International Journal of Pure and Applied Mathematics

Volume 55 No. 3 2009, 301-310

TRIDIAGONAL MATRICES AND THE COMPUTATION OF GAUSSIAN QUADRATURES

Qassem Al-Hassan

Department of Mathematics
University of Sharjah
P.O. Box 27272, Sharjah, UNITED ARAB EMIRATES
e-mail: qassem@sharjah.ac.ae

Abstract: There is a close relation between tridiagonal matrices and orthogonal polynomials, this relation is founded upon a second-order homogeneous linear difference equation that serves as a recursion relation used for generating these polynomials. This relation among tridiagonal matrices, orthogonal polynomials, and second-order homogeneous linear difference equations is employed in order to compute the nodes and weights of Gaussian quadratures.

AMS Subject Classification: 65D32, 65Q05, 39A05, 65F15
Key Words: tridiagonal matrices, Guassian quadrature, nodes and weights, recurrence relations

1. Introduction

In [1], it was shown that the set of orthogonal polynomials $\{p_n(x)\}_{n\geq 0}$ on an interval $[a,b]$ with respect to a nonnegative weight function $w(t)$, has a close relation with tridiagonal matrices, in the sense that $p_n(x) = \det(C)$ where C is the tridiagonal $n\times n$ matrix given by:

$$\begin{bmatrix} \frac{x-a_1}{b_1} & 1 & 0 & \cdots & 0 \\ \frac{c_2}{b_2} & \frac{x-a_2}{b_2} & 1 & 0\ \cdots & 0 \\ 0 & \frac{c_3}{b_3} & \frac{x-a_3}{b_3} & 1\ \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots & 1 \\ 0 & 0 & \cdots & \frac{c_n}{b_n} & \frac{x-a_n}{b_n} \end{bmatrix}.$$

This result is a consequence of the fact that such orthogonal polynomials satisfy

Received: May 5, 2008

the second-order homogeneous linear difference equation:

$$p_n(x) = \frac{(x-a_n)}{b_n}\, p_{n-1}(x) - \frac{c_n}{b_n}\, p_{n-2}(x), \quad n \geq 2 \tag{1}$$

with initial conditions: $p_0(x) = 1$, $p_1(x) = \frac{x-a_1}{b_1}$, $b_n \neq 0$, and $c_n \neq 0$ for all n.

In this paper, this result is employed in order to calculate the nodes $\{x_i\}_{i=1}^n$ and the weights $\{w_i\}_{i=1}^n$ of Gaussian quadratures, which are usually used to approximate $\int_a^b f(x)dx$ by $\sum_{i=1}^n w_i f(x_i)$, so that this approximation is exact when $f(x)$ is a polynomial of degree less than $2n$.

It is known that the zeros of $p_j(x), j = 1, ..., n$ are all real, distinct, and lie in the interval (a, b), so that $a < x_1 < x_2 < ... < x_j < b$, and the nodes of the Gaussian quadrature are exactly the zeros of $p_n(x)$, while the weights $\{w_i\}_{i=1}^n$ are structured in a way to force the approximation to be exact for all polynomials of degree less than $2n$.

The procedure we present here for computing the nodes and the weights of Gaussian quadratures is robust, with low cost in terms of complexity when compared with existing other procedures. This issue will be discussed in the last section of this work. In Section 2, we mention methods of computing the nodes $\{x_i\}_{i=1}^n$, and in Section 3 we present the method we use in order to determine the weights $\{w_i\}_{i=1}^n$. In Section 4, we introduce numerical implementation of the method, and some concluding remarks.

