Solving Secular Equations Stably and Efficiently *

Ren-Cang Li
Department of Mathematics
University of California at Berkeley
Berkeley, California 94720

April, 1993

Computer Science Division Technical Report UCB//CSD-94-851, University of California, Berkeley, CA 94720, December, 1994.

Dedicated to B. N. Parlett and W. Kahan on the occasion of their 60th birthdays

Abstract

A divide-and-conquer method for solving symmetric tridiagonal eigenproblems has evolved from work by Cuppen, Dongarra, Sorensen, Tang, and most recently Gu and Eisenstadt. At the heart of their methods is the solution of a so-called *Secular Equation.* Proposed here is a more efficient organization of the equation–solving process, including some crucial implementation details.

*This material is based in part upon work supported by Argonne National Laboratory under grant No. 20552402 and the University of Tennessee through the Advanced Research Projects Agency under contract No. DAAL03-91-C-0047, by the National Science Foundation under grant No. ASC-9005933, and by the National Science Infrastructure grants No. CDA-8722788 and CDA-9401156.

1 Introduction

A continuing element in divide-and-conquer algorithms descended from Cuppen's [4, 5] is the solution of an eigensystem that differs from diagonal by a rank–1 perturbation. Let $D = \mathrm{diag}\,(\delta_1, \cdots, \delta_n)$ be a real diagonal matrix, ρ a nonzero real number, and $z = (\zeta_1, \cdots, \zeta_n)^T$ a real vector; the perturbed system's matrix is

$$D + \frac{1}{\rho} z z^T. \tag{1}$$

Its eigenvalues are the zeros $x = \lambda_1, \cdots, \lambda_n$ of the *secular function* (see [7])

$$f(x) = \rho + \sum_{j=1}^{n} \frac{\zeta_j^2}{\delta_j - x}; \tag{2}$$

and the eigenvector corresponding to each λ_k is parallel to

$$(D - \lambda_k I)^{-1} z. \tag{3}$$

(Here I is the $n \times n$ identity matrix.) Numerical difficulties arose when these were not computed accurately enough to be orthogonal, but recent work [11, 9] has overcome these difficulties. Ours is a small contribution to solving the secular equation $f(x) = 0$ as accurately as necessary but faster than before. Certain details necessary for robustness in the code will be discussed too. Inattention to such details can cause accidents like *Division by Zero*, which the author has encountered while testing others' programs.

The paper is organized as follows: In Section 2, we study various historic ways to rationally interpolate the secular function (2) and then develop three different schemes for solving the equation $f(x) = 0$, two of which are essentially due to [3] and the other one is new and fastest among the three. The way to interpolate $f(x)$ in Section 2 is not always efficient because, in which, attention is largely paid to the positions of two nearby poles. Section 3 gives a closer look into cases when attention has to be paid not only to the positions of most relevant poles but also to weights over particular poles. Important implementation issues like securing initial guesses and selecting the best scheme are discussed in detail in Sections 4 and 6. Numerical examples with detailed explanation are given in Section 7. Discussions of various stopping criteria and justifications of our proposed stopping criteria, are presented in Section 5. Numerical examples with detailed explanation are given in Section 7. Section 8 presents our conclusions.

Before we proceed, let's make some convenient assumptions. Without loss of generality, we assume that

$$\delta_1 < \delta_2 < \cdots < \delta_n, \quad \text{and every } \zeta_j \neq 0.$$

Otherwise, deflation would give a new matrix of form (1) with a lower dimension [4, 5, 10]. Then it is easy to see that $f(x)$ has precisely n zeros λ_i, one in each open interval (δ_j, δ_{j+1}), and one to the right of δ_n if $\rho > 0$ or to the left of δ_1 otherwise. To simplify our presentation, in the following we assume $\rho > 0$, and set $\delta_{n+1} = \delta_n + \frac{z^T z}{\rho}$. Then the n eigenvalues $\lambda_1, \cdots, \lambda_n$ of the matrix (1) satisfy

$$\delta_i < \lambda_i < \delta_{i+1}, \quad i = 1, \cdots, n.$$

2 Secular Equation Solvers (I)

One obvious method to solve the secular equation $f(x) = 0$ is Newton's method. However, as argued in [3], since $f(x)$ is a rational function having poles at $\delta_1, \cdots, \delta_n$, Newton's method, based upon a local linear interpolation, may not be a good method. Bunch, Nielsen and Sorensen [3] proposed a method based upon rational osculatory interpolation. A major drawback of their method is that it takes too many iterations in certain circumstances, as we shall see.

The following subsection explores three rational approximations to $f(x)$, from which we will develop three iteration schemes to solve the equation $f(x) = 0$. One of them we call the *Middle Way* is new and fastest. Most of material in this section is presented for historic reasons and for comparisons.

2.1 Rational Osculatory Interpolations

In this section, we explore the osculatory interpolation of the secular function $f(x)$ in (2) by a combination of the following two kinds of simple rational functions:

$$F(x; p, q) \stackrel{\text{def}}{=} \frac{q}{p - x} \quad \text{and} \quad G(x; \delta, r, s) \stackrel{\text{def}}{=} r + \frac{s}{\delta - x}. \tag{4}$$

They will be used in conjunction with a partition of the secular function $f(x) = \rho + \psi_k(x) + \phi_k(x)$ where, for $k = 1, 2, \cdots,$ or n,

$$\psi_k(x) = \sum_{j=1}^{k} \frac{\zeta_j^2}{\delta_j - x}, \quad \phi_k(x) = \sum_{j=k+1}^{n} \frac{\zeta_j^2}{\delta_j - x}. \tag{5}$$

(Convention sets $\phi_n(x) \equiv 0$.) The choice of k will depend upon which eigenvalue λ_k is being computed. It is easy to see that for $\delta_k < x < \delta_{k+1}$

$$-\infty < \psi_k(x) < 0 < \phi_k(x) < +\infty.$$

For any approximation y to λ_k, we shall approximate each of $\psi_k(x)$ and $\phi_k(x)$ by a simpler form F or G chosen to match ψ_k and ϕ_k in value and derivative at $x = y$; in other words, we perform osculatory interpolation.

So far, nothing we have said differs from past practice [3, 5, 9]. What will be novel will be the way we choose which of F and G should be used with each of ψ_k and ϕ_k; in fact, we shall see that F is best not used at all.

2.1.1 Two Ways to Rationally Interpolate $\psi_k(x)$

Let y be a fixed approximation to λ_k somewhere between δ_k and δ_{k+1}. Now we have a choice. We can choose parameters p and q in $F(x; p, q)$ so that

$$F(y; p, q) = \psi_k(y), \quad F'(y; p, q) = \psi_k'(y).$$

Or we can choose $\delta = \delta_k$ to match the pole of $\psi_k(x)$ next to y (and λ_k) and then choose parameters r and s in $G(y; \delta_k, r, s)$ so that

$$G(y; \delta_k, r, s) = \psi_k(y),\ G'(y; \delta_k, r, s) = \psi_k'(y).$$

Here are the formulas for the parameters:

- In $F(x; p, q) = q/(p - x)$,

$$\begin{aligned} p &= y + \psi_k(y)/\psi_k'(y) && (6) \\ &= \frac{1}{\psi_k'(y)} \sum_{j=1}^{k} \frac{\zeta_j^2}{(\delta_j - y)^2}\delta_j, \\ q &= \psi_k(y)^2/\psi_k'(y) > 0. && (7) \end{aligned}$$

- In $G(x; \delta_k, r, s) = r + s/(\delta_k - x)$,

$$\begin{aligned} r &= \psi_k(y) - (\delta_k - y)\psi_k'(y) && (8) \\ &= \sum_{j=1}^{k-1} \frac{\delta_j - \delta_k}{(\delta_j - y)^2}\zeta_j^2 \le 0, \\ s &= (\delta_k - y)^2\psi_k'(y) > 0. && (9) \end{aligned}$$

In fact, $\zeta_k^2 \le s \le \sum_{j=1}^{k} \zeta_j^2$; if $k > 1$, $\sum_{j=1}^{k-1} \zeta_j^2/(\delta_j - y) = \psi_{k-1}(y) < r < 0$; and if $k = 1$, $r = 0$.

Let these assignments to p, q, r, s hold throughout the rest of this § 2.1.1. Here are some of the properties of the two osculatory interpolating functions that our subsequent analysis will exploit.

Proposition 1 $\quad \delta_1 < p < \delta_k < y$.

This proposition indicates that the pole p of $F(x; p, q)$ lies farther from y than δ_k does. Therefore $F(x; p, q)$ may approximate $\psi_k(x)$ poorly between its pole δ_k and y. This happens when ζ_k is relatively small, in which case λ_k lies very close to δ_k, and therefore y will do likewise.

Since the interpolating rationals are unique, as we can see from the above equations, they are identically equal to $\psi_k(x)$ if $k = 1$.

