

Survey of power, QR, and oepomo's iterative methods for solution of largest eigenvalue of essentially positive matrices.

Tedja Santanoe Oepomo*

Science, Engineering and Mathematics Division, West LA College, Riverside Community College and LA Harbor College, California, USA

(Received December 12 2007, Revised April 19 2008, Accepted September 6 2008)

Abstract. Many of the popular methods for the solution of largest eigenvalue of essentially positive irreducible matrices are surveyed with the hope of finding an efficient method suitable for electromagnetic engineering, radiation problems, system identification problems, and solid mechanics. Eigenvalue computations are both fundamental and ubiquitous in computational science and its fast application areas. Some comparisons between several known algorithms, i.e. Power and QR methods, and earlier theory of Oepomo iterative techniques for solving largest eigenvalue of nonnegative irreducible matrices are presented since there is a continuing demand for new algorithm and library software that efficiently utilize and adapt to new applications.

Keywords: Collatz's theorem, Perron-Frobernius' theorem, and eigenvalue

1 Introduction

Without doubt the power and QR iterative methods have proved most popular among applied mathematician-that is if popularity is measured by frequency of use. In our discussions of the various methods and the results of comparisons with Oepomo methods, only references that are directly relevant are noted. No attempt has been made to cite the earliest sources.

2 Eigenvalue computation

The second fundamental problem in linear algebra is to compute the eigenvalues and eigenvectors of a square matrix. In other words, to diagonalize a square matrix. In theory we know what to do. We have to compute the characteristic polynomial $P(\lambda)$ find the roots λ_i , then for each i we have the equation $(A - \lambda_i)x = 0$ for the eigenvectors. However, this procedure is not satisfactory.

Newtown's method give a very fast way to compute roots of polynomial $P(\lambda)$, it can also easily be extended to complex roots. But remember, that Newton's method usually works very well once you are reasonably "close" to the root, but it may diverge if you start far away. Still you need *ad hoc* method to get "close" to the root.

To reduce the chance element in this "hit or miss" strategy, another numerical computations were invented, i.e. power, QR iterative, and Oepomo's methods. Moreover, we also have to care about the stability of the method, as the eigenvalue problem can be quite ill conditioned, so we do not want to make it much worse by an unstable algorithm.

* Corresponding author. *E-mail address:* oepomots@elac.edu; oepomot@wlac.edu.

Example 1. Consider the $n \times n$ matrix

$$A = \begin{bmatrix} 0 & & & & & & & \varepsilon \\ 1 & 0 & & & & & & \\ & 1 & 0 & & & & & \\ & & 1 & 0 & & & & \\ & & & 1 & 0 & & & \\ & & & & \ddots & \ddots & & \\ & & & & & \ddots & \ddots & \\ & & & & & & \ddots & \\ & & & & & & & 1 & 0 \end{bmatrix} \quad (1)$$

where ε is a tiny number with all entries are zero. The characteristic polynomial of A is $P(\lambda) = \lambda^n - \varepsilon$. Clearly all eigenvalues are 0 if $\varepsilon = 0$. However for $\varepsilon \neq 0$ there are n complex eigenvalues, the n -th roots of unity. Even if we focus on the real eigenvalue $\varepsilon^{1/n}$, it is clear that the problem is very ill conditioned. Let $n = 40$ and choose $\varepsilon = 10^{-40}$ which is an extremely tiny relative error of order $\frac{10^{-40}}{1} = 10^{-40}$.

However, one of the eigenvalue is $\lambda = 10^{-1} = 0.1$ which is at a distance of 10^{-1} from the “unperturbed” eigenvalue of 0. Hence the change in the eigenvalues is equal to the change in the perturbation parameter ε multiplied by 10^{+39} . There is another disquieting aspect of this phenomenon. The number $\varepsilon = 10^{-40}$ is automatically replaced by 0 in the computer, and this rounding introduces an error of order 10^{-1} in the result.