2. Determination of the Nodes

The difference equation in (1) can be rewritten as:

$$\frac{x}{b_n}\, p_{n-1}(x) = \frac{a_n}{b_n}\, p_{n-1}(x) + \frac{c_n}{b_n}\, p_{n-2}(x) + p_n(x) \tag{2}$$

and this in matrix form yields:

$$x \begin{bmatrix} \frac{1}{b_1} & 0 & & & \\ 0 & \frac{1}{b_2} & 0 & & \\ & 0 & . & 0 & \\ & & 0 & . & 0 \\ & & & 0 & \frac{1}{b_n} \end{bmatrix} \begin{bmatrix} p_0(x) \\ p_1(x) \\ . \\ . \\ p_{n-1}(x) \end{bmatrix}$$

$$= \begin{bmatrix} \frac{a_1}{b_1} & 1 & 0 & . & 0 \\ \frac{c_2}{b_2} & \frac{a_2}{b_2} & 1 & . & \\ 0 & . & . & . & 0 \\ & . & . & . & 1 \\ & & 0 & \frac{c_n}{b_n} & \frac{a_n}{b_n} \end{bmatrix} \begin{bmatrix} p_0(x) \\ p_1(x) \\ . \\ . \\ p_{n-1}(x) \end{bmatrix} + \begin{bmatrix} 0 \\ . \\ . \\ . \\ p_n(x) \end{bmatrix}, \quad (3)$$

or

$$x\ D\ \mathbf{p} = A\ \mathbf{p} + p_n(x)\ \mathbf{e}_n,$$

where $D = \operatorname{diag}\left(\frac{1}{b_1}, ..., \frac{1}{b_n}\right)$, A is a tridiagonal matrix, $\mathbf{p} = [p_0(x), ..., p_{n-1}(x)]^T$, and $\mathbf{e}_n = [0, ..., 0, 1]^T$.

When $x = x_j$, where x_j is a zero of $p_n(x)$, this last equation reduces to

$$x_j\ D\ \mathbf{p}(x_j) = A\ \mathbf{p}(x_j). \quad (4)$$

So, $x_j, j = 1, ..., n$, is the solution of the generalized eigenvalue problem:

$$\lambda\ D\ \mathbf{u} = A\ \mathbf{u}$$

which is equivalent to the eigenvalue problem:

$$(D^{-1}A)\mathbf{u} = \lambda\mathbf{u},$$

where

$$D^{-1}A = T = \begin{bmatrix} a_1 & b_1 & 0 & & \\ c_2 & a_2 & b_2 & . & \\ 0 & . & . & . & 0 \\ & . & . & . & b_{n-1} \\ & & 0 & c_n & a_n \end{bmatrix}.$$

So, the nodes $\{x_i\}_{i=1}^n$ — which are the zeros of $p_n(x)$ — are exactly the eigenvalues of the matrix T, and since T is an upper Hessenberg matrix, the most popular choice for computing the eigenvalues of T — hence the nodes of the quadrature — is the QR algorithm, or the shifted QR algorithm. We used *Maple 9.5* in order to generate the sequence of orthogonal polynomials up to a prescribed degree n, and compute the zeros of $p_n(x)$. The values obtained were exactly equal to the analytic values presented in the *Wolfram Mathworld Website*.

3. Determination of the Weights

We first need the following two theorems.

Theorem 1. *Let $\{p_k(x)\}_{k=0}^n$ be a set of orthogonal polynomials on an interval $[a, b]$ with respect to the weight function $w(x)$, and let $\{x_i\}_{i=1}^n$ be the* *zeroes of $p_n(x)$, then the matrix B given by:*

$$B = \begin{bmatrix} p_0(x_1) & p_0(x_2) & . & . & p_0(x_n) \\ p_1(x_1) & p_1(x_2) & . & . & p_1(x_n) \\ . & . & . & . & . \\ . & . & . & . & . \\ p_{n-1}(x_1) & p_{n-1}(x_2) & . & . & p_{n-1}(x_n) \end{bmatrix}$$

is nonsingular.