Theorem 1 *If $k \ge 2$, then for $\delta_k < x < +\infty$ and $x \ne y$*

$$G(x; \delta_k, r, s) < \psi_k(x) < F(x; p, q).$$

Proof: The inequality $F(x; p, q) > \psi_k(x)$ for $\delta_k < x < +\infty$ was proved by [3]. Now we prove $G(x; \delta_k, r, s) < \psi_k(x)$ for $\delta_k < x < +\infty$ in a similar way.

Let $g(x) = G(x; \delta_k, r, s) - \psi_k(x)$. It suffices to prove $g(x) < 0$ for $\delta_k < x < +\infty$. To this end, set

$$h(x) = g(x) \prod_{j=1}^{k} (\delta_j - x). \tag{10}$$

Then

$$\begin{aligned} h(x) &= r \prod_{j=1}^{k} (\delta_j - x) + (s - \zeta_k^2) \prod_{j=1}^{k-1} (\delta_j - x) \\ &\quad + \sum_{i=1}^{k-1} \zeta_i^2 \prod_{j=1, j\ne i}^{k} (\delta_j - x), \end{aligned} \tag{11}$$

is a polynomial of degree k. It is easy to verify that $h(y) = 0$ and $h'(y) = 0$, so we have

$$h(x) = \beta (x - y)^2 \prod_{j=1}^{k-2} (x - \gamma_j), \tag{12}$$

for some β, γ_j. (In fact, $\beta = (-1)^k r$, from (11).)

We claim that $\gamma_j < \delta_k$ for $1 \le j \le k-2$. The zeros of $h(x)$ are the zeros of $g(x)$. There is a zero of $g(x)$ in each interval (δ_j, δ_{j+1}) for $j = 1, \cdots, k-2$. Therefore $h(x)$ has $k-2$ of its zeros less than δ_k. As $y > \delta_k$, we have proved our claim.

Since $\beta = (-1)^k r$ and (from (8)) $r < 0$, so from (12) $\mathrm{sign}\,(h(x)) = (-1)^{k-1}$ when $\delta_k < x$. On the other hand, for $\delta_k < x < +\infty$, we find from (10) that $\mathrm{sign}\,(g(x)) = \mathrm{sign}\,(h(x))(-1)^k = -1$, which means $g(x) < 0$. ∎

2.1.2 Two Ways to Rationally Interpolate $\phi_k(x)$

Similarly, we can perform osculatory interpolations of $\phi_k(x)$ by $F(x; p, q)$ and by $G(x; \delta_{k+1}, r, s)$, respectively, such that

$$F(y; p, q) = \phi_k(y), \quad F'(y; p, q) = \phi_k'(y);$$

$$G(y; \delta_k, r, s) = \phi_k(y),\ G'(y; \delta_k, r, s) = \phi_k'(y).$$

It is easily verified that

$$\begin{aligned} p &= y + \phi_k(y)/\phi_k'(y) && (13) \\ &= \frac{1}{\phi_k'(y)} \sum_{j=k+1}^{n} \frac{\zeta_j^2}{(\delta_j - y)^2}\delta_j, && (14) \\ q &= \phi_k(y)^2/\phi_k'(y) > 0, && (15) \\ r &= \phi_k(y) - (\delta_{k+1} - y)\phi_k'(y) && (16) \\ &= \sum_{j=k+2}^{n} \frac{\delta_j - \delta_{k+1}}{(\delta_j - y)^2}\zeta_j^2 > 0, && (17) \\ s &= (\delta_{k+1} - y)^2 \phi_k'(y) > 0. && (18) \end{aligned}$$

Moreover $\zeta_{k+1}^2 \le s \le \sum_{j=k+1}^{n} \zeta_j^2$; if $k < n-1$, $0 < r < \phi_{k+1}$; and if $k = n-1$, $r = 0$.

F and G are identically equal to $\phi_k(x)$ if $k = n-1$.

Proposition 2 $\quad y < \delta_{k+1} < p < \delta_n$.

Proposition 2 indicates the same phenomenon as we observed in Proposition 1.

Theorem 2 $\quad$ *If $k \le n-2$, then for $y \ne x < \delta_{k+1}$*

$$F(x; p, q) < \phi_k(x) < G(x; \delta_{k+1}, r, s).$$

2.2 Solving $f(x) = 0$

Three ways come to mind to find the kth zero λ_k of the secular function (2) by combining different rational osculatory interpolating functions to $\psi_k(x)$ and $\phi_k(x)$ studied above.

The first one we are going to study is the one in [3], and we describe it as *Approaching from the Left* because the algorithm will produce a sequence of monotonically increasing approximations to λ_k provided the initial guess is between δ_k and λ_k. We reproduce it here for completeness.

We describe the second one as *Approaching from the Right* for the same reason.

The third iteration scheme is a new and fastest one which we shall call the *Middle Way*.

Assume, we have a guess y to λ_k (with $\delta_k < y < \delta_{k+1}$); Later, in Section 4, we shall discuss how to choose such a y.

2.2.1 Approaching from the Left

Assume for the moment $1 \leq k < n$.

To approach from the left, we interpolate $\psi_k(x)$ by $F(x; p, q)$ and $\phi_k(x)$ by $G(x; \delta_{k+1}, r, s)$, where p, q, r and s are determined by Equations (6), (7), (16) and (18). Instead of solving $\rho + \psi_k(x) + \phi_k(x) = 0$, we solve

$$\rho + \frac{q}{p - x} + r + \frac{s}{\delta_{k+1} - x} = 0 \tag{19}$$

for x. Equation (19) has two roots x of which just one lies between p and δ_{k+1}; that one is our new approximation $y + \eta$. Set $\Delta_{k+1} = \delta_{k+1} - y$. Then the correction η to y is

$$\eta = \frac{a - \sqrt{a^2 - 4bc}}{2c} \quad \text{if} \quad a \leq 0, \tag{20}$$

$$= \frac{2b}{a + \sqrt{a^2 - 4bc}} \quad \text{if} \quad a > 0, \tag{21}$$

where

$$\begin{aligned} a &= f(y)\left(\Delta_{k+1} + \frac{\psi_k(y)}{\psi_k'(y)}\right) - \Delta_{k+1}\psi_k(y)\left(1 + \frac{\phi_k'(y)}{\psi_k'(y)}\right), \\ b &= \Delta_{k+1} f(y) \frac{\psi_k(y)}{\psi_k'(y)}, \\ c &= \rho + r = \rho + \phi_k(y) - \Delta_{k+1}\phi_k'(y). \end{aligned}$$

Eventually, as $f(y) \to 0$, we find $a > 0$, so (21) will be used more often than (20).

The case for $k = n$ is special since $\phi_n \equiv 0$. Solving $\rho + \frac{q}{p-x} = 0$ gives

$$\eta = x - y = \frac{\rho + \psi_n(y)}{\rho \psi_n'(y)} \psi_n(y) = \frac{f(y)\psi_n(y)}{f'(y)\rho}. \tag{22}$$

Regarding the foregoing scheme, we have

Theorem 3 *If $\delta_k < y < \lambda_k$, then $\eta > 0$ and $y < y + \eta < \lambda_k$; if $\lambda_k < y < \delta_{k+1}$, then $\eta < 0$ and $y + \eta < \lambda_k < y$.*

Proof: It is clear from Theorems 1 and 2. ∎

Theorem 3 says, starting with an initial guess which is less than λ_k, the foregoing scheme will yield a sequence of approximations converging monotonically upwards to λ_k; on the other hand if the initial guess exceeds λ_k the first increment η will be negative enough to make the next approximation less than λ_k, and all later approximations will go upwards to λ_k again.

Remark: The above statement is true, generally, except for one case when the initial guess exceeds λ_k so much that the next approximation falls below δ_k. Recall that we can only guarantee $p < y + \eta < \delta_{k+1}$ while $p < \delta_k$ by Proposition 1. The author has encountered examples in which, because of inaccurate initial guess y, the next approximation by *Approaching from the Left* went indeed below δ_k. Care must be taken to prevent that from happening.

2.2.2 Approaching from the Right

Reversing to the choice of interpolating rationals for $\psi_k(x)$ and $\phi_k(x)$ in *Approaching from the Left*, we now interpolate $\psi_k(x)$ by $G(x; \delta_k, r, s)$ and $\phi_k(x)$ by $F(x; p, q)$, where p, q, r and s are determined by Equations (8), (9), (13) and (15). Set $\Delta_k = \delta_k - y$. Then the correction η to y is, again, given by (20) and (21), but with

$$\begin{aligned}
a &= f(y)\left(\Delta_k + \frac{\phi_k(y)}{\phi_k'(y)}\right) - \Delta_k \phi_k(y)\left(1 + \frac{\psi_k'(y)}{\phi_k'(y)}\right), \\
b &= \Delta_k f(y) \frac{\phi_k(y)}{\phi_k'(y)}, \\
c &= \rho + r = \rho + \psi_k(y) - \Delta_k \psi_k'(y).
\end{aligned}$$

From which we see eventually $a > 0$.