Fortunately the eigenvalue problem is not so ill conditioned for “most” matrices, this example were particularly nasty one. However, it shows that we should aim at stable algorithms, which at least do not make “bad things much worse”, since the problem itself can be quite “bad”.

3 Power method

This method is extremely simple. Let A be a square matrix. Pick any vector x_0 and start successively multiplying it with A . We can claim, that unless you are extremely unlucky with the choice of x_0 , we can easily find the eigenvalue with the largest modulus.

Theorem 1. *Suppose that the square matrix A has one eigenvalue λ_1 that is greater in absolute value than all other eigenvalues, and let $x_{n+1} = Ax_n$. Then from randomly chosen nonzero vectors x_0 and w we have*

$$\lim_{n \rightarrow \infty} \frac{w^t \bullet x_{n+1}}{w^t \bullet x_n} = \lambda_1 \text{ “almost surely”}.$$

Moreover, the sequence of normalized vectors $\tilde{x}_n := \frac{x_n}{\|x_n\|}$ converges to the eigenvector of λ_1 .

Remark 1. The probabilistic expression “almost surely” in the theorem can be made mathematically precise, i.e. it means “for almost all” vector, with the exception of vectors lying in a smaller subspace, but we do not want to go into this detail. For the present purpose of the manuscript you should take it literally.

Proof. We give a proof only for a case where is symmetric $k \times k$ matrix i.e. $A = VDV^t$, and λ_1 is a simple eigenvalue. The proof for non-symmetric diagonalizable matrices is slightly harder, and the case of a general matrix A is moderately harder. We can use the spectral theorem, i.e.

$$A = \sum_{i=1}^k \lambda_i v_i v_i^t, \text{ clearly } A^n = \sum_{i=1}^k \lambda_i^n v_i v_i^t \text{ and } w^t \bullet x_n = w^t \bullet A^n x_0 = \sum_{i=1}^k \lambda_i^n (w^t \bullet v_i)(v_i^t \bullet x_0).$$

We compute

$$\lim_{n \rightarrow \infty} \frac{w^t \bullet x_n}{\lambda_1^n} = \lim_{n \rightarrow \infty} \sum_{i=1}^k \left(\frac{\lambda_i}{\lambda_1} \right)^n (w^t \bullet v_i)(v_i^t \bullet x_0) = (w^t \bullet v_1)(v_1^t \bullet x_0),$$

since $|\lambda_1| > |\lambda_i|$ for $i > 1$. Assume that x_0 and w are not orthogonal to v_1 (this is the case that one has to exclude by choosing the vectors “randomly”). Then

$$\lim_{n \rightarrow \infty} \frac{w^t \bullet x_{n+1}}{w^t \bullet x_n} = \lambda_1 \frac{\lim_{n \rightarrow \infty} \frac{w^t \bullet x_{n+1}}{\lambda_1^{n+1}}}{\lim_{n \rightarrow \infty} \frac{w^t \bullet x_n}{\lambda_1^n}} = \lambda_1 \frac{(w^t \bullet v_1)(v_1^t \bullet x_0)}{(w^t \bullet v_1)(v_1^t \bullet x_0)} = \lambda_1$$

which completes the proof of the eigenvalue convergence. To see the convergence of the normalized eigenvectors, notice that

$$x_n = A^n x_0 = \sum_{i=1}^k \lambda_i v_i (v_i^t \bullet x_0) \text{ here } \|x_n\| = \sqrt{\sum_{i=1}^k |\lambda_i|^{2n} (v_i^t \bullet x_0)^2},$$

so

$$\begin{aligned} \tilde{x}_n &= \frac{x_n}{\|x_n\|} = \frac{\sum_{i=1}^k \lambda_i^n (v_i^t \bullet x_0)^2}{\sqrt{\sum_{i=1}^k |\lambda_i|^{2n} (v_i^t \bullet x_0)^2}} \\ &= \left(\frac{\lambda_1 (v_1^t \bullet x_0)}{|\lambda_1 (v_1^t \bullet x_0)|} \right)^n \frac{v_1 + \sum_{i=2}^k \left(\frac{\lambda_i}{\lambda_1} \right)^n \frac{v_i^t \bullet x_0}{v_1^t \bullet x_0} v_i}{\sqrt{1 + \sum_{i=2}^k \left| \frac{\lambda_i}{\lambda_1} \right|^{2n} \frac{(v_i^t \bullet x_0)^2}{(v_1^t \bullet x_0)^2}}} \end{aligned}$$

and if $|\lambda_1| > |\lambda_i|$ for all other i , then this clearly converges v_1 .