Proof. Since the columns of the matrix B are the eigenvectors of the matrix T, which has distinct eigenvalues, the columns of B are linearly independent. □

Theorem 2. *Let $x_1, x_2, ..., x_n$ be as in Theorem 1, and define μ by*

$$\mu = \int_a^b w(x)\ dx.$$

(a) If $w_1, w_2, ..., w_n$ are solutions of the system:

$$\sum_{i=1}^{n} p_k(x_i)\ w_i = \begin{cases} \mu & \text{for} \quad k = 0, \\ 0 & \text{for} \quad k = 1, 2, ..., n-1, \end{cases} \tag{5}$$

then $w_i > 0$ for $i = 1, 2, ..., n$, and

$$\int_a^b q(x)\ w(x)\ dx = \sum_{i=1}^{n} w_i\ q(x_i) \tag{6}$$

holds for all polynomials with degree less than $2n$.

(b) Conversely, if the numbers $w_i, i = 1, ..., n$ are chosen such that (6) holds for all polynomials with degree less than $2n$, then $x_i, i = 1, ..., n$ are the zeroes of $p_n(x)$ and the weights $w_i, i = 1, ..., n$ are the solutions of the system (5).

Proof. See [4], p. 153. □

Since the matrix T is a nonsingular matrix, with distinct eigenvalues, it is known that there exists a diagonal similarity transformation D_1 which symmetrizes the matrix T, i.e. there exists a nonsingular diagonal matrix D_1 such that the matrix $J_n = D_1 T D_1^{-1}$ is symmetric (see [2, p. 150] and [3, p. 209]). In the next theorem, we give an explicit form for the diagonal matrix D_1 which is used to symmetrize the matrix T, and, also, give an explicit form for the symmetrized matrix J_n.

Theorem 3. *Let D_1 be the $n \times n$ diagonal matrix given by:* $D_1 =$

diag$(\gamma_1, ..., \gamma_n)$*, where*

$$\gamma_n = 1 \quad \textit{and} \quad \gamma_i = \sqrt{\frac{c_{i+1}c_{i+2}...c_n}{b_i b_{i+1}...b_{n-1}}}, \quad i = 1, ..., n-1,$$

then the matrix $J_n = D_1 T D_1^{-1}$ *is a symmetric matrix of the form:*

$$J_n = \begin{bmatrix} a_1 & \sqrt{b_1c_2} & 0 & & \\ \sqrt{b_1c_2} & a_2 & \sqrt{b_2c_3} & . & \\ 0 & \sqrt{b_2c_3} & . & . & 0 \\ & . & . & . & \sqrt{b_{n-1}c_n} \\ & & 0 & \sqrt{b_{n-1}c_n} & a_n \end{bmatrix}.$$

Proof. The entries of $D_1 T D_1^{-1}$ are as follows:

(i) The diagonal entries are just a_j for $j = 1, ..., n$.

(ii) The upper diagonal entries for $j = 1, ..., n-2$ are of the form:

$$b_j \frac{\gamma_j}{\gamma_{j+1}} = b_j \sqrt{\frac{c_{j+1}c_{j+2}\cdots c_n}{b_j b_{j+1}\cdots b_{n-1}}} \sqrt{\frac{b_{j+1}\cdots b_{n-1}}{c_{j+2}\cdots c_n}} = b_j \sqrt{\frac{c_{j+1}}{b_j}} = \sqrt{b_j\ c_{j+1}}.$$

(iii) The lower diagonal entries for $j = 2, ..., n-1$ are of the form:

$$c_j \frac{\gamma_j}{\gamma_{j-1}} = c_j \sqrt{\frac{c_{j+1}c_{j+2}\cdots c_n}{b_j b_{j+1}\cdots b_{n-1}}} \sqrt{\frac{b_{j-1}b_j\cdots b_{n-1}}{c_j c_{j+1}\cdots c_n}} = c_j \sqrt{\frac{b_{j-1}}{c_j}} = \sqrt{b_{j-1}c_j}.$$

