(\lambda)$. Dashed: $f = 0$; the intersections of this with the solid curve mark the six roots of $f_{61}(\lambda)$ on this interval. Dotted vertical lines: The two points of the 61-point Chebyshev interpolation grid which lie on the interval illustrated.

To compute the roots of the $f_{61}(\lambda)$, we sampled it on a fine grid and then took $\lambda_{root} \sim (\lambda_j + \lambda_{j+1})/2$, where λ_j and λ_{j+1} are any two successive points on the fine grid such that sign $(f_{61}(\lambda_j)) = -\text{sign}(f_{61}(\lambda_{j+1}))$. We then applied the secant iteration to compute the roots of $f_N(\lambda)$ to machine precision (except for the three where we obtained convergence only to a small interval).

We can purge the errors in the Chebyshev approximation by performing the final step of the algorithm, refining the roots of $f_N(\lambda)$ by secant iterations using $f(\lambda)$ itself.

For this problem, we experimented with a variety of scaling

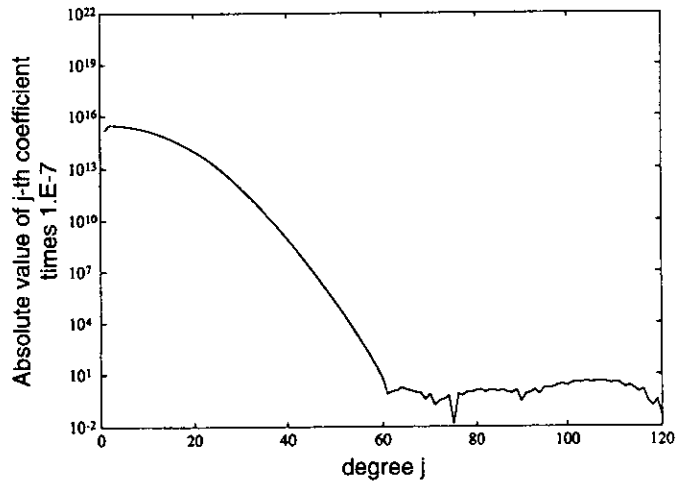

FIG. 6. $|a_j|$, scaled by multiplication by 1E − 7, versus j for the Chebyshev series for det($\mathbf{A}(\lambda)$) on $\lambda \in [-8, 4]$. The spectral coefficients plateau for $j >$ 60 at roughly $|a_j| \sim O(\varepsilon)$ times the largest coefficient, where ε is the machine epsilon (=2.E − 16).

functions, that is, computing the eigenvalues by finding the roots of

$$f(\lambda) = \det(\mathbf{A}(\lambda))/S(\lambda), \tag{4.8}$$

where $S(\lambda)$ is a smooth function chosen to reduce the huge variations in the magnitude of $f(\lambda)$ that are shown in Fig. 1. A product of the empirical functions failed. Choosing $f(\lambda)$ as a sum of Gaussians, each centered on a Chebyshev grid point with amplitude equal to $|f(\lambda_i)|$, failed, too. Using the smallest singular value of $\mathbf{A}(\lambda)$, $\sigma_{\min}(\lambda)$, or $f(\lambda) = \text{sign}(\det(\mathbf{A}(\lambda))\,\sigma_{\min}(\lambda)$, failed also. In each case, the scaling function reduced the range of $|f(\lambda)|$, but only at the price of clobbering the convergence of the Chebyshev series. Splitting an interval into subintervals is the safest practical strategy we have found so far. Perhaps the reader can suggest something better!

We also solved this example by three separate expansions, each with $N = 24$, for the intervals in λ of $[-8, -1]$, $[1, 1]$, and $[1, 8]$, and then computed the Chebyshev roots by converting to ordinary polynomials and applying a library implementation of the Traub–Jenkins algorithm. Despite the additional errors caused by the Chebyshev-to-powers-of-λ conversion, all 16 roots were computed to at least five decimal place accuracy.

Ruhe [5] has developed an alternative algorithm which is guaranteed to converge for so-called "symmetric overdamped" problems, which includes this example. His algorithm requires the solution of four linear eigenproblems per root, which at a cost of $O(15\ M^3)$ operations per eigenproblem gives a total for this example which is roughly equal to 1200 evaluations of $\det(\mathbf{A}(\lambda))$—an order of magnitude more expensive than our method.