When $k = n$, we partition and interpolate the secular function $f(x)$ the same as we did for the case $k = n-1$, but choose our new approximation to λ_n to be the root of $\rho + G(x; \delta_{n-1}, r, s) + F(x; p, q) = 0$ which lies between δ_n and δ_{n+1} if any. It is easy to see that $F(x; p, q) \equiv \phi_{n-1}(x)$, and there is such a root provided $\rho + r > 0$. The correction η to y is

$$\begin{aligned} \eta &= \frac{a + \sqrt{a^2 - 4bc}}{2c} \quad \text{if} \quad a \geq 0, \\ &= \frac{2b}{a - \sqrt{a^2 - 4bc}} \quad \text{if} \quad a < 0, \end{aligned} \tag{23}$$

with a, b, c as shown above for $k = n-1$, which can be simplified as

$$\begin{aligned} a &= (\Delta_{n-1} + \Delta_n) f(x) - \Delta_{n-1}\Delta_n f'(y), \\ b &= \Delta_{n-1}\Delta_n f(y), \\ c &= \rho + r = \rho + \psi_{n-1}(y) - \Delta_{n-1}\psi'_{n-1}(y) \\ &= f(y) - \Delta_{n-1}\psi'_{n-1}(y) - \frac{\zeta_n^2}{\Delta_n}. \end{aligned}$$

(23) is usable provided $f(y) - \Delta_{n-1}\psi'_{n-1}(y) - \frac{\zeta_n^2}{\Delta_n} > 0$ since otherwise $y + \eta < \delta_n$ or η is infinite. Eventually as $f(y) \to 0$, so does $f(y) - \Delta_{n-1}\psi'_{n-1}(y) - \frac{\zeta_n^2}{\Delta_n}$ become positive.

The foregoing scheme is called *Approaching from the Right* because of this:

Theorem 4 *If $\lambda_k < y < \delta_{k+1}$, then $\eta < 0$ and $\lambda_k < y + \eta < y$. If $\delta_k < y < \lambda_k$ and $k < n$, then $\eta > 0$ and $y < \lambda_k < y + \eta$. If $\delta_n < y < \lambda_n$ and y is not too close to δ_n, then $y + \eta > \lambda_n$.*

Proof: It is clear from Theorems 1 and 2. ∎

Remark: As we remarked after Theorem 3, in Theorem 4 it is possible for $y + \eta$ to jump above δ_{k+1} when $\delta_k < y < \lambda_k$. This must be prevented in practice.

2.2.3 The Middle Way: a New Method

A new way (we call it the *Middle Way*) to solve the secular equation $f(x) = 0$ will be developed here. Unlike the previous two ways, it need not yield a sequence of approximations that converges monotonically to λ_k, but it converges faster as we will see.

We interpolate both $\psi_k(x)$ and $\phi_k(x)$ by rationals of type $G(x;\delta,r,s)$, taking both nearby poles into consideration. To be specific, when $0 < k < n$ we let

$$r + \frac{s}{\delta_k - x} \quad \text{approximate} \quad \text{to} \quad \psi_k(x), \text{and}$$
$$R + \frac{S}{\delta_{k+1} - x} \quad \text{approximate} \quad \text{to} \quad \phi_k(x),$$

where

$$\left\{ \begin{array}{l} s = \Delta_k^2 \psi_k'(y) > 0, \\ r = \psi_k(y) - \Delta_k \psi_k'(y) \le 0, \end{array} \right. \quad \text{and} \quad \left\{ \begin{array}{l} S = \Delta_{k+1}^2 \phi_k'(y) > 0, \\ R = \phi_k(y) - \Delta_{k+1} \phi_k'(y) \ge 0, \end{array} \right. \tag{24}$$

and $\Delta_k = \delta_k - y < 0 < \Delta_{k+1} = \delta_{k+1} - y$ as before. We compute our new "better" approximation $y + \eta$ to λ_k by solving the equation

$$\rho + r + \frac{s}{\delta_k - x} + R + \frac{S}{\delta_{k+1} - x} = 0. \tag{25}$$

Equation (25) has two roots x, one of them infinite if $\rho + r + R = 0$. The root we need is the one between δ_k and δ_{k+1}. Set $\eta = x - y$. Then the correction η to y has forms (20) and (21) with

$$\begin{array}{rcl} a & = & (\Delta_k + \Delta_{k+1}) f(y) - \Delta_k \Delta_{k+1} f'(y), \\ b & = & \Delta_k \Delta_{k+1} f(y), \\ c & = & f(y) - \Delta_k \psi_k'(y) - \Delta_{k+1} \phi_k'(y). \end{array}$$

Eventually $a > 0$.

The case $k = n$ is exactly the same as in § 2.2.2.

Remark: Quadratic convergence of *Approaching from the left* was proved in [3]. A similar proof does likewise for *Approaching from the Right*, and proves that the *Middle Way* converges at least quadratically.

3 Secular Equation Solvers (II): A Close Look

The *Middle Way* outperforms *Approaching from the Left* and *Approaching from the Right* because it takes both nearby poles into considerations while each of latter two uses one of the poles. However, as we shall see, the *Middle*

Way still behaves badly in a few circumstances. The following is a simple intuitive explanation for why. In $\frac{\zeta_j^2}{\delta_j - x}$, ζ_j^2 is the weight over the pole δ_j and controls how much role the pole δ_j plays in the secular function $f(x)$. Most of the time the simple interpolating rational $r + \frac{s}{\delta_k - x}$ approximates $\psi_k(x)$ very well, but there are situations when s overestimates ζ_k^2 so much that iterations are forced to move slowly towards the desired roots. Recall $\frac{s-\zeta_k^2}{\delta_k - x}$ functions for the rest of the poles in $\psi_k(x)$. Similar things happen to ϕ_k, too. In this section, we shall provide ways to conquer this as well as difficulties when just employing two nearby poles is not enough.

The case $k = n$ will not be handled here since the *Middle Way* (*Approaching from the Right* also) has used the two nearby poles on the left of λ_n already while there is no pole on its right. We begin with a new look at our iteration formulas whose flexibility will be of big help.

3.1 Iteration Formulas

Let y be a fixed approximation to λ_k somewhere between δ_k and δ_{k+1}. The *Middle Way* is, eventually, based upon an osculatory interpolation of $f(x)$ at y by

$$Q(x; c, s, S) \stackrel{\text{def}}{=} c + \frac{s}{\delta_k - x} + \frac{S}{\delta_{k+1} - x}, \tag{26}$$

for which

$$c + \frac{s}{\delta_k - y} + \frac{S}{\delta_{k+1} - y} = f(y), \tag{27}$$

$$\frac{s}{(\delta_k - y)^2} + \frac{S}{(\delta_{k+1} - y)^2} = f'(y). \tag{28}$$

However, if we start with $Q(x; c, s, S)$ satisfying (27) and (28), we cannot determine $Q(x; c, s, S)$ uniquely because of three unknowns and only two equations available. Therefore an additional condition has to be imposed in order to determine $Q(x; c, s, S)$. For the moment, let's not worry about this, but instead assume $Q(x; c, s, S)$ is a rational of form (26) with (27) and (28) satisfied.

The idea for computing a correction η to y for the next ("better") approximation $y + \eta$ to λ_k is to solve the equation $Q(x; r, s, S) = 0$. Let

$$\Delta_k = \delta_k - y,\ \Delta_{k+1} = \delta_{k+1} - y,\ x = y + \eta.$$

$Q(x; c, s, S) = 0$ yields

$$c(\Delta_k - \eta)(\Delta_{k+1} - \eta) + s(\Delta_{k+1} - \eta) + S(\Delta_k - \eta) = 0,$$

giving

$$c\eta^2 - \eta[c(\Delta_k + \Delta_{k+1}) + s + S] + c\Delta_k\Delta_{k+1} + s\Delta_{k+1} + S\Delta_k = 0. \tag{29}$$

Proposition 3 *With c, s and S subject to (27) and (28), we have*

$$\begin{aligned} c(\Delta_k + \Delta_{k+1}) + s + S &= (\Delta_k + \Delta_{k+1})f(y) - \Delta_k\Delta_{k+1}f'(y), \\ c\Delta_k\Delta_{k+1} + s\Delta_{k+1} + S\Delta_k &= \Delta_k\Delta_{k+1}f(y). \end{aligned}$$

Proof: It follows from (27) and (28) that

$$\begin{aligned} s &= \frac{\Delta_k^2}{\Delta_k - \Delta_{k+1}}(f(y) - c - \Delta_{k+1}f'(y)), \\ S &= \frac{\Delta_{k+1}^2}{\Delta_{k+1} - \Delta_k}(f(y) - c - \Delta_k f'(y)). \end{aligned}$$

Substituting them into $c(\Delta_k + \Delta_{k+1}) + s + S$ and $c\Delta_k\Delta_{k+1} + s\Delta_{k+1} + S\Delta_k$ leads to the desired results. ∎

Proposition 3 illustrates a surprising fact that the iteration formula by solving (29) depends only upon c, alone. As surely, the equations (27) and (28) are solvable for any c. Without worrying about global convergence, any c will give rise an ultimately at least quadratically convergent iteration scheme!