(iv) The upper diagonal entry in the position $(n-1, n)$ has the form:

$$b_{n-1}\gamma_{n-1} = b_{n-1}\sqrt{\frac{c_n}{b_{n-1}}} = \sqrt{b_{n-1}c_n}.$$

(v) Finally, the lower diagonal entry in the position $(n, n-1)$ has the form:

$$\frac{c_n}{\gamma_{n-1}} = c_n \sqrt{\frac{b_{n-1}}{c_n}} = \sqrt{b_{n-1}c_n}.$$ □

Remark 4. Since T and J_n are similar, they have the same eigenvalues. Thus the nodes $\{x_i\}_{i=1}^n$, which are the eigenvalues of T, are also the eigenvalues of the symmetric tridiagonal matrix J_n.

The symmetric matrix J_n is closely related with the theory of orthogonal polynomials. The characteristic polynomials $p_j(x)$ of the $j \times j$ principal submatrices of J_n are exactly the orthogonal polynomials defined by the recursion relation:

$$\begin{aligned} p_{-1}(x) &= 0, \\ p_0(x) &= 1, \end{aligned}$$

$$p_j(x) = (x-a_j)p_{j-1}(x) - b_{j-1}c_jp_{j-2}(x) \quad \text{for} \quad 1 \le j \le n. \tag{7}$$

As a consequence, $p_n(x)$ is the characteristic polynomial of the matrix J_n, and the eigenvalues of J_n are the zeroes of $p_n(x)$.

Notice that the recursion relation in (7) can be rewritten as:

$$x\ p_{j-1}(x) = a_j\ p_{j-1}(x) + b_{j-1}\ c_j\ p_{j-2}(x) + p_j(x), \quad 1 \le j \le n \tag{8}$$

Theorem 5. *Let* $\mathbf{v}^{(i)} = \left[v_1^{(i)}, v_2^{(i)}, ..., v_n^{(i)}\right]^T$ *be a normalized eigenvector of the matrix* J_n *associated with the eigenvalue* x_i*, that is* $J_n\mathbf{v}^{(i)} = x_i\mathbf{v}^{(i)}$. *Then* $w_i = \mu\left(v_1^{(i)}\right)^2$, $i = 1, ..., n$.

Proof. First, we have to notice that $b_{i-1}c_i \neq 0, i = 2, ..., n$. Second, we define the sequence $\{\alpha_j\}_{j=0}^{n-1}$ by:

$$\alpha_j = \begin{cases} 1 & \text{for} \quad j = 0, \\ \frac{1}{\sqrt{\prod_{i=1}^{j} b_ic_{i+1}}} & \text{for} \quad j = 1, ..., n-1, \end{cases}$$

and consider the vector $\mathbf{u}^{(i)} = [\alpha_0p_0(x_i), \alpha_1p_1(x_i), ..., \alpha_{n-1}p_{n-1}(x_i)]^T$. We shall show that $\mathbf{u}^{(i)}$ is an eigenvector of J_n associated with the eigenvalue x_i. Third, the sequence $\{\alpha_j\}_{j=0}^{n-1}$ satisfies the relation $\alpha_{j-1} = \sqrt{b_jc_{j+1}}\ \alpha_j$.

Now, the product $J_n\mathbf{u}^{(i)}$ produces expressions in one of the following forms:

(i) For $j = 1$,

$$\begin{aligned} a_1\alpha_0p_0(x_i) + \sqrt{b_1c_2}\ \alpha_1p_1(x_i) &= a_1p_0(x_i) + p_1(x_i) = x_ip_0(x_i) \\ &\quad \text{by (8) since } p_{-1}(x_i) = 0 \\ &= x_i\ (\alpha_0p_0(x_i)). \end{aligned}$$

