A ROOT–FINDING METHOD FOR THE GENERALIZED TRIDIAGONAL EIGENPROBLEM

Kuiyuan Li and Matoteng Ncube

Abstract. In this paper a globally convergent, parallel method for the generalized real symmetric eigenproblem is presented. The method can be easily implemented on computers.

1. Introduction. In this paper we consider the generalized eigenvalue problem

$$Ax = \lambda Mx, \tag{1}$$

where A and M are both real symmetric tridiagonal and one of them, say M, is positive definite. Assumptions imply real eigenvalues and the existence of an M–orthogonal basis of eigenvectors. Eigenvalue problems with such a special structure arise in many applications, such as numerical solutions of Sturm–Liouville and radial Schrödinger equations [1,9] and finite element approximation for free longitudinal vibrations problems of non–uniform rod [10].

There are some approaches to solving (1). The first approach is to reduce (1) to a standard eigenvalue problem [11]:

$$L^{-1}AL^{-T}(L^T x) = \lambda(L^T x), \tag{2}$$

where $M = LL^T$ is the Cholesky factorization of M. Then the eigenvalue problem (2) can be solved by many efficient algorithms, such as the QR algorithm [11], the bisection/multisection algorithms [7], the divide–conquer algorithm [2], and the homotopy algorithm [5]. However, this approach is less attractive because it cannot take advantage of the tridiagonal form of A and M, and a full matrix $L^{-1}AL^{-T}$ is generated in the process. Furthermore, the accuracy of this method also depends on the conditioning of M since the inverse of L is explicitly required. The complexity of this method is $O(n^3)$.

The second approach, the QZ method [11], disregards the symmetry and the tridiagonal structure of the problem, and after a direct phase requiring $O(n^3)$ operations, enters an iterative phase requiring $O(n^3)$ operations to simultaneously reduce A and M to triangular–quasi–triangular form. This algorithm is not very sensitive to the conditioning of M though it is even more expensive than the first approach.

The third approach is the bisection/multisection method [8]. This parallel method is very easy to implement in practice and is reliable and accurate. However, the rate of the convergence is linear.

Another approach is the homotopy–like method [6]. The method employs the determinant evaluation, split–and–merge strategy and the Laguerre's iteration. Since Laguerre's iteration requires us to compute the first and second derivatives of $f(\lambda) = \det(A - \lambda M)$, more computations are needed.

Recently, the Durand–Kerner root finding method [4] was proposed for this problem. The method does not require us to compute $f'(\lambda)$ or $f''(\lambda)$. Although it always converges in practice [4], and no counterexample has been given so far, no proof of this property has been given.

In this paper, we present a parallel method which converges monotonically and quadratically without computing $f'(\lambda)$ or $f''(\lambda)$.

2. Proposed Root Finding Method. Let $p(x) = \frac{1}{\alpha}\det(A - \lambda M)$, where $\alpha = (-1)^n \det(M)$. Since M is positive definite, $\alpha \neq 0$. Clearly, to find the eigenvalues, one only needs to find the zeros of $p(x)$.

For this purpose, we propose the following root–finding method. Let

$$x_i^{(k+1)} = x_i^{(k)} - \frac{p(x_i^{(k)})}{(x_i^{(k)} - x_1^{(k)})\cdots(x_i^{(k)} - x_{i-1}^{(k)})(x_i^{(k)} - y_{i+1}^{(k)})\cdots(x_i^{(k)} - y_n^{(k)})}, \qquad (3)$$

for $i = 1, 2, \ldots, n$ and

$$y_i^{(k+1)} = y_i^{(k)} - \frac{p(y_i^{(k)})}{(y_i^{(k)} - x_1^{(k)})\cdots(y_i^{(k)} - x_{i-1}^{(k)})(y_i^{(k)} - y_{i+1}^{(k)})\cdots(y_i^{(k)} - y_n^{(k)})}, \qquad (4)$$

for $i = 1, 2, \ldots, n$, where the numbers $x_i^{(k)}$ and $y_i^{(k)}$, $i = 1, 2, \ldots, n$, are sets of approximations to n zeroes of $p(x)$. The numbers $x_i^{(k+1)}$ and $y_i^{(k+1)}$, $i = 1, 2, \ldots, n$, are sets of better approximations than $x_i^{(k)}$ and $y_i^{(k)}$, $i = 1, 2, \ldots, n$, respectively.

Clearly, (3) and (4) give us a fully parallel method for finding all zeros of $p(x)$, that is, $x_i^{(k+1)}$ and $y_i^{(k+1)}$, $i = 1, 2, \ldots, n$ can be computed independently.