Many other alternatives are available such as the methods of Hayes and Wasserstrom [8], Kublanovskaya [9], and Anselone and Rall [10]. We have not explicitly tested these other schemes. However, the continuation method of Hayes and Wasserstrom [8] often fails if the continuation path is real because collisions of real roots are common, so one must usually employ a complex-valued path and suffer the expense of complex arithmetic. The local iteration of Kublanovskaya often, when combined with orthogonalization strategy of Anselone and Rall, gave all the eigenvalues—but not always. Our Chebyshev strategy is a systematic search for all roots on an interval, so it seems fair to say that it is very competitive with other algorithms for solving nonlinear eigenvalue problems.

Our second example is an M-dimensional matrix whose elements are

$$\begin{aligned} A_{ij} = &\{-(j-1)^2 - 1 - 1/c + c^2\} \\ &\cos(\pi[i-1][j-1]/(M-1)), \quad i,j = 1, \ldots, M, \end{aligned} \tag{4.9}$$

which is the Fourier cosine pseudospectral discretization of the differential equation

$$u_{yy} + \{-1/c - 1 + c^2\}u = 0, \tag{4.10}$$

where c is the eigenvalue. Differential equations of this form are common in meterology and oceanography, where c is the phase speed (or frequency) of a linearized wave. A more realistic model would include y-varying coefficients and non-periodic boundary conditions, but (4.9) and (4.10) is the simplest equation which is representative of this larger class [7]. (In the calculations below, we took $M = 40$.)

After multiplying through by c, Eq. (4.9) becomes a matrix whose elements are *polynomials* in the eigenparameter c. Peters and Wilkinson [8] have shown that problems with polynomial nonlinearity of degree r and dimension M can always be converted into a *linear* eigenproblem of dimension $M' = Mr$. Indeed, (4.10) is actually the result of reducing three differential equations in three unknowns, each linear in c, to a single equation. By solving the original trio of equations ("linearized shallow water wave equations"), we can apply a standard library routine for the QR or QZ algorithm and avoid all need for specialized nonlinear eigenvalue software.

The only rub is that the QZ algorithm has a cost which scales as the cube of the matrix, being roughly $O(15\ [M']^3)$. This implies that the QZ algorithm is as expensive as 600 evaluations of $\det(\mathbf{A}(c))$. The situation for higher polynomial nonlinearity of matrix elements is even worse. Thus, there is ample reason to look for less expensive alternatives to linear eigensolvers.

The first step, as always, is "to look over the patient rather carefully," in Acton's words. In this case, geophysicists have known for several decades that the solutions of (4.10) fall into three classes: (i) Rossby waves with $c \in [-1, 0]$; (ii) westward-travelling gravity waves with $c \in [-\infty, -1]$; and (iii) eastward-travelling gravity waves with $c \in [1, \infty]$. There is a single root for $m = 0$ ($c = 1.32$) and the roots for larger integer m are approximately given by $c_{\text{Rossby}} \approx -1/(1 + m^2)$ and $c_{\text{gravity}} \approx \pm(1 + m^2)^{1/2}$.

This suggests that it will be convenient to search the modes of each class separately; it is impractical to search a finite interval in c that spans the Rossby region, because the differential equation has an infinite number of modes within $[-1, 0]$. The M-dimensional discretization matrix, of course, has only M Rossby modes, but their narrow spacing is likely to cause problems. So, we shall not only make three separate searches, one for each class of modes, but to compute the Rossby modes, we shall set the eigenparameter $\lambda = 1/c$. We can then search for the lowest few Rossby modes—often the only ones of geophysical interest—by searching a portion of the negative λ-axis.

To compute the lowest 14 Rossby eigenvalues, for example, we chose $N = 31$ (following our rule of roughly 2 points/zero at least) and expanded $f(\lambda)$ [$= \det(\mathbf{A}(c))$], where $\mathbf{A}(c)$ is a 40×40 matrix, on the interval $\lambda \in [-200, -1]$. The values of $f(\lambda)$ at the interpolation points is shown in Fig. 7. Sampling the Chebyshev series on a fine grid produces Fig. 8, whose downward spikes clearly mark the 15 roots of the approximant. Table III shows that all roots of $f_{31}(\lambda)$ on the search interval are good approximations to those of the differential equations.