In the case $s > 0$ and $S > 0$ in which we are interested, it follows from (29) that

$$\begin{aligned} \eta &= \frac{a - \sqrt{a^2 - 4bc}}{2c} \quad \text{if} \quad a \leq 0, && (30) \\ &= \frac{2b}{a + \sqrt{a^2 - 4bc}} \quad \text{if} \quad a > 0, && (31) \end{aligned}$$

where

$$\begin{aligned} a &= (\Delta_k + \Delta_{k+1})f(y) - \Delta_k\Delta_{k+1}f'(y), \\ b &= \Delta_k\Delta_{k+1}f(y). \end{aligned}$$

In what follows, for a particular iteration scheme, we shall provide the necessary number c, as well as a proof for $s > 0$ and $S > 0$. Different choices of c give rise to different iteration schemes. By choosing c properly, a very efficient iteration scheme may be obtained.

The *Middle Way* falls into the category, i.e., $s > 0$ and $S > 0$, and

$$\begin{aligned} c &= f(y) - \Delta_k \psi_k'(y) - \Delta_{k+1}\phi_k'(y) \\ &= f(y) - \Delta_{k+1} f'(y) - \psi_k'(y)(\delta_k - \delta_{k+1}) \\ &= f(y) - \Delta_k f'(y) - \phi_k'(y)(\delta_{k+1} - \delta_k). \end{aligned}$$

Reorganization of c is for the reader to see its differences from those in some other schemes to which we are about to get.

3.2 The Fixed Weight Method

We have been aware that the weight ζ_k^2 or ζ_{k+1}^2 may be overestimated in the *Middle Way*. Although not all overestimations are harmful, there are cases where iterations move slowly because of overestimations. In order to handle these cases efficiently, we propose the following scheme so-called the *Fixed Weight Method* because it fixes one of the weights ζ_k^2 and ζ_{k+1}^2 while satisfying (27) and (28).

1. *The Case λ_k closer to δ_k:* We set $s = \zeta_k^2$, then

$$s = \zeta_k^2, \tag{32}$$

$$\begin{aligned} S &= \Delta_{k+1}^2 \left(f'(y) - \frac{\zeta_k^2}{\Delta_k^2} \right) \qquad (33) \\ &= \zeta_{k+1}^2 + \sum_{j \neq k,k+1} \frac{\Delta_{k+1}^2}{\Delta_j^2} \zeta_j^2 > \zeta_{k+1}^2, \\ c &= f(y) - \frac{\zeta_k^2}{\Delta_k} - \Delta_{k+1} \left(f'(y) - \frac{\zeta_k^2}{\Delta_k^2} \right) \\ &= f(y) - \Delta_{k+1} f'(y) - \frac{\zeta_k^2}{\Delta_k^2}(\delta_k - \delta_{k+1}). \qquad (34) \end{aligned}$$

2. *The Case λ_k closer to δ_{k+1}:* We set $S = \zeta_{k+1}^2$, then

$$\begin{aligned} s &= \Delta_k^2 \left(f'(y) - \frac{\zeta_{k+1}^2}{\Delta_{k+1}^2} \right) \qquad (35) \\ &= \zeta_k^2 + \sum_{j \neq k,k+1} \frac{\Delta_k^2}{\Delta_j^2} \zeta_j^2 > \zeta_k^2, \end{aligned}$$

$$
\begin{aligned}
S &= \zeta_{k+1}^2, && (36)\\
c &= f(y) - \Delta_k f'(y) - \frac{\zeta_{k+1}^2}{\Delta_{k+1}^2}(\delta_{k+1} - \delta_k). && (37)
\end{aligned}
$$

The following theorem gives a basic property of the interpolating function $Q(x; c, s, S)$ and the next approximation $y + \eta$ to λ_k.

Theorem 5 *If c, s and S are defined by (32), (33) and (34), then $f(x) \leq Q(x; c, s, S)$ for $\delta_k < x < \delta_{k+1}$ and either $\delta_k < y < y + \eta < \lambda_k < \delta_{k+1}$ or $\delta_k < y + \eta < \lambda_k < y < \delta_{k+1}$; if, on the other hand, c, s and S are defined by (35), (36) and (37), then $f(x) \geq Q(x; c, s, S)$ for $\delta_k < x < \delta_{k+1}$ and either $\delta_k < \lambda_k < y + \eta < y < \delta_{k+1}$ or $\delta_k < y < \lambda_k < y + \eta < \delta_{k+1}$.*

3.3 Gragg's Scheme, a Scheme of Ultimately Cubic Convergence

Gragg proposed to choose c, s and S so that $Q(x; c, s, S)$ matches $f(x)$ at y up to the second derivative. In another word, besides (27) and (28), it is also required that

$$
\frac{s}{(\delta_k - y)^3} + \frac{S}{(\delta_{k+1} - y)^3} = \frac{f''(y)}{2}, \tag{38}
$$

which, together with (27) and (28), yield

$$
\begin{aligned}
s &= \frac{\Delta_k^3 \Delta_{k+1}}{\Delta_k - \Delta_{k+1}} \left(\frac{f'(y)}{\Delta_{k+1}} - \frac{f''(y)}{2} \right) \\
&= \zeta_k^2 + \frac{(\delta_k - y)^3}{\delta_k - \delta_{k+1}} \sum_{i \neq k,\, k+1} \frac{\delta_i - \delta_{k+1}}{(\delta_i - y)^3} \zeta_i^2 > \zeta_k^2, \\
S &= \frac{\Delta_k \Delta_{k+1}^3}{\Delta_{k+1} - \Delta_k} \left(\frac{f'(y)}{\Delta_k} - \frac{f''(y)}{2} \right) \\
&= \zeta_{k+1}^2 + \frac{(\delta_{k+1} - y)^3}{\delta_{k+1} - \delta_k} \sum_{i \neq k,\, k+1} \frac{\delta_i - \delta_k}{(\delta_i - y)^3} \zeta_i^2 > \zeta_{k+1}^2, \\
c &= f(y) - (\Delta_k + \Delta_{k+1}) f'(y) + \Delta_k \Delta_{k+1} \frac{f''(y)}{2}.
\end{aligned}
$$

This scheme needs to compute the second derivative of the secular function $f(x)$. Thus it needs more work.

Remark: Gragg's scheme will yield a sequence of approximations which converge monotonically to the desired eigenvalue λ_k for $1 \leq k < n$ (ref. [8] and notice the difference between their secular function and ours). The interpolation of $f(x)$ for finding λ_n should be done in the same way as for finding λ_{n-1}, except monotonic convergence is lost.

3.4 Using Three Poles When Necessary: a Hybrid Scheme

The *Middle Way* and the *Fixed Weight Method* can be combined to design more powerful secular equation solvers by properly switching between the two. But there are cases where three poles have to be used in order to make iterations go faster. For $m = 2, \cdots, n-1$, set

$$f_m(x) = \rho + \sum_{j=1, j\neq m}^{n} \frac{\zeta_j^2}{\delta_j - x},$$

which is the secular function $f(x)$ with the mth term in the summation removed. It is easy to see that $f_m(x)$ has a zero between δ_{m-1} and δ_{m+1}. Numerically, we have discovered the following cases are possible difficult ones:

1. $\delta_k < \lambda_k < \frac{\delta_k + \delta_{k+1}}{2}$ and $f_k(x)$ has a zero between δ_k and λ_k;
2. $\frac{\delta_k + \delta_{k+1}}{2} < \lambda_k < \delta_{k+1}$ and $f_{k+1}(x)$ has a zero between λ_k and δ_{k+1}.

To see why the first case may be difficult, we let $\zeta_k^2 \to 0$, then, as functions of ζ_k^2, λ_{k-1} goes monotonically upward until it hits δ_k while λ_k goes monotonically downward until it hits the zero of $f_k(x)$ between δ_k and λ_k. Now, on the contrary, let ζ_k^2 go back from zero to its original value, what we will see is the exact contrary phenomena. Based on this simple observation, we can think, roughly, that λ_{k-1} depends largely on the pole δ_k and its weight ζ_k^2 and, therefore, the *Fixed Weight Method* should be good at finding it. On the other hand, λ_k depends on the pole δ_k and its weight ζ_k^2 and the zero of $f_k(x)$ where it starts from and which is controlled roughly by the the poles δ_{k-1} and δ_{k+1} with appropriate weights. Similar intuitive arguments apply to the second case above.