Theorem 2.1. Assume $x_i^{(k)} < \lambda_i < y_i^{(k)} < x_{i+1}^{(k)} < \lambda_{i+1} < y_{i+1}^{(k)}$, $i = 1, 2, \ldots, n-1$ and for some k, then for any i,

$$x_i^{(k)} < x_i^{(k+1)} < \lambda_i, \text{ for } k > 0,$$

and

$$y_i^{(k)} > y_i^{(k+1)} > \lambda_i, \text{ for } k > 0.$$

Proof.

$$x_i^{(k+1)} = x_i^{(k)} - \frac{p(x_i^{(k)})}{(x_i^{(k)} - x_1^{(k)}) \cdots (x_i^{(k)} - x_{i-1}^{(k)})(x_i^{(k)} - y_{i+1}^{(k)}) \cdots (x_i^{(k)} - y_n^{(k)})}$$

$$= x_i^{(k)} - \frac{(x_i^{(k)} - \lambda_1)(x_i^{(k)} - \lambda_2) \cdots (x_i^{(k)} - \lambda_n)}{(x_i^{(k)} - x_1^{(k)}) \cdots (x_i^{(k)} - x_{i-1}^{(k)})(x_i^{(k)} - y_{i+1}^{(k)}) \cdots (x_i^{(k)} - y_n^{(k)})}$$

$$= x_i^{(k)} - \frac{(x_i^{(k)} - \lambda_1) \cdots (x_i^{(k)} - \lambda_{i-1})}{(x_i^{(k)} - x_1^{(k)}) \cdots (x_i^{(k)} - x_{i-1}^{(k)})} \frac{(x_i^{(k)} - \lambda_{i+1}) \cdots (x_i^{(k)} - \lambda_n)}{(x_i^{(k)} - y_{i+1}^{(k)}) \cdots (x_i^{(k)} - y_n^{(k)})} \cdot (x_i^{(k)} - \lambda_i).$$

Since

$$\frac{(x_i^{(k)} - \lambda_1) \cdots (x_i^{(k)} - \lambda_{i-1})}{(x_i^{(k)} - x_1^{(k)}) \cdots (x_i^{(k)} - x_{i-1}^{(k)})} > 0,$$

$$\frac{(x_i^{(k)} - \lambda_{i+1}) \cdots (x_i^{(k)} - \lambda_n)}{(x_i^{(k)} - y_{i+1}^{(k)}) \cdots (x_i^{(k)} - y_n^{(k)})} > 0,$$

and

$$x_i^{(k)} - \lambda_i < 0,$$

$$x_i^{(k+1)} > x_i^{(k)}.$$

On the other hand,

$$\frac{(x_i^{(k)} - \lambda_1)\cdots(x_i^{(k)} - \lambda_{i-1})}{(x_i^{(k)} - x_1^{(k)})\cdots(x_i^{(k)} - x_{i-1}^{(k)})} < 1,$$

$$\frac{(x_i^{(k)} - \lambda_{i+1})\cdots(x_i^{(k)} - \lambda_n)}{(x_i^{(k)} - y_{i+1}^{(k)})\cdots(x_i^{(k)} - y_n^{(k)})} < 1,$$

and

$$x_i^{(k)} - \lambda_i < 0,$$

therefore,

$$x_i^{(k+1)} < x_i^{(k)} - (x_i^{(k)} - \lambda_i) = \lambda_i.$$

Similarly, we can prove that $y_i^{(k)} > y_i^{(k+1)} > \lambda_i$, for $k > 0$.

Theorem 2.1 implies that $x_i^{(k)}$ and $y_i^{(k)}$ converge to λ_i strictly monotonically.

Theorem 2.2. Assume $x_i^{(k)} < \lambda_i < y_i^{(k)} < x_{i+1}^{(k)} < \lambda_{i+1} < y_{i+1}^{(k)}$, $i = 1, 2, \ldots, n-1$, then $\lambda_i - x_i^{(k+1)} < y_i^{(k)} - \lambda_i$ and $y_i^{(k+1)} - \lambda_i < \lambda_i - x_i^{(k)}$ for sufficiently large k.