This method gave only the eigenvalue of largest modulus. With a little trick, one can get all other eigenvalues,

Theorem 2. Suppose that the square matrix A has an eigenvalue λ_p , which is closer to p than all other eigenvalues. Run the $x_{n+1} = (A - pI)^{-1} x_n$, iteration with some initial vector x_0 . If the vectors x_0 and w are chosen randomly, then

$$\lim_{n \rightarrow \infty} \frac{w^t \bullet x_{n+1}}{w^t \bullet x_n} = \frac{1}{\lambda_p - p} \text{ “almost surely”}.$$

Hence λ_p is obtained as

$$\lambda_p = \lim_{n \rightarrow \infty} \frac{w^t \bullet x_{n+1}}{w^t \bullet x_n} + p \text{ (if } \lambda_p \neq p \text{)}.$$

Moreover, again, the normalized vectors $\tilde{x}_n := \frac{x_n}{\|x_n\|}$ converge to the eigenvector belonging to λ_p .

Proof. Simply that the eigenvalues of $B := (A - pI)^{-1}$ are $(\lambda_1 - p)^{-1}$, $(\lambda_2 - p)^{-1}$, ..., where λ_i are the eigenvalues of A . Hint: this statement does not require A being symmetric, so spectral theorem cannot be used. From the condition it follows that $(\lambda_1 - p)^{-1}$ is the largest in modulus among them, so we can apply Theorem 1. The statement on the eigenvector is straightforward from Theorem 1.

Remark 2. The power method looks very simple and elegant. However, notice it is really powerful only for the eigenvalue largest in modulus. Applying Theorem 2 already requires inverting a matrix, and more importantly it requires knowing a point p “near” the eigenvalue. Hence we run into similar difficulties when we applied Newton’s iteration for finding the roots of the characteristics polynomial. In fact there is essentially a Newton’s iteration behind Theorem 2.

For iterative Power algorithm let A be an (n, n) matrix. Let $\lambda_1, \lambda_2, \dots, \lambda_n$ be the eigenvalues of A and let

$$|\lambda_1| = |\lambda_2| = \dots = |\lambda_k| > |\lambda_{k+1}| \geq \dots \geq |\lambda_n| \tag{2}$$

and $\lambda_1 = \lambda_2 = \dots = \lambda_k$ for $1 \leq k \leq n$.

We shall refer to the eigenvalues $\lambda_1, \lambda_2, \dots, \lambda_k$ as dominant eigenvalues and to the corresponding eigenvectors as the dominant eigenvectors.

Let v_0 be a linear combination of the eigenvectors. In the power method we let A operate repeatedly on the vector v_0 . In essence, the m^{th} iterated vector is given by

$$v_m = A^m v_0 \quad (3)$$

Apart from exceptional cases,

$$\lim_{m \rightarrow \infty} \left(\frac{1}{\lambda^m} \right) A^m v_0 = x_1 \quad (4)$$

is the dominant eigenvector.