The treatment of the two cases is almost identical. In what follows, we discuss the first case only. A natural way to handle the first case is to interpolate $f(x)$ with the following simple rational:

$$\widetilde{Q}(x; c, s, S) \stackrel{\text{def}}{=} c + \frac{s}{\delta_{k-1} - x} + \frac{\zeta_k^2}{\delta_k - x} + \frac{S}{\delta_{k+1} - x}. \tag{39}$$

The parameters c, s and S can be determined by either interpolating $f_k(x) = \rho + \psi_{k-1}(x) + \phi_k(x)$ in the way that the *Middle Way* does or in the way that the *Fixed Weight Method* does depending on situations. Once we have (39), its zero between δ_k and δ_{k+1} can be computed in various iterative ways with negligible cost. However, care has to be taken in evaluating $\widetilde{Q}(x; c, s, S)$ at a given point. As a matter of fact, because of roundoff, sometimes (though rarely) computed $\widetilde{Q}(y; c, s, S)$ differs significantly from computed $f(y)$ though the two should be the same by interpolation in theory. It turns out that we can evaluate $\widetilde{Q}(x; c, s, S)$ in an indirectly way to avoid this from happening. Since cimputed $f(y)$ is available at the time we interpolate the secular function $f(x)$, we do not compute $\widetilde{Q}(y; c, s, S)$ at all while simply setting it to be $f(y)$. At the very next time when we need to compute $\widetilde{Q}(x; c, s, S)$ we update $f(y)$ by adding a correction to it because

$$\begin{aligned}
\widetilde{Q}(x; c, s, S) &= \widetilde{Q}(y; c, s, S) + (\widetilde{Q}(x; c, s, S) - \widetilde{Q}(y; c, s, S)) \\
&= f(y) + (x - y) \left(\frac{s}{(\delta_{k-1} - y) - (x - y)} \right. \\
&\qquad \left. + \frac{\zeta_k^2}{(\delta_k - y) - (x - y)} + \frac{S}{(\delta_{k+1} - y) - (x - y)} \right).
\end{aligned}$$

Subsequent evaluation of $\widetilde{Q}(x; c, s, S)$ is done in the same way by adding correction to its value at the previous x.

According to our experience, proper treatment of poles and their weights is crucial and delicate, especially in difficult cases as we discussed above. It accelerates convergence significantly in the sense that it reduces the overall average number of iterations per eigenvalue finding and keeps the peak number of iterations for finding an eigenvalue reasonably small. which is essential for avoiding load-balancing in parallel computations. In view of this, we propose the following scheme–the *Hybrid Scheme* which combines the *Middle Way* and the *Fixed Weight Method* in a clever way and which of course employs three poles when necessary: Suppose we are computing λ_k and suppose $\delta_k < \lambda_k < (\delta_k + \delta_{k+1})/2$. In practice, we compute $\lambda_k - \delta_k$ instead of λ_k itself. We have an initial guess $\delta_k + \tau$ for λ_k for which $\delta_k < \delta_k + \tau < \lambda_k$ is guaranteed (ref. Section 4). The value of the secular function is evaluated in such a way

$$f(\delta_k + \tau) = f_k(\delta_k + \tau) + \frac{\zeta_k^2}{-\tau} < 0$$

that $f_k(\delta_k+\tau)$ is obtained as a by-product. At this point, we will make a decision between using two poles or three poles:

```
if  fk(δk + τ) > 0, then
    two poles δk and δk+1 are used;
else
    three poles δk−1, δk and δk+1 are used.
end if
```

After the decision is made, we use the *Fixed Weight Method* to interpolate $f(x)$ if the decision favors two poles or interpolate $f_k(x)$ if the decision favors three poles, and do one iteration to get an new approximation τ_1. If $f(\delta_k+\tau_1)<0$ and $|f(\delta_k+\tau_1)|>0.1\times|f(\delta_k+\tau)|$, we switch to the *Middle Way* for the next iteration starting at $\delta_k+\tau_1$. From now on, just for guarding, we compare the value $f_{\rm new}$ of the secular function at the newest approximation with its value $f_{\rm pre}$ at the previous one and another switch is made from the current iterative scheme to the other one if $f_{\rm new}f_{\rm pre}>0$ and $|f_{\rm new}|>0.1\times|f_{\rm pre}|$.

The philosophy behind our switch making is that, taking $\delta_k<\delta_k+\tau<\lambda_k<\frac{\delta_k+\delta_{k+1}}{2}$ for example where $\delta_k+\tau$ is an initial guess, after first interpolation using the *Fixed Weight Method* $\delta_k+\tau<\delta_k+\tau_1<\lambda_k$. Now if $|f(\delta_k+\tau_1)|>0.1\times|f(\delta_k+\tau)|$, it indicates that iteration is going slow with the *Fixed Weight Method*. Generally, it is always a danger when the values of $f(x)$ at two consecutive approximations have the same sign but the lastest value improves little in comparing with the previous one.

If, however, we are computing λ_k under $(\delta_k+\delta_{k+1})/2<\lambda_k<\delta_{k+1}$, similar principle can be applied in a straightforward way.

Remark: A theorem similar to Theorem 5 holds for $\widetilde{Q}(x;c,s,S)$ if we use the *Fixed Weight Method* to interpolate $f_k(x)$ or $f_{k+1}(x)$.

4 Initial Guesses

An iteration that could ultimately converge quadratically, which is quite fast, may get converge slowly (if at all) from a sufficiently bad first guess. This sad possibility becomes a probability when ζ_k or ζ_{k+1} is tiny compared with the other ζ_j's, in which case λ_k is very close to δ_k or δ_{k+1}. In extreme cases, as we have already observed, the iterations *Approaching from the left* or *Right* can even jump out from between δ_k and δ_{k+1}.

In what follows we will present an inexpensive way to obtain relatively accurate initial guesses. At the same time we shall show how to decide which of δ_k and δ_{k+1} is closes to λ_k. This decision is important because, if decided wrongly, roundoff could cause an iterate to collide with an endpoint δ_k or δ_{k+1}, or even overshoot it. A collision would soon lead to *Division by Zero*; overshooting could jeopardize subsequent convergence. To avoid these problems, we shall translate the origin temporarily, while λ_k is being computed, to whichever of δ_k or δ_{k+1} is closer.

A natural and effective way to make the correct decision is to look at the sign of $f\left(\frac{\delta_k+\delta_{k+1}}{2}\right)$. If this value is positive, we know λ_k is closer to δ_k than to δ_{k+1} and thus the origin should be translated to δ_k; otherwise λ_k is closer to δ_{k+1} and the origin should be translated to there. If, however, roundoff obscures the sign of $f\left(\frac{\delta_k+\delta_{k+1}}{2}\right)$, in which case λ_k is almost half way between the two poles, the origin could be translated to either one of them without making much difference. The computation of $f\left(\frac{\delta_k+\delta_{k+1}}{2}\right)$ can be done in such a way that an initial guess y is obtained as a by-product.

First we consider the case $1 \le k < n$.

Rewrite the secular function (2) as $f(x) = g(x) + h(x)$, where

$$g(x) = \rho + \sum_{j=1, j\neq k,k+1}^{n} \frac{\zeta_j^2}{\delta_j - x}, \quad \text{and} \quad h(x) = \frac{\zeta_k^2}{\delta_k - x} + \frac{\zeta_{k+1}^2}{\delta_{k+1} - x}. \tag{40}$$

We choose our initial guess y to be that one of the two roots of the equation

$$g\left(\frac{\delta_k + \delta_{k+1}}{2}\right) + h(y) = 0. \tag{41}$$

lying between δ_k and δ_{k+1}, where $g\left(\frac{\delta_k+\delta_{k+1}}{2}\right)$ was retained from the computation of $f\left(\frac{\delta_k+\delta_{k+1}}{2}\right)$. (Thus it is a gift!) By sketching the graph of the sum of the last two terms on the left hand-side of (41), we can tell without any difficulty which root is needed. In case $f\left(\frac{\delta_k+\delta_{k+1}}{2}\right) \ge 0$, Equation (41) should be solved for $\tau = y - \delta_k$, while in case $f\left(\frac{\delta_k+\delta_{k+1}}{2}\right) < 0$, it should be solved for $\tau = y - \delta_{k+1}$. Define $\Delta = \delta_{k+1} - \delta_k$ and $c = g\left(\frac{\delta_k+\delta_{k+1}}{2}\right)$. Here are the formulas for τ:

$$\tau = y - \delta_K \quad = \quad \frac{a - \sqrt{a^2 - 4bc}}{2c} \quad \text{if} \quad a \le 0, \tag{42}$$

$$= \quad \frac{2b}{a + \sqrt{a^2 - 4bc}} \quad \text{if} \quad a > 0,$$

where if $f\left(\frac{\delta_k+\delta_{k+1}}{2}\right) \geq 0$,

$$K = k,\ a = c\Delta + (\zeta_k^2 + \zeta_{k+1}^2),\ b = \zeta_k^2 \Delta, \tag{43}$$

and if $f\left(\frac{\delta_k+\delta_{k+1}}{2}\right) < 0$,

$$K = k+1,\ a = -c\Delta + (\zeta_k^2 + \zeta_{k+1}^2),\ b = -\zeta_{k+1}^2 \Delta. \tag{44}$$

Theorem 6 *If $f\left(\frac{\delta_k+\delta_{k+1}}{2}\right) > 0$, then τ given by (42) and (43) satisfies*

$$\delta_k < \delta_k + \tau < \lambda_k < \frac{\delta_k + \delta_{k+1}}{2};$$

If $f\left(\frac{\delta_k+\delta_{k+1}}{2}\right) < 0$, then τ given by (42) and (44) satisfies

$$\frac{\delta_k + \delta_{k+1}}{2} < \lambda_k < \delta_{k+1} + \tau < \delta_{k+1}.$$