Proof. Let

$$\epsilon = \max_{1 \le j \le n} \{\lambda_j - x_j^{(k)}, y_j^{(k)} - \lambda_j\}.$$

For any i,

$$|x_i^{(k+1)} - \lambda_i|$$

$$= \left|(x_i^{(k)} - \lambda_i) - \frac{(x_i^{(k)} - \lambda_1)(x_i^{(k)} - \lambda_2)\cdots(x_i^{(k)} - \lambda_n)}{(x_i^{(k)} - x_1^{(k)})\cdots(x_i^{(k)} - x_{i-1}^{(k)})(x_i^{(k)} - y_{i+1}^{(k)})\cdots(x_i^{(k)} - y_n^{(k)})}\right|$$

$$= \left|(x_i^{(k)} - \lambda_i) - \prod_{j=1}^{i-1} \frac{x_i^{(k)} - \lambda_j}{x_i^{(k)} - x_j^{(k)}} \prod_{j=i+1}^{n} \frac{x_i^{(k)} - \lambda_j}{x_i^{(k)} - y_j^{(k)}} \cdot (x_i^{(k)} - \lambda_i)\right|$$

$$= \left|(x_i^{(k)} - \lambda_i) - \prod_{j=1}^{i-1} \left(1 + \frac{x_j^{(k)} - \lambda_j}{x_i^{(k)} - x_j^{(k)}}\right) \prod_{j=i+1}^{n} \left(1 + \frac{x_j^{(k)} - \lambda_j}{x_i^{(k)} - y_j^{(k)}}\right) \cdot (x_i^{(k)} - \lambda_i)\right|$$

$$= |(x_i^{(k)} - \lambda_i)| \left| \sum_{j=1}^{i-1} \frac{x_j^{(k)} - \lambda_j}{x_i^{(k)} - x_j^{(k)}} + \sum_{j=i+1}^{n} \frac{x_j^{(k)} - \lambda_j}{x_i^{(k)} - y_j^{(k)}} + O(\epsilon^2) \right|$$

$$\leq |x_i^{(k)} - \lambda_i| \left| \epsilon \cdot \left(\sum_{j=1}^{i-1} \frac{1}{x_i^{(k)} - x_j^{(k)}} + \sum_{j=i+1}^{n} \frac{1}{y_j^{(k)} - x_i^{(k)}} \right) O(\epsilon^2) \right|$$

$$\leq \epsilon \left(\epsilon \left(\sum_{j=1}^{i-2} \frac{1}{\lambda_{i-1} - \lambda_j} + \frac{1}{x_i - \lambda_{i-1}} + \sum_{j=i+1}^{n} \frac{1}{\lambda_j - \lambda_i} \right) + O(\epsilon^2) \right)$$

$$\leq \epsilon^2 \left(\sum_{j=1}^{i-2} \frac{1}{\lambda_{i-1} - \lambda_j} + \frac{1}{\lambda_i - \lambda_{i-1} - \epsilon} + \sum_{j=i+1}^{n} \frac{1}{\lambda_j - \lambda_i} \right) + O(\epsilon^3)$$

$$\leq c\epsilon^2 \leq c(y_i^{(k)} - \lambda_i)^2.$$

Since $y_i^{(k)}$ converges to λ_i, for sufficiently large k, $c(y_i^{(k)} - \lambda_i) < 1$. Therefore, $\lambda_i - x_i^{(k)} < y_i^{(k)} - \lambda_i$.

Similarly, we can show that $y_i^{(k+1)} - \lambda_i < \lambda_i - x_i^{(k)}$.

It seems that to compute the approximation of λ_i, two sequences are needed, and therefore, more computations are needed. However, Theorem 2.2 shows that more computations may not be really needed. We do two iterations at each step.

Theorem 2.2 also implies that the root finding method converges quadratically.

Remark. The method we discussed is a parallel method, but at the same time it is possible to use a Gauss–Seidel–like procedure, that is, we use the new approximations, $x_j^{(m+1)}$, where $j = 1, 2, \dots, i-1$, to replace $x_j^{(m)}$ and $y_j^{(m+1)}$, where $j = n, n-1, \dots, i+1$, to replace $y_j^{(m)}$ in (3) and (4). We compute $x_j^{(m+1)}$ and $y_j^{(m+1)}$ in the following order: $x_1^{(m+1)}$, $y_n^{(m+1)}$, $x_2^{(m+1)}$, $y_{n-1}^{(m+1)}$, $\dots$, $x_n^{(m+1)}$, and $y_1^{(m+1)}$. In this case, the sequences converge to λ_i super quadratically.

3. Numerical Results. In this section, we present our numerical results.

Our method for finding all eigenvalues can be divided into two stages: root isolation and root extraction. The root isolation can be done by using the Sturm sequence. This technique is discussed in detail in [4]. This step results in a list of intervals containing single roots. When multiple roots are present, the isolation process continues until the required precision is reached. The root extraction step uses the proposed root–finding method.