The convergence of the power method can be proven when the elementary divisor of A is non-linear. Detail of this can be found in ([12], p. 582). Here we will prove the convergence of the power method with the assumption that the elementary divisors of A are linear ([2], p. 226). Let x_1, x_2, \dots, x_n be n linearly independent eigenvectors of A corresponding to $\lambda_1, \lambda_2, \dots, \lambda_k$ respectively. Let the dominant eigenvalue be of multiplicity k . We assume that the vector v_0 is "sufficiently rich" in x_1 i.e. v_0 has not too small component in the x_1 direction. This is a rather mild condition, as during the iteration round off error may anyway introduce a spurious component in the x_1 direction. We will express v_0 as a linear combination of the eigenvectors

$$v_0 = \sum_{i=1}^n \alpha_i x_i \quad (5)$$

Equation (5) is multiplied by A repeatedly and in general the m^{th} iterated vector is given by

$$\begin{aligned} v_m = A^m v_0 &= \sum_{i=1}^n A^m \alpha_i x_i = \sum_{i=1}^n \alpha_i \lambda_i^m x_i = \lambda_1^m \sum_{i=1}^k \alpha_i x_i + \sum_{i=k+1}^n \lambda_i^m \alpha_i x_i \\ &= \lambda_1^m \left[\sum_{i=1}^k \alpha_i x_i + \sum_{i=k+1}^n \left(\frac{\lambda_i}{\lambda_1} \right)^m \alpha_i x_i \right] \end{aligned} \quad (6)$$

remembering that the dominant eigenvalue is of multiplicity k . For $m \rightarrow \infty$ the term in the bracket converges to $\sum_{i=1}^k \alpha_i x_i$. The sequence converges to an eigenvector corresponding to λ_1 .

$$\lim_{m \rightarrow \infty} \left(\frac{1}{\lambda_1^m} \right) A^m v_0 = \sum_{i=1}^k \alpha_i x_i.$$

It follows from (6) $\lambda_1 = \lim_{m \rightarrow \infty} \frac{(A^{m+1} v_0)_r}{(A^m v_0)_r}$.

3.1 Power algorithm

- (1) Choose an initial vector y_0 .
- (2) Given a vector y_k the vectors y_{k+1} and z_{k+1} are formed as follows

$$z_{k+1} = A y_k \text{ and } y_{k+1} = \frac{z_{k+1}}{c_{k+1}} \text{ where } c_{k+1} = \max_i |(z_{k+1})_i| \quad (7)$$

The introduction of c_{k+1} in (7) will prevent z_k from being too large. Successive approximations to λ_1 are computed as the ratio of $\frac{(z_{k+1})_r}{(y_k)_r}$.

3.2 Rate of convergence

Let the eigenvalues of the given matrix be defined in (2). The rate of convergence of the power method depends on how fast the ratio of $\left(\frac{|\lambda_{k+1}|}{|\lambda_k|}\right)^m$ goes to zero. **discussions** The power method converges very slowly if the separation between λ_{k+1} and λ_k is poor; in such cases it may be feasible to speed up the convergence as follows. Consider the matrix $A - qI$ where q is a number and I is the identity matrix. The matrix $A - qI$ has eigenvalues $\lambda_i - q$, and if q is chosen properly, convergence to an eigenvector may be sped up. If, for example, $\lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_n > 0$, then the optimum value of q for the maximum rate of convergence is $\left(\frac{\lambda_2 + \lambda_n}{2}\right)$. Of course we need some knowledge of the eigenvalues to apply this method, but if the eigenvalues are known to be positive, then trial values of q can be easily chosen to find the value yielding the best rate of convergence. This is the shift of origin technique and has proven to be effective for some eigenvalue distribution [[12], p. 572].

If λ is an eigenvalue of A then λ^{-1} is an eigenvalue of A^{-1} . According to a variant of the power method that may be called the inverse power algorithm, sequences $\{x^m\}\{y^m\}$ are formed by the following recursion

$$(i) x^{m+1} = A^{-1}y^m; \quad (ii) y^{m+1} = \frac{x^{m+1}}{\max_i |x_i^{m+1}|}$$

Each step here requires the solution of the linear system $Ax^{m+1} = y^m$.

Using the inverse power method we can find the absolutely smallest eigenvalue and the corresponding eigenvector of the matrix.