(ii) For $j = 2, ..., n-1$,

$$\begin{aligned} &\sqrt{b_{j-1}c_j}\ \alpha_{j-2}p_{j-2}(x_i) + a_j\alpha_{j-1}p_{j-1}(x_i) + \sqrt{b_jc_{j+1}}\ \alpha_jp_j(x_i) \\ &\qquad = (b_{j-1}c_j\ \alpha_{j-1})p_{j-2}(x_i) + a_j\alpha_{j-1}p_{j-1}(x_i) + \alpha_{j-1}p_j(x_i) \\ &\qquad = \alpha_{j-1}\ [b_{j-1}c_jp_{j-2}(x_i) + a_jp_{j-1}(x_i) + p_j(x_i)] \\ &\qquad = \alpha_{j-1}\ x_ip_{j-1}(x_i) = x_i\ (\alpha_{j-1}p_{j-1}(x_i)) \qquad \text{by (8)}. \end{aligned}$$

(iii) Finally, for $j = n$,

$$\begin{aligned} &\sqrt{b_{n-1}c_n}\alpha_{n-2}p_{n-2}(x_i) + a_n\alpha_{n-1}p_{n-1}(x_i) \\ &\qquad = b_{n-1}c_n\alpha_{n-1}p_{n-2}(x_i) + a_n\alpha_{n-1}p_{n-1}(x_i) \\ &\qquad = \alpha_{n-1}\ [b_{n-1}c_np_{n-2}(x_i) + a_np_{n-1}(x_i)] \\ &\qquad = \alpha_{n-1}\ [x_ip_{n-1}(x_i) - p_n(x_i)] \qquad \text{by (8) since } p_n(x_i) = 0 \end{aligned}$$

$$= x_i\ (\alpha_{n-1} p_{n-1}(x_i)).$$

The system in (5) is equivalent to:

$$\left[\mathbf{u}^{(1)}\ \mathbf{u}^{(2)}\ \cdots \mathbf{u}^{(n)}\right] \mathbf{w} = \mu \mathbf{e}_1, \tag{9}$$

where $\mathbf{w} = [w_1, w_2, ..., w_n]^T$, and $\mathbf{e}_1 = [1, 0, ..., 0]^T$.

It is known that the eigenvectors of real symmetric matrices corresponding to distinct eigenvalues are orthogonal. Thus, multiplying (9) by $\left(\mathbf{u}^{(i)}\right)^T$ on the left yields:

$$\left[\left(\mathbf{u}^{(i)}\right)^T \left(\mathbf{u}^{(i)}\right)\right] w_i = \mu\ u_1^{(i)}$$

and since α_0 and $p_0(x)$ are both equal to 1, $u_1^{(i)}$ must also be equal to 1. Therefore, we have:

$$\left[\left(\mathbf{u}^{(i)}\right)^T\ \left(\mathbf{u}^{(i)}\right)\right]\ w_i = \mu. \tag{10}$$

Using the fact that $u_1^{(i)} = 1$ again, we find that $v_1^{(i)}\ \mathbf{u}^{(i)} = \mathbf{v}^{(i)}$, and when (10) is multiplied by $\left(v_1^{(i)}\right)^2$ we obtain:

$$\left[\left(\mathbf{v}^{(i)}\right)^T\ \left(\mathbf{v}^{(i)}\right)\right] w_i = \mu\ \left(v_1^{(i)}\right)^2$$

and this completes the proof. □

Remark 6. It is worth to mention here that the proof above is similar to that of Theorem 3.6.21 in [4], p. 157.

Theorem 7. *An eigenvector of the matrix J_n that corresponds to the eigenvalue x_i is $D_1\mathbf{p}(x_i)$, where $\mathbf{p}(x_i) = [p_0(x_i), p_1(x_i), ..., p_{n-1}(x_i)]^T$.*

Proof. Using equation (4), we have

$$x_i\ \mathbf{p}(x_i) = T\ \mathbf{p}(x_i) = D_1^{-1} J_n D_1\ \mathbf{p}(x_i),$$

and this implies that

$$J_n \left[D_1 \mathbf{p}(x_i)\right] = x_i\ \left[D_1 \mathbf{p}(x_i)\right].$$ □

4. Numerical Implementation

In Section 2, it is shown that the eigenvalues of the tridiagonal matrix T are the nodes of the Gaussian quadrature, and the corresponding eigenvectors of T are the columns of the matrix B in Theorem 1. This means that for each eigenvalue(node) x_i, the corresponding eigenvector is $[p_0(x_i), p_1(x_i), ..., p_{n-1}(x_i)]^T$.