Proof: One way to see these is to sketch graphs. It can also be seen by subtracting one from the other of the following two equations

$$\begin{aligned} \frac{\zeta_k^2}{\delta_k - \lambda_k} + \frac{\zeta_{k+1}^2}{\delta_{k+1} - \lambda_k} &= -g(\lambda_k), \\ \frac{\zeta_k^2}{\delta_k - y} + \frac{\zeta_{k+1}^2}{\delta_{k+1} - y} &= -g\left(\frac{\delta_k + \delta_{k+1}}{2}\right), \end{aligned}$$

which produces

$$\begin{aligned} &(\lambda_k - y)\left(\frac{\zeta_k^2}{(\delta_k - \lambda_k)(\delta_k - y)} + \frac{\zeta_{k+1}^2}{(\delta_{k+1} - \lambda_k)(\delta_{k+1} - y)}\right) \\ &= \left(\frac{\delta_k + \delta_{k+1}}{2} - \lambda_k\right) \sum_{j \neq k, k+1} \frac{\zeta_j^2}{\left(\delta_j - \frac{\delta_k+\delta_{k+1}}{2}\right)(\delta_j - \lambda_k)}. \end{aligned} \tag{45}$$

Thus $\lambda_k - y$ has the same sign as has $\frac{\delta_k+\delta_{k+1}}{2} - \lambda_k$. ∎

For the case of $k = n$, we partition the secular function $f(x) = g(x) + h(x)$ as when $k = n-1$. An initial guess y is obtained basically by solving

$$g\left(\frac{\delta_n + \delta_{n+1}}{2}\right) + h(y) = 0.$$

A detail but simple analysis yields the following formula for $\tau = y - \delta_n$:

- *The case:* $\frac{\delta_n+\delta_{n+1}}{2} \leq \lambda_n$, i.e., $f\left(\frac{\delta_n+\delta_{n+1}}{2}\right) \leq 0$.

  1. If $g\left(\frac{\delta_n+\delta_{n+1}}{2}\right) \leq -h(\delta_{n+1})$, then $\tau = y - \delta_n = z^T z/\rho$.
  2. If $g\left(\frac{\delta_n+\delta_{n+1}}{2}\right) > -h(\delta_{n+1})$, then

  $$\begin{aligned} \tau = y - \delta_n &= \frac{a + \sqrt{a^2 - 4bc}}{2c} \quad \text{if} \quad a \geq 0, \\ &= \frac{2b}{a - \sqrt{a^2 - 4bc}} \quad \text{if} \quad a < 0, \end{aligned} \tag{46}$$

  where $\Delta = \delta_n - \delta_{n-1}$, $c = g\left(\frac{\delta_n+\delta_{n+1}}{2}\right)$ and

  $$a = -c\Delta + (\zeta_{n-1}^2 + \zeta_n^2),\ b = -\zeta_n^2 \Delta. \tag{47}$$

  It can be proved that in this case $\frac{\delta_n+\delta_{n+1}}{2} \leq \lambda_n < \delta_n + \tau < \delta_{n+1}$.

- *The case:* $\frac{\delta_n+\delta_{n+1}}{2} > \lambda_n$, i.e., $f\left(\frac{\delta_n+\delta_{n+1}}{2}\right) > 0$. Then $g\left(\frac{\delta_n+\delta_{n+1}}{2}\right) > -h(\delta_{n+1})$. We compute $\tau = y - \delta_n$ by (46) and (47). It is easy to show that in this case $\delta_n < \delta_n + \tau < \lambda_n < \frac{\delta_n+\delta_{n+1}}{2}$.

The approach to compute initial guesses we just proposed costs marginal since to make program robust we have to calculate $f\left(\frac{\delta_k+\delta_{k+1}}{2}\right)$ anyway. An alternative approach proposed by [3] is to solve the equation (41) with $g\left(\frac{\delta_k+\delta_{k+1}}{2}\right)$ replaced by $g(\delta_{k+1})$ or $g(\delta_k)$, which yields two guesses that happen to be a lower bound and an upper bound for λ_k. First of all their guesses require extra work. Second, this extra work may not be worth doing. Take the case $\lambda_k < \frac{\delta_k+\delta_{k+1}}{2}$, for example. In this case, this lower bound is worse than our guess while the upper bound may possibly be greater than $\frac{\delta_k+\delta_{k+1}}{2}$. In a divide-and-conquer code from Sorensen and Tang[1], they simply solve the equation (41) with $g\left(\frac{\delta_k+\delta_{k+1}}{2}\right)$ replaced by ρ to obtain initial guesses for $1 \leq k \leq n-1$. They handle the case $k = n$ by solving $\rho + \psi_{n-1}(\delta_{n+1}) + \phi_{n-1}(y) = 0$. In any event, we believe our approach should be better averagely. Our numerical experience confirms our intuition.

[1] I would like to thank Prof. D. Sorensen and Dr. Peter Tang for sending me their divide and conquer code.

5 Two Stopping Criteria

People have discovered in order to guarantee the computed eigenvectors by (3) to be fully orthogonal one must be able to compute the distances between each λ_i and δ_j to almost full accuracy [10, 11]. So simulated "double" double precision was invented to evaluate the value of the secular function extra precisely when necessary. To guarantee full accuracy of computed distances $\delta_i - \lambda_j$, the stopping criterion was set to be

$$|\eta| \leq c\epsilon_m \min\{|\delta_k - x|, |\delta_{k+1} - x|\}, \tag{48}$$

where x is the current iterate, η is the last iterative correction that was computed, ϵ_m is the machine's roundoff threshold and c is fairly small constant [1, 5, 11]. ($c = 2^4$ in Sorensen and Tang's code.) However we decide to stop when

$$\eta^2 \leq \epsilon_m \min\{|\delta_k - x|, |\delta_{k+1} - x|\}(|\eta_0| - |\eta|), \tag{49}$$

where η_0 is the correction before last. This new stopping criterion often save one iteration per eigenvalue found. The following observation due to W. Kahan justifies (49).

Let $\{x_j\}_{j=1}^{\infty}$ be a sequence of numbers, produced by some rapidly convergent iteration scheme, such that $\lim_{j\to\infty} x_j = z$. If $|x_{j+1} - x_j|/|x_j - x_{j-1}|$ are decreasing for $j \geq k$, and if $|x_{k+1} - x_k|/|x_k - x_{k-1}| < 1$, then

$$|x_{k+1} - z| < \frac{|x_{k+1} - x_k|^2}{|x_k - x_{k-1}| - |x_{k+1} - x_k|}.$$

To see why this observation works, let's denote $\gamma = |x_{k+1} - x_k|/|x_k - x_{k-1}|$. For $i > k$, we have

$$x_{i+1} - x_i = \prod_{j=k}^{i} \frac{x_{j+1} - x_j}{x_j - x_{j-1}}(x_k - x_{k-1}).$$

which gives $|x_{i+1} - x_i| \leq \gamma^{i-k+1}|x_k - x_{k-1}|$. Therefore

$$\begin{aligned} |x_{k+1} - z| &= \left| \sum_{i=k+1}^{\infty} (x_i - x_{i+1}) \right| \\ &\leq |x_k - x_{k-1}| \sum_{i=k+1}^{\infty} \gamma^{i-k+1} = \frac{|x_{k+1} - x_k|^2}{|x_k - x_{k-1}| - |x_{k+1} - x_k|}, \end{aligned}$$

as required.

Recently, Gu and Eisenstadt [9] found a new way to compute eigenvectors after all eigenvalues are computed. This new way makes the simulated "double" double precision unnecessary, and the stopping criterion was set to be

$$|f(x)| \leq n\epsilon_m \left(\rho + \sum_{j=1}^{n} \left| \frac{\zeta_j^2}{\delta_j - x} \right| \right) \tag{50}$$

by [9]. However, for large n, it may allow too much error while for small n there is a danger that it might never be satisfied. Note that not every term in secular equation is of the same magnitude. Generally, only two terms will dominate all others. In our code, we decide to compute error bounds with marginal costs at each iteration. Our implementation is as follows: We compute $\psi_k(x)$ by summing up each term from $j = 1$ to k, while $\phi_k(x)$ from $j = n$ to $k+1$ (refer to (5)) since hopefully in this way we add terms from small magnitude to large ones. Note that we actually work with

$$f(\delta_K + \tau) = \rho + \sum_{j=1}^{n} \frac{\zeta_j^2}{(\delta_j - \delta_K) - \tau}, \tag{51}$$

where $\tau = x - \delta_K$ and $K = k$ if λ_k comes more close to δ_k than to other δ_j and $K = k+1$ otherwise. It is easy to show that

$$|f(\delta_K + \tau) - (\text{Computed}\, f(\delta_K + \tau))| \leq \epsilon_m e,$$

where

$$e = 2\rho + \sum_{j=1}^{k} (k - j + 6) \left| \frac{\zeta_j^2}{\delta_j - x} \right| + \sum_{j=n}^{k+1} (j - k + 5) \left| \frac{\zeta_j^2}{\delta_j - x} \right| + |f(\delta_K + \tau)|,$$

which can be computed recursively on our way to compute $\psi_k(x)$ and $\phi_k(x)$ and costs about $2n$ additions for each iteration cycle. On the other hand, if τ is a neighbor of the desired zero τ^* to the function (51), i.e., $|\tau - \tau^*| \leq \epsilon_m |\tau|$, then

$$\begin{aligned} |f(\delta_K + \tau) - 0| &\leq |\tau - \tau^*|\, |f'(\delta_K + \tau)| + O(|\tau - \tau^*|^2) \\ &\leq \epsilon_m |\tau|\, |f'(\delta_K + \tau)| + O(|\tau - \tau^*|^2). \end{aligned}$$

In view of these, we set our stopping criterion to be

$$|f(\delta_K + \tau)| \leq e\epsilon_m + \epsilon_m |\tau|\, |f'(\delta_K - \tau)| \tag{52}$$

This stopping criterion is more reasonable and tighter than (50). The last term in (52) is usually much smaller than $e\epsilon_m$ since $|\tau| \leq |\delta_{k+1} - \delta_k|/2$.