The inverse power method can be considerably accelerated by the shift of origin technique, if A is a Stieltjes matrix. A real (n, n) matrix $A = (a_{ij})$ with $a_{ij} \leq 0$ for all $i \neq j$ is a Stieltjes matrix if A is symmetric and positive definite. Let the eigenvalue of A be given by $0 < \lambda_1 \leq \lambda_2 \leq \dots \leq \lambda_n$. If A is irreducible Stieltjes matrix it can be proven ([11], p. 288) that for all $0 < \sigma_m < \lambda_1$ (for $m = 0, 1, \dots$).

$$(A - \sigma_m I)^{-1} > 0, \text{ where: } I \text{ is identity matrix}$$

A^{-1} has the same eigenvectors as $(A - \sigma_m I)^{-1}$, therefore we replace in (i) A^{-1} by $(A - \sigma_m I)^{-1}$ and we get the recursive relation

$$(i) x^{m+1} = (A - \sigma_m I)^{-1}y^m; \quad (ii) y^{m+1} = \frac{x^{m+1}}{\max_i |x_i^{m+1}|}$$

where $\sigma_0 = 0$ and $y^0 > 0$ is the initial value of starting vector. Using Collatz's theorem to the last relation it can be shown in ([1], p. 288) that

$$\underline{\lambda_1^{(m)}} \equiv \sigma_m + \min_{1 \leq i \leq n} \left(\frac{y_i^m}{x_i^{m+1}} \right) \leq \lambda_1 \leq \sigma_m + \max_{1 \leq i \leq n} \left(\frac{y_i^m}{x_i^{m+1}} \right) \equiv \overline{\lambda_1^{(m)}}$$

where the first and the last terms are lower and upper bounds for λ_1 . If equality holds in the above equation than $\underline{\lambda_1^{(m)}} = \overline{\lambda_1^{(m)}} = \lambda$, and the procedure terminates. On the other hand if these bounds are unequal we define the following relationship $\sigma_{m+1} = \underline{\lambda_1^{(M)}}$ and continue the iterative process.

4 QR iteration for eigenvalues

We will discuss the QR algorithm, which is the most frequently used for calculation of the set of eigenvalues of a *general* matrix. However, it does not compute eigenvectors unless the matrix is symmetric. The QR method seeks the reduction of a general matrix to a triangular form with the aid of unitary transformation. Instead of the triangular decomposition, as in LR method, the QR method uses the factors of the type

$$A = QR \quad (8)$$

where Q is a unitary matrix and R an upper triangular matrix.

Theorem 3. *Schur's theorem, every non-singular matrix can be factored in the form (8).*

Proof ([6], pp. 142).

If A is non-singular, it will be shown that the QR decomposition is unique, apart from a diagonal multiplier having elements of modulus unity.

Let Q_1R_1 and Q_2R_2 be the QR decomposition of a non-singular matrix A . $A = Q_1R_1 = Q_2R_2$. Then, $\det(A) = \det(Q_1)\det(R_1) = \det(Q_2)\det(R_2)$. The non-singularity of R_1 and R_2 follows from that of A . Then, $Q_2^H Q_1 = R_2 R_1^{-1} = R^1$, where H denotes the conjugate transpose, and R^1 is upper triangular. Taking the conjugate transpose of the above equation, $Q_1^H Q_2 = (R^1)^H$. Therefore, $(R^1)^H R^1 = I$. From the above equation, it follows that R^1 is also unitary. Equating elements on both sides, it can be easily seen that R^1 is a diagonal matrix, such that the element of R^1 are given by $|r_{ij}^1| = 1$. Therefore,

$$Q_2^H Q_1 = R_2 R_1^{-1} = R^1 D, \quad Q_2 = Q_1 D^H, \quad R_2 = D R_1$$

where D is a diagonal matrix with modulus unity's elements. Thus QR decomposition given in (8), is unique apart from a diagonal multiplier having elements of modulus unity. We can always chose D such that $(R_2)_{ii} > 0$. Then the $Q_2 R_2$ decomposition will be considered normalized. On the other hand, this normalized decomposition is clearly unique. Hence forth, whenever a QR 's decomposition is formed, we will assume that the diagonal elements of R are real positive.