This suggests that the set of orthogonal polynomials $\{p_j(x)\}_{j=0}^n$ must first be generated using the recursion relation (3.3). Then the zeros of $p_n(x)$ are computed. As mentioned before, we used *Maple 9.5* to generate the set $\{p_j(x)\}_{j=0}^n$, and to compute the zeros of $p_n(x)$. Having computed the nodes $\{x_i\}_{i=1}^n$, we now introduce the method for computing the weights $\{w_i\}_{i=1}^n$.

Theorem 5 gives a formula for the i-th eigenvector of the matrix J_n, $v^{(i)}$, such an eigenvector has the form: $v^{(i)} = [\gamma_1 p_0(x_i), \gamma_2 p_1(x_i), ..., \gamma_n p_{n-1}(x_i)]^T$, and this yields the following:

$$\mathbf{v}^{(i)} = \left[\sqrt{\frac{c_2c_3\cdots c_n}{b_1b_2\cdots b_{n-1}}}\, p_0(x_i),\ \sqrt{\frac{c_3\cdots c_n}{b_2\cdots b_{n-1}}}\, p_1(x_i),\ ..., \right. \\ \left. \sqrt{\frac{c_n}{b_{n-1}}}\, p_{n-2}(x_i),\ p_{n-1}(x_i)\right]^T. \quad (11)$$

The components of the eigenvector $\mathbf{v}^{(i)}$ are computed as follows:

(i) Compute the sequence $\{\gamma_i\}_{i=1}^n$ using the recursion:

$$\begin{aligned} \gamma_n &= 1, \\ \gamma_i &= \gamma_{i+1}\cdot\sqrt{\frac{c_{i+1}}{b_i}}, \qquad i = n-1, ..., 1. \end{aligned} \quad (12)$$

(ii) The recursion relation (7) is used to compute the components of the vector $\mathbf{p}(x_i)$. Notice that the computation of each components costs three flops only, where a flop is equal to one multiplication, one division, or one exponentiation.

(iii) Set $v_j^{(i)} := \gamma_j p_{j-1}(x_i), j = 1, ..., n$.

(iv) Finally, the eigenvectors must be normalized, and the weights are computed using: $w_i = \mu \left(v_1^{(i)}\right)^2, i = 1, ..., n$, where $\mu = \int_a^b w(x)dx$.

To summarize, we compute the nodes and the weights of the Gaussian quadrature by doing the following steps:

1. Use the recursion relation in (1) to generate the sequences: $\{a_j\}_{j=1}^n$, $\{b_j\}_{j=1}^n$, and $\{c_j\}_{j=2}^n$.

2. Use the recursion relation in (7) to generate the sequence of orthogonal polynomials $\{p_j(x)\}_{j=0}^n$.

3. Compute the zeros of $p_n(x)$ using any convenient method (such as QR algorithm or any of its versions, Newton's method, or an existing software package (*Maple* or *Matlab* package), these zeros are the nodes of the quadrature.

4. The weights are computed as follows:

(i) Use (12) to compute the eigenvectors $v^{(i)}$ of the matrix $J_n\mathbf{v}^{(i)},\ i = 1, ..., n$.

(ii) Normalize $\mathbf{v}^{(i)},\ 1 \le i \le n$.

(iii) Set $w_i = \mu\ \left(v_1^{(i)}\right)^2,\ i = 1, ..., n$.