Experiment 1. We implemented our algorithm LGT and the routine RSG in EISPACK [3] on equation (1). The matrices A and M are obtained from piecewise linear finite element discretization [9] of the Sturm–Liouville problem

$$-\frac{d}{dx}\left(p(x)\frac{du}{dx}\right) + q(x)u = \lambda u,$$

where $u = u(x)$, $0 < x < \pi$, $u(0) = u'(\pi) = 0$, and $p(x) > 0$. When $[0, \pi]$ is divided into $n + 1$ subintervals of equal length, equation (1) is obtained. Here, both A and M are symmetric tridiagonal, and positive definite. To make the problem simpler, we use $p(x) = 1$ and $q(x) = 6$.

The experiment was conducted on a Sun–1000 workstation with IEEE standard and machine precision $\epsilon \approx 2.2 \times 10^{-16}$.

For the generalized symmetric tridiagonal eigenproblem, the subroutine RSG in EISPACK is somewhat less attractive since it transforms the generalized eigenproblem $Ax = \lambda Mx$ to a standard eigenproblem $\tilde{T}y = \lambda y$ of a dense matrix $\tilde{T}$. It cannot take advantage of the tridiagonal structure of the pencil (A, M) and clearly requires $O(n^3)$ flops. The RSG is also sensitive to the conditioning of M. We conduct the comparison here mainly because it is the only algorithm available in EISPACK for eigenproblems of symmetric definite pencils.

Table 1 shows the results of our algorithm LGT and the algorithm RSG. The execution time apparently shows that the complexities of LGT and RSG are $O(n^2)$ and $O(n^3)$, respectively for the generalized tridiagonal problem.

Order Execution Time of All Eigenvalues		
n	RSG	LGT
100	2.25	0.31
200	19.02	1.21
300	70.29	2.79
400	194.71	4.87

Table 1: Execution time (seconds) of computed eigenvalues.

Experiment 2. Our algorithm LGT and the routine BIS, the bisection method are applied to equation (1), where A and M are symmetric random matrices with $m_{i,i} = 2(m_{i,i-1} + m_{i,i+1})$ so that M is positive definite. For each n, we ran 50 problems to find the average time. Both the LGT and BIS can fully take advantage of the symmetric tridiagonal forms of A and M. Since the bisection converges only linearly, Table 2 shows that our algorithm leads in speed by a considerable margin.

Order Execution Time of All Eigenvalues		
n	BIS	LGT
100	0.76	0.34
200	3.09	1.26
300	6.47	2.99
400	11.46	5.04
1000	77.19	27.24

Table 2: Execution time (seconds) of computed eigenvalues.

References

1. R. F. Boisvert, "Families of Higher Order Accurate Discretizations of Some Elliptic Problems," *SIAM J. Sci. Stat. Comput.*, 2 (1981), 268–284.

2. J. J. Dongarra and D. C. Sorensen, "A Fully Parallel Algorithm for the Symmetric Eigenvalue Problem," *SIAM J. Sci. Stat. Comput.*, 8 (1987), 139–154.

3. B. S. Garbow et. al., *Matrix Eigensystem Routines – EISPACK Guide Extension*, Springer–Verlag, New York, 1977.

4. K. Li, "Durand–Kerner Root–Finding Method for the Generalized Tridiagonal Eigenproblem," *Missouri Journal of Mathematical Sciences*, (to appear).

5. K. Li and T. Y. Li, "An Algorithm for Symmetric Tridiagonal Eigen–Problems – Divide and Conquer with Homotopy Continuation," *SIAM J. Sci. Comput.*, 14 (1993), 735–751.

6. K. Li, T. Y. Li, and Z. Zeng, "An Algorithm for the Generalized Symmetric Tridiagonal Eigenvalue Problem," *Numerical Algorithms*, 8 (1994), 269–291.

7. S–S. Lo, B. Phillipe, and A. Sameh, "A Multiprocessor Algorithm for the Symmetric Tridiagonal Eigenvalue Problems," *SIAM J. Sci. Comput.*, 8 (1987), 155–165.

8. B. Phillipe and B. Vital, "Parallel Implementations for Solving Generalized Eigenvalue Problems with Symmetric Sparse Matrices," *Applied Numerical Mathematics*, 12 (1993), 391–402.

9. G. Strang and G. Fix, *An Analysis of the Finite Element Method*, Prentice–Hall, Inc., Englewood Cliffs, N.J., 1973.

10. W. Weaver, Jr. and P. R. Johnson, *Finite Elements for Structural Analysis*, Prentice–Hall, New Jersey, 1984.

11. J. H. Wilkinson, *The Algebraic Eigenvalue Problem*, Oxford University Press, Oxford, 1965.

Kuiyuan Li
Department of Mathematics and Statistics
University of West Florida
Pensacola, FL 32514

Matoteng Ncube
Department of Mathematics and Statistics
University of West Florida
Pensacola, FL 32514