In the QR algorithm, we start from a given matrix $A = A_1$ and from sequences of matrices A_K , Q_K and R_K by the use of

$$A_K = Q_K R_K \text{ for } k = 1, 2, 3, \dots \quad (9a)$$

$$A_{k+1} = R_k Q_k = Q_k^H A_k Q_k = Q_{k+1} R_{k+1} \quad (9b)$$

It can be proved under certain condition that $\lim_{k \rightarrow \infty} A_k$ is an upper triangular matrix.

The proof of the convergence of a generalized form of the QR method, when some eigenvalues are complex conjugate and when the elementary divisors are nonlinear, can be found in [10, 13]. If pairs of eigenvalues of equal modulus are allowed the sequence of A_k is not convergent. For sufficiently large k , A_k is close to a matrix \bar{A}_k of block upper triangular structure. Each diagonal block (1,1) or (2,2). The eigenvalues of \bar{A}_k are those of the diagonal blocks. For sufficiently large k the eigenvalues and eigenvectors of \bar{A}_k are good approximation of those of A_k . The proof of the convergence of the QR method can be found in [5, 13]. The following assumption is made:

The eigenvalues of A are of distinct modulus, say, $|\lambda_2| > \lambda_2 > \dots > |\lambda_n|$.

The proof of the convergence of the QR algorithm follows very similar line to those of LR algorithm. Using (9a) and (9b), it can be shown by induction that

$$A_{k+1} = \xi_k^H A_1 \zeta_k \quad (9)$$

where $\zeta_k = Q_1 Q_2 \dots Q_k$. Thus A_{k+1} is unitarily similar to A_1 . Then, as in the derivation of LR algorithm,

$$\xi_k R_k = (Q_1 R_1)^k = A_1^k \quad (10)$$

where $R_k = R_k R_{k-1} \dots R_1$ Using (10), and as the elementary divisors of A are linear, we can write

$$\xi_k R_k = A_1^k = X D^k X^{-1} = X D^k Y^T \quad (11)$$

where X and Y^T are matrices formed by the right hand and left hand eigenvectors of A , and D is the diagonal matrix. We will now consider the triangular decomposition of matrix PY , defined in connection with (11), and remembering that P is a permutation matrix from (11),

$$\xi_k R_k = X D^k P^T P Y = X D^k P^T L_y R_y = X P^T \tilde{D}^k L_y R_y \quad (12)$$

where $\tilde{D}^k = PD^kP^T$ is a diagonal matrix, the element of which are in the order $\lambda_{i1}, \lambda_{i2}, \lambda_{i3}, \dots, \lambda_{in}$. Let

$$XP^T = Q_x R_x \tag{13}$$

where Q_x is a unitary matrix and R_x is an upper triangular matrix. From (12) and (13),

$$\begin{aligned} \xi_k R_k &= Q_x R_x \tilde{D}^k L_y R_y = Q_x R_x \tilde{D}^k L_y \{ \tilde{D}^{-k} \tilde{D}^k \} R_y \\ &= Q_x R_x \{ \tilde{D}^k L_y \tilde{D}^{-k} \} \tilde{D}^k R_y. \end{aligned} \tag{14}$$

Let $D^k L_y \tilde{D}^{-k} = I + F_k$. F_k is a lower triangular matrix, the element (r, s) of which is given by [13] in the LR method. Just as in LR method, $F_k \rightarrow 0$ as $k \rightarrow \infty$. Hence, we obtain from (14), $\xi_k R_k = Q_x R_x (I + F_k) \tilde{D}^k R_y$.