6 Choosing the Right Scheme

With many different iterative schemes at hand, which one is the right one that we should use in order to achieve best performance that we could get from the divide-and-conquer method? Obviously, they could not be equally efficient. Based on our intuition and numerical experience, we think the *Hybrid Scheme* is the best choice.

Approaching from the Left has been used for finding zeros of the secular function (2) for over a decade since it first appeared in [3]. Why was it favored and why were not the other schemes? Two possible reasons are:

- Only a small portion of the total cost for Cuppen's divide-and-conquer algorithm is spent on solving secular equations[2]. Therefore a faster secular equation solver might affect the overall performance of the algorithm so little that people's attention were not caught.

- *Approaching from the Left* yields a sequence of approximations that converge monotonically to the desired eigenvalue. Monotonicity is good, for example, no need to compute error bound (refer to Section 5), but simply stop whenever monotonicity is lost.

The rational interpolation resulting in *Approaching from the Left* forces δ_{k+1} to be the pole of the interpolating rational $G(x;\delta_{k+1},r,s)$ for $\phi_k(x)$, while pushes the pole δ_k to the left to p as the pole of the interpolating rational $F(x;p,q)$ for $\psi_k(x)$. The former is quite reasonable; on the other hand, the latter raises a question: Are rationals of form $F(x;p,q)$ good enough as candidates to interpolate $\psi_k(x)$, especially for the case when x comes close to δ_k? The answer turns out to be "*No*", as it becomes clear from numerical results in the following section. This gives us an idea that *Approaching from the Left* works better in the case when λ_k comes closer to δ_{k+1} than in the case when λ_k closer to δ_k.

The above argument applies to *Approaching from the Right.* And the *Middle Way* should work perfectly in any case in the sense that it matches the better one of the two.

The average number of iteration of Gragg's cubic scheme, which needs to compute the second derivative of the secular function $f(x)$, is about the same as that of our *Hybrid Scheme*. But, as we will see later, it does iterates more

[2]This is not true if only eigenvalues are requested because, in this case, the total cost is $O(n^2)$ [6].

in some cases where poles and weights have to be taken into considerations more closely.

Few cases have been encountered by the author in which there are big differences in numbers of iterations for finding λ_n. Our suggestion is strongly supported by numerical results below.

7 Numerical Examples

In numerical results that follows, double precision arithmetic was used throughout. Both stopping criteria will be used for testing. We will demonstrate numerically

1. How good are our initial guesses?
2. *Approaching from the Left* and *Right* are bad schemes;
3. The stopping criterion (52) usually terminates computations 1–2 iterations earlier than (49);
4. Comparisons among the *Middle Way*, the *Fixed Weight Method*, Gragg's scheme and the *Hybrid Scheme*.

Our first example is taken from [11]: $n = 4$, $\delta_1 = 1$, $\delta_2 = 2-\beta$, $\delta_3 = 2+\beta$ and $\delta_4 = 3\frac{1}{3}$, where $\beta = 10^{-\ell}$ is a parameter chosen on purpose to make λ_1 close to δ_2 and λ_3 close to δ_2. We make $\lambda_2 = 2$ in exact arithmetic, half way between δ_2 and δ_3, by letting $w^T = (2, \beta, \beta, 2)$, $\rho = \|w\|^{-2}$ and $z = (\zeta_1, \zeta_2, \zeta_3, \zeta_4)^T = w/\|w\|$.

In the following table, we compare the numbers of iterations taken by using *Approaching from the Left* with our initial guesses (the third row) and with those taken by using Sorensen and Tang's secular equation solver code. Iteration stops whenever (49) is satisfied, and extra precise evaluations [11] of secular functions are invoked.

Table 1: Initial Guess Issue[3]

	$\beta = 10^{-3}$				$\beta = 10^{-6}$				$\beta = 10^{-10}$			
Eigenvalues	1	2	3	4	1	2	3	4	1	2	3	4
Left	5	2	13	4	5	1	24	4	5	2	37	4
Sorensen & Tang's	12	15	16	4	18	26	25	5	28	37	38	4

[3]The initial guess is not counted. This is also the case for all following tables.

This example shows our initial guesses provide better starting points. But still the third eigenvalue λ_3 takes lots of iterations before required accuracy is reached. The reason is that λ_3 is extremely close to δ_3 and *Approaching from the Left* behaves badly in this situation. In exact arithmetic, $\lambda_2 = 2$ lies exactly in the half way between δ_2 and δ_3. However, as Table 1 indicates, a few iterations are still required even with our way of initial guessing, which should give correct result in this situation in exact arithmetic. As a matter of fact, the corresponding problem that computer had seen is the one after a few roundoffs. The second eigenvalue to that problem is $2+O(\epsilon_m)$, which, in general, does not lie in the half way between rounded δ_2 and rounded δ_3. At the end of computation, extra precise evaluations of the secular function basically yields λ_2 as $2+\eta$ with η computed to certain relative accuracy which is unnecessarily lot more for just computing the eigenvalue but is necessary for computing the full orthogonal eigenvectors later on [11].

From now on, our way of initial guessing is always used except in Sorensen and Tang's code. The following table exhibits how many iterations are required respectively by the three different zero finders we discussed in Section 2. Invoke extra precise evaluation [11] whenever necessary.

Table 2: Schemes in Subsection 2.2

	$\beta = 10^{-3}$				$\beta = 10^{-6}$				$\beta = 10^{-10}$			
Eigenvalues	1_2	2_3	3_3	4_4	1_2	2_3	3_3	4_4	1_2	2_3	3_3	4_4
Left	5	2	13	4	5	1	24	4	5	2	37	4
Right	13	2	5	4	23	1	6	4	36	2	5	4
Middle	5	2	5	4	5	1	6	4	5	2	5	4

In this table each item i_j in the second row says that the corresponding column is for finding the ith eigenvalue with origin translated to δ_j. The first column lists which iteration schemes in Subsection 2.2 are being used. Others are numbers of iterations required.

Table 2 confirms our previous speculations. As λ_2 sits "right" on the half way between δ_2 and δ_3, the three different schemes required almost the same number of iterations; *Approaching from the Left* is suitable for finding λ_1 while *Approaching from the Right* is obviously not because λ_1 comes very close to δ_2; the contrary conclusion holds for finding λ_3 because λ_3 comes very close to δ_3. In any event, the *Middle Way* is the best among the three.

The stopping criterion (52) usually terminates an eigenvalue computing 1–2 iterations earlier than the stopping criterion (49). The following table illustrates the point.

Table 3: (49) *vs.* (52)

	$\beta = 10^{-3}$				$\beta = 10^{-6}$				$\beta = 10^{-10}$			
Eigenvalues	1_2	2_3	3_3	4_4	1_2	2_3	3_3	4_4	1_2	2_3	3_3	4_4
Middle with (49)	5	2	5	4	5	1	6	4	5	2	5	4
Middle with (52)	4	0	5	3	4	0	5	3	3	0	3	3

Table 4 lists the numbers of iterations for an example of a rank–1 perturbed diagonal eigensystem arising in applying the divide-and-conquer algorithm to the glued Wilkinson matrix, i.e., the matrices in *Test 1* of [11]. Here $n = 30$. The first column refers to which eigenvalue and the corresponding temporary origin translated to. The 2nd to 4th columns refer to the three iteration schemes in Subsection 2.2 with stopping criterion (49). The 5th column refers to the divide-and-conquer code from Sorensen and Tang. The last column refers to the *Middle Way* with stopping criterion (52).