The complexity of this algorithm lies mainly in step 4 above in which we calculate the weights $\{w_i\}_{i=1}^n$. First, the calculation of the vector $[p_0(x_i), p_1(x_i), ..., p_{n-1}(x_i)]^T$ is achieved by performing 3 flops for each component. This part has a total of $3n$ flops. Then the computation of $v^{(i)}$ using (4.3), and counting the flops for each component yields the sequence, $(2n-1)$ flops for first component, $(2n-3)$ for second component, down to 1 flop for the last component, and this sums up to $2n^2-n$ flops. This gives a total of $2n^3-n^2$ for all the n eigenvectors. The normalization of each eigenvector costs $n+2$ flops, which totals to $2n^2+n$. Finally, the last step costs 2 flops for each weight, which totals to $2n$ flops. Therefore, the total count of flops is: $(3n) + (2n^3 - n^2) + (2n^2 + n) + (2n) = 2n^3 + n^2 + 6n$, and for moderate values of n (e.g. $n \ge 6$), this sum is bounded by $2n^3 + 2n^2$.

As an alternative, we consider the most popular choice for computing eigenvalues and eigenvectors of real symmetric tridiagonal matrices, that is the QR algorithm or any of its versions, such as shifted QR algorithm. This algorithm is outlined in the following steps:

Algorithm 8. 1. Set $J_0 := J_n$.

2. For $i = 0, 1, 2, ...$, until convergence, do

2.1 Find the QR factorization of J_i, $J_i = Q_i R_i$, where Q_i is an orthogonal matrix, and R_i is upper triangular.

2.2 Set $J_{i+1} := R_i Q_i$

For sufficiently large i, the matrix

$$J_i = (Q_0 Q_1 ... Q_{i-1})^T J_0 (Q_0 Q_1 ... Q_{i-1})$$

will converge to a diagonal matrix which contains the eigenvalues on the diagonal, and the product $Q_0 Q_1 ... Q_{i-1}$ converges to an orthogonal matrix Q, whose columns are the orthonormal eigenvectors of J_n.

We implemented the QR algorithm from *Maple 9.5* software for Gauss-Legendre, Gauss-Laguerre, Gauss-Hermite, and Gauss-Chebyshev quadratures, in order to compute the eigenvalues and the eigenvectors of the symmetric matrix J_n, the nodes and the weights where exactly equal to their analytic values obtained from *Wolfram Mathworld Website*.

The complexity of this algorithm is of order $3n^3$ due to the tridiagonality of J_n, see [3], p. 473. Moreover, the workload can significantly be reduced by observing the fact that we just need the first component of each eigenvector, so we do not need to do complete multiplication in the computation of the product $Q_0Q_1...Q_{i-1}$. Instead, we just need to multiply the first row of Q_0 by Q_1, and then multiply the result by Q_2, and so on.

Therefore, we think that our algorithm is competitive with the QR algorithm, and as a matter of fact, it is less costly for relatively large values of n, due to the complexity of both algorithms.

Acknowledgments

This last observation of calculating the first component of each eigenvector was brought to my attention by private communication with Professor M. Hajji, from the Department of Mathematical Sciences, United Arab Emirates University. The author wishes also to thank Professor R. Abu Saris from the Department of Mathematics, University of Sharjah, for his valuable advice and constant support during the preparation of this work.

References

[1] Q. Al-Hassan, An algorithm for computing inverses of tridiagonal matrices with applications, *Soochow Journal of Mathematics*, **31**, No. 3 (2005), 449-466.

[2] R. Horn, C. Johnson, *Matrix Analysis*, Cambridge University Press (1990).

[3] W. Press, S. Teukolsky, W. Vetterling, B. Flannery, *Numerical Recipes in Fortran 77*, Cambridge University Press (1992).

[4] J. Stoer, R. Bulirsch, *Introduction to Numerical Analysis*, Springer-Verlag, NY (1993).