Following derivation of LR method in [13], we get

$$\xi_k R_k = Q_x (I + E_k) R_x \tilde{D}^k R_y \tag{15}$$

where $E_k = R_x F_k R_x^{-1}$, and $E_k \rightarrow 0$; $I + E_k \rightarrow I$ as $I \rightarrow \infty$. Therefore, for sufficiently large k , $(I + E_k)$ cannot be singular. Hence it has QR decomposition and we may write

$$I + E_k = Q_E^k R_E^k, \text{ where: } I \text{ is identity matrix} \tag{16}$$

where Q_E^k, R_E^k are unitary and upper triangular respectively. It can be shown that $Q_E^k \rightarrow I$ and $R_E^k \rightarrow I$ as $E_k \rightarrow 0$, and we sketch here the proof*. Multiply (16) by $(Q_E^k)^H$, we get $(Q_E^k)^H + G_k = R_E^k$. Where $G_k = (Q_E^k)^H E_k \rightarrow 0$ together with E_k as $k \rightarrow \infty$. By an argument, similar to the uniqueness proof of the QR decomposition,

$$(R_E^k)^H R_E^k = I + H_k, \text{ where: } I \text{ is identity matrix} \tag{17}$$

where $H_k = Q_E^k G_k + G_k^H (Q_E^k)^H + G_k^H G_k \rightarrow 0$ as $k \rightarrow \infty$, and thus R_E^k is “near unitary” for large k . For simplicity’s sake, let us consider the (3,3) case, as $(n \times n)$ case can be handled similarly. Let R_E^k and H_k be given by:

$$R_E^k = \begin{pmatrix} r_{11} & r_{12} & r_{13} \\ 0 & r_{22} & r_{23} \\ 0 & 0 & r_{33} \end{pmatrix}, \text{ and } H_k = \begin{pmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{pmatrix}$$

Then (17) implies the following system of equations: $\overline{r_{11}} r_{11} = 1 + h_{11}$, where the bar above r denotes the conjugate.

$$\overline{r_{11}} r_{12} = h_{12}, \quad \overline{r_{11}} r_{13} = h_{13}, \quad \overline{r_{11}} r_{12} = h_{21}$$

Note:*Publications used, as source for this manuscript did not discuss this point.

$$\begin{aligned} \overline{r_{12}} r_{12} + \overline{r_{22}} r_{22} &= 1 + h_{22} \\ \overline{r_{12}} r_{13} + \overline{r_{22}} r_{23} &= h_{23} \\ \overline{r_{13}} r_{11} &= h_{31} \\ \overline{r_{13}} r_{12} + \overline{r_{23}} r_{22} &= h_{32} \\ \overline{r_{13}} r_{13} + \overline{r_{23}} r_{23} + \overline{r_{33}} r_{33} &= 1 + h_{33} \end{aligned}$$

If each $r_{ij} \rightarrow 0$ with $k \rightarrow \infty$ and if r_{ii} are chosen to be positive, then it can be shown recursively that

$$\begin{aligned} r_{ii} &\rightarrow 1, \\ \text{and if } i \neq j \text{ then,} \\ r_{ij} &\rightarrow 0, \end{aligned}$$

Thus the matrix $R_E^k \rightarrow I$ with $k \rightarrow \infty$. From (16), if $R_E^k \rightarrow I$ and $E_k \rightarrow 0$ as $k \rightarrow \infty$, then $Q_E^k \rightarrow I$ as $k \rightarrow \infty$. We will now return to the proof of the QR method. From (15) and (16), $\xi_k R_k = Q_x(Q_E^k R_E^k)R_x D^k R_y$.

We proved at the beginning of this section that the decomposition given by (8) is unique if we select the diagonal element of R_k ($k = 1, 2, 3, \dots$) to be positive. Therefore $\xi_k = Q_x Q_E^k$. For sufficiently large k , $Q_E^k \rightarrow I$. Therefore $\xi_k \rightarrow Q_x$ thus ξ_k tends to Q_x as $k \rightarrow \infty$. From (9), and (11),

$$A_{k+1} = \xi_k^H A_1 \zeta_k \rightarrow Q_x^H X D X^{-1} Q_x = Q_x^H X (P^T P) D (P^T P) X^{-1} Q_x.$$

Using the decomposition of $(X P^T)$ from (13), we get

$$A_{k+1} \rightarrow Q_x^H Q_x R_x (P D P^T) R_x^{-1} Q_x^