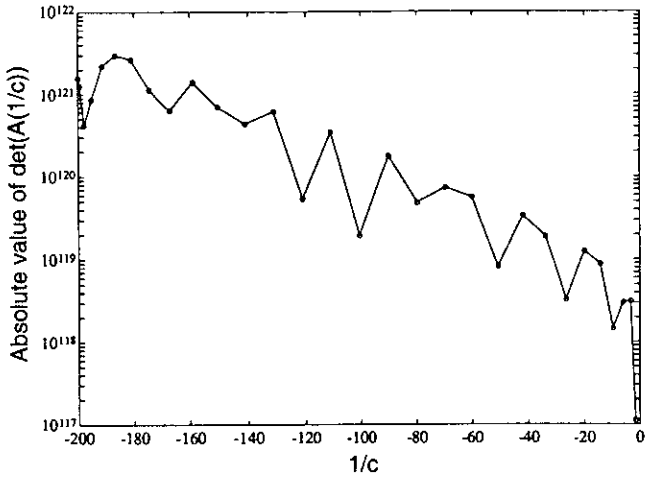

FIG. 7. (Example 2: equatorial waves). Circles: $|f(\lambda)|$ at the points of the 31-point Chebyshev interpolant on $\lambda \in [-200, -1]$ for Example 2. The data points have been connected by line segments (solid) for visual clarity.

The largest relative error (as opposed to the absolute errors in the table) is less than 0.8%.

The gravity modes can be computed in the same way. When eigenvectors are desired, they can be easily computed by inverse iteration, just as in a linear eigenproblem, once λ is known to high accuracy [5].

The cost of the 31 evaluations of the determinant of the 40×40 matrix to generate Table III is only $\frac{1}{20}$ that of the Peters–Wilkinson strategy of converting the problem to a linear eigenproblem and then applying the QZ algorithm. Alternatively, one could find the roots by simply evaluating the determinant on a fine grid, without Chebyshev interpolation, to generate a plot similar to Fig. 8. The number of grid points needed for such a non-interpolating search depends on the desired accuracy, but would be much greater than 31. Counting the sign changes between the points plotted in Fig. 7 misses one root completely and gives estimates in error by as much as 50% for some others. Chebyshev interpolation, in contrast, is a way of maximizing the amount of information that can be extracted from N samples of $f(\lambda)$.

TABLE III

Roots of Problem 2, Roots of Exact and 31-point Chebyshev Approximation

m	$f(1/c)$ $f_{31}(1/c)$	|Absolute error|
1*G*	−1.0000	5.5E − 13
	−1.0000	
1	−1.6180	9.1E − 3
	−1.6089	
2	−4.9593	0.036
	−4.9233	
3	−9.9900	8.0E − 3
	−9.9980	
4	−16.9965	0.018
	−17.0143	
5	−25.9985	2.8E − 3
	−26.0013	
6	−36.9993	6.5E − 3
	−49.9985	
7	−49.9996	1.1E − 3
	−49.9985	
8	−64.9998	2.3E − 3
	−65.0021	
9	−81.9999	7.7E − 4
	−81.9991	
10	−100.9999	9.4E − 5
	−101.0000	
11	−121.9999	7.3E − 5
	−121.9999	
12	−145.0000	1.2E − 4
	−145.0001	
13	−170.0000	4.1E − 5
	−170.0000	
14	−197.0000	5.6E − 6
	−197.0000	

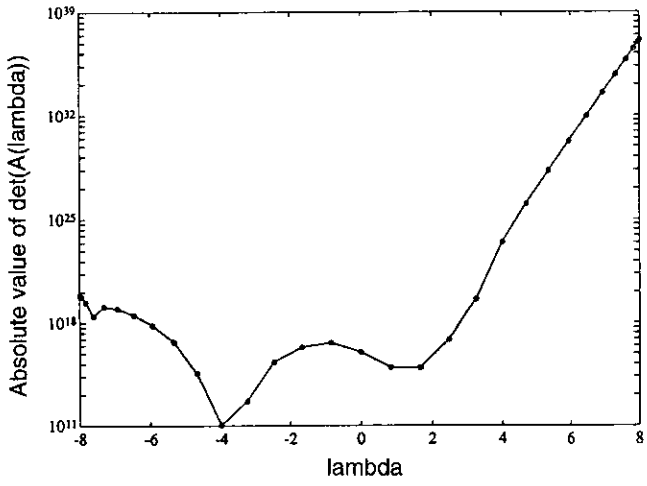

FIG. 8. $|f_{31}(\lambda)|$ on $\lambda \in [-200, -1]$, where λ is the reciprocal of the phase speed c. In contrast to Fig. 7, the (inexpensive) Chebyshev interpolant, $f_{31}(\lambda)$, was evaluated on a fine grid with a spacing of 0.1 to produce the curve shown.