Table 4

	Left	*Right*	*Middle*	Sorensen & Tang	*Middle & (52)*
1_1	2	2	2	2	1
2_2	2	2	2	3	1
3_3	2	2	2	4	1
4_4	2	2	2	4	1
5_5	2	2	2	4	1
6_6	2	2	2	4	1
7_7	2	2	2	4	1
8_8	2	2	2	4	1
9_9	3	2	2	5	2
10_{10}	2	2	2	3	1
11_{12}	4	19	4	8	2
12_{12}	28	11	11	28	9
13_{13}	3	2	2	5	1
14_{15}	4	18	3	6	2
15_{15}	27	16	16	27	15
16_{17}	2	6	2	7	1
17_{18}	3	24	3	23	2
18_{18}	34	13	12	37	10
19_{19}	7	2	2	7	1
20_{21}	2	3	2	6	1
21_{21}	16	17	17	19	16
22_{22}	7	2	2	7	1
23_{23}	3	2	3	6	2
24_{24}	5	2	2	7	2
25_{25}	3	3	3	7	3
26_{26}	5	2	2	5	1
27_{27}	3	3	3	3	2
28_{29}	2	6	2	7	2
29_{29}	8	7	7	10	6
30_{30}	1	1	1	6	0

Table 4 has demonstrated all the points we have made about the schemes in Section 2 and about our initial guesses over Sorensen and Tang's. On the other hand, it also explodes the inability of the *Middle Way* to compute the 12th, 15th, 18th and 21st eigenvalues efficiently. The reason is exactly what we have made at the beginning of Section 3 and which motivates us to create the *Fixed Weight Method* and the *Hybrid Scheme*. The following table presents numbers of iterations required by the *Middle Way*, the *Fixed Weight Method*, Gragg's scheme and the *Hybrid scheme* on finding these four "difficult" eigenvalues. From now on, all tests are done under the stopping criterion (52) (except for Gragg's scheme) as we have observed neither stopping criterion affects demonstrations on the efficiency of a particular scheme. For Gragg's scheme, we stop whenever we detect loss of monotonicity, and thus last iterations might be wastes.

Table 5

Eigenvalues	12	15	18	21	Total	Per Eig	Peak
Middle	9	15	10	16	91	3.03	16
Fixed	2	2	2	2	42	1.40	5
Gragg's	2	2	3	2	56	1.87	4
Hybrid	1	1	1	2	38	1.27	4

Here, "Total" refers to the total number of iterations required by the corresponding scheme for finding all 30 eigenvalues; "Per Eig" the average number of iterations required per eigenvalue; and "Peak" the largest one among numbers of iterations for finding each of the eigenvalues. Three poles were used by the *Hybrid Scheme* for the 12th, 15th and 18th eigenvalues. One might interpret Table 5 in a wrong way that the *Fixed Weight Method* is good enough and there is no necessity for developing the *Hybrid Scheme*. To discover the drawback of the *Fixed Weight Method*, we invented another artificial problem in which $n = 50$ and in which the 31st eigenvalue is so special that the *Fixed Weight Method* takes as many as 31 iterations to compute it. In that problem, δ_{32} and δ_{33} agree as many as 9 decimal digits, while ζ_{32}^2 is much smaller than ζ_{33}^2. The λ_{31}, however, is fairly away from both of its nearby poles though it comes closer to δ_{32}. In this case, setting $S = \zeta_{32}^2$ underestimate the role played by δ_{33} and the rest other than δ_{32} so much that the iterations generated by the *Fixed Weight Method* go unbearably slow. Numerical data are displayed in Table 6. One thing we have to say is that for the *Middle Way* the first iteration overshoots because of roundoff errors in computing the correction. We handle this by restarting the iteration at the middle point between δ_{31} and δ_{32}.

Table 6

	Middle	*Fixed*	Gragg's	*Hybrid*
Eigenvalue 31	4	31	10	3
Total	159	181	150	145
Per Eig	3.18	3.62	3.00	2.90
Peak	6	31	10	5

To see how iterations go, we list values of the secular function after each iteration:

Middle: `1.4D+08` → `-2.0D+00` → `1.5D-02` → `3.6D-07` → `0.0D+00`
Fixed: `1.4D+08` → `6.9D+07` → `3.5D+07` → `1.7D+07` → `8.7D+06`
→ `4.3D+06` → `2.2D+06` → `1.1D+06` → `5.4D+05`
→ `2.7D+05` → `1.4D+05` → ⋯
Gragg's: `1.4D+08` → `6.9D+07` → `2.6D+07` → `7.2D+06` → `1.2D+06`
→ `1.0D+05` → `2.6D+03` → `1.1D+01` → `1.1D-02`
→ `9.4D-10` → `0.0D+00`
Hybrid: `1.4D+08` → `1.4D-02` → `4.2D-07` → `9.0D-16`

The following is another table justifying that including three most relevant poles helps. It is about a problem of $n = 30$.

Table 7

Eigenvalues	12	15	19	22	Total	Per Eig	Peak
Middle	6	7	6	10	65	2.17	10
Fixed	7	7	8	7	64	2.13	8
Gragg's	5	6	6	5	65	2.17	6
Hybrid	3	3	6	6	53	1.77	6

In theory all the schemes we studied share a *fundamental* property that every correction is in the right direction. To be more specific, whenever the value of the secular function is negative the next correct must be positive and vice versa. However, it does not hold in the face of roundoff. To handle this, we simply do one Newton step instead whenever fatal roundoff in computing corrections is detected. In Table 7, for the *Fixed Weight Method* the first iterations for the 19th and 22nd eigenvalues are Newton steps.

Numerous other examples generated either randomly or artificially have been run. The results turn out to be very satisfactory. And randomly generated problems do not give difficulties. Table 8 below lists a few more data on real rank-1 perturbed problems extracted from applying the divide-and-conquer algorithm to either randomly generated symmetric tridiagonal matrices or to symmetric tridiagonal matrices obtained by reducing randomly generated symmetric dense matrices to tridiagonal forms.

Table 8

	$n = 100$			$n = 364$			$n = 700$		
	Total	Per Eig	Peak	Total	Per Eig	Peak	Total	Per Eig	Peak
Middle	175	1.75	10	1084	2.98	6	2125	3.04	7
Fixed	146	1.46	6	1106	3.04	6	2157	3.08	7
Gragg's	185	1.85	6	1031	2.83	5	2054	2.93	5
Hybrid	146	1.46	5	1074	2.95	5	2093	2.99	5

It is worth mentioning an problem suggested to the author by W. B. Gragg: $D = \mathrm{diag}\,(1, 2, \cdots, n)$, $\rho = 1$ and $z = (1, 10^{-1}, \cdots, 10^{-(n-1)})^T$. In practice, there will be lots of deflation for large n. But for the purpose of testing the robustness of our code, we run it on this problem for n as large as 100 without deflating. The numerical results indicate our code has no difficulties in solving the problems.

8 Conclusions

We have studied different rational interpolations for the secular function, each of which has different strong points and based upon which many schemes have been proposed and studied. A proper combination leads to the *Hybrid Scheme* which competes with Gragg's cubic convergent scheme on random problems, however, as our numerical results indicated, there are artificial problems in which the regions of at least quadratic convergence for the *Hybrid Scheme* are larger than those of cubic convergence for Gragg's scheme. Also the *Hybrid Scheme* keeps peak number of iterations relatively small, which is extremely helpful for solving the secular equation in parallel because the total time is determined roughly by whichever eigenvalue takes the largest number of iterations.

We also discussed a few implementation details for making a robust code and for achieving the desired solution as accurately as possible.

Acknowledgments. The author is grateful for the supervision of Professor W. Kahan. Thanks also go to Professors J. Demmel and B. N. Parlett for valuable discussions.

References

[1] J. L. Barlow, Error analysis of update methods for the symmetric eigenvalue problem, submitted for publication.

[2] C. F. Borges and W. B. Gragg, A parallel divide and conquer algorithm for the generalized real symmetric definite tridiagonal eigenproblem, Working paper, (1992).

[3] J. R. Bunch, Ch. P. Nielsen, and D. C. Sorensen, Rank-one modification of the symmetric eigenproblem, *Numer. Math.,* **31**(1978), 31–48.

[4] J. J. M. Cuppen, A divide and conquer method for the symmetric tridiagonal eigenproblem, *Numer. Math.,* **36**(1981), 177–195.

[5] J. J. Dongarra and D. C. Sorensen, A fully parallel algorithm for the symmetric eigenvalue problem, *SIAM J. Sci. Stat. Comput.*, **8**(1987), s139–s154.

[6] D. Gill and E. Tadmor, An $O(N^2)$ method for computing the eigensystem of $N \times N$ symmetric tridiagonal matrices by the divide and conquer approach, *SIAM J. Sci. Stat. Comput.*, **11**(1990), 161–173.

[7] G. H. Golub, Some modified matrix eigenvalue problems, *SIAM rev.*, **15**(1973), 318–334.

[8] W. B. Gragg, J. R. Thornton, and D. D. Warner, Parallel divide and conquer algorithms for the symmetric tridiagonal eigenproblem and bidiagonal singular value problem, in Modeling and Simulation, vol. 23, part 1, W. G. Vogt and M. H. Mickle, eds., Univ. Pittsburgh School of Engineering, 1992, 49–56.

[9] Ming Gu and S. C. Eisenstadt, A stable and efficient algorithm for the rank–one modification of the symmetric eigenproblems, Research Report YaleU/DCS/RR–916, 1992.

[10] W. Kahan, Symmetric rank-1 perturbed diagonal's eigensystem, preprint, 1989.

[11] D. C. Sorensen, and P. T. P. Tang, On the orthogonality of eigenvectors computed by divide-and-conquer techniques, *SIAM J. Numer. Anal.*, **28**(1991), 1752–1775.