Increasing N to 61 gave only modest improvement because the determinant has an 80th-order pole at $\lambda = 0$. We rescaled the determinant by (i) multiplying through by λ^2 and (ii) multiplication by $\lambda^2/(1 + \lambda^2/100)$; but neither was successful because the result was too poorly scaled to be represented by a single Chebyshev series over the whole interval (although the series with scaling (ii) converged much more rapidly than without the scaling). Using two $N = 30$ expansions on $[-200, -10]$ and $[-10, -1]$ gave all the roots of the Chebyshev series within no worse than 5.7×10^{-7} of the zeros of $f(\lambda)$. For this example, too, splitting into subintervals was more effective than any scaling function we could devise.

5. SUMMARY

C. Lanczos introduced the idea of "economizing" or "telescoping" a power series by approximating M terms of a Taylor series by an N-term Chebyshev series, where N is much smaller than M. In [9], he applied this idea to compute the roots of a cubic polynomial. First, just as we have done, he analyzed the cubic to put bounds on the search interval for the roots of interest. Then he approximated the cubic by a three-term Chebyshev series and calculated the roots of this quadratic to approximate those of the cubic. Since an arbitrary function $f(\lambda)$ may be represented as a power series and is therefore (in some sense) a polynomial of infinite degree, it follows that the algorithm we propose is really just Lanczos economization: Replacing the expensive function by a truncated Chebyshev series and finding the roots of the latter to approximate those of $f(\lambda)$ itself.

For expository purposes, we have chosen our examples such that $f(\lambda)$ is the determinant of a large matrix whose elements depend nonlinearly on λ. However, the "Lanczos economization" algorithm is completely general and can be applied to any $f(\lambda)$. The only constraint is the practical one that economization is only needed when $f(\lambda)$ is expensive.

The adaptive Clenshaw–Curtis strategy of systematically doubling N until the Chebyshev series of $f(\lambda)$ displays fast convergence is simple and effective. There is no general method for choosing a search interval, since this is highly problem-dependent, but usually the physics of the problem will dictate at least the approximate bounds on the range of interesting zeros, and this is sufficient.

The main unresolved difficulty is scaling; an unscaled $f(\lambda)$ may, especially if it is the determinant of a large matrix, vary by many orders of magnitude on the search interval. The Chebyshev series will, for sufficiently large N, have small *absolute* errors, relative to the maximum of $|f(\lambda)|$ $(= f_{\max})$, but the *relative* errors will be enormous on those parts of the interval where $|f(\lambda)| \ll f_{\max}$. A good general remedy is domain decomposition: Splitting the search interval into several pieces and computing a separate Chebyshev expansion on each.

However, a better strategy would be to multiply $f(\lambda)$ by a smooth, zero-free scaling function $S(\lambda)$ to reduce the variatons in the magnitude of the scaled function. Unfortunately, finding a good scaling function is hard. An open problem is to devise a good scaling strategy.

ACKNOWLEDGMENTS

This work was supported by the NSF through Grants OCE8812300, DMS8716766, ECS9012263, and OCE9119459 and by the Department of Energy through Grant KC070101. I thank Paul Swartztrauber and an anonymous reviewer for helpful discussions.

REFERENCES

1. J. P. Boyd, *Chebyshev and Fourier Spectral Methods* (Springer-Verlag, Heidelberg, 1989).
2. L. Fox and I. B. Parker, *Chebyshev Polynomials in Numerical Analysis* (Oxford Univ. Press, Oxford, 1968).
3. C. W. Clenshaw and A. R. Curtis, *Numer. Math.* **2,** 197 (1960).
4. W. M. Gentleman, *Commun. Assoc. Comput. Mach.* **15,** 353 (1972).
5. A. Ruhe, *SIAM J. Number. Anal.* **10** 674 (1973).
6. F. Acton, *Numerical Methods That Work* (Harper–Row, New York, 1970).
7. R. S. Lindzen, *Physics of the Atmosphere* (Cambridge Univ. Press, Cambridge, 1991).
8. L. Hayes and E. Wasserstrom, *J. Inst. Math. Appl.* **17,** 5 (1976).
9. V. N. Kublanovskaya, *SIAM J. Numer. Anal.* **7,** 532 (1970).
10. P. M. Anselone and L. B. Rall, *Numer. Math.* **10,** 38 (1968).
11. G. Peters and J. H. Wilkinson, *SIAM J. Numer. Anal.* **7,** 479 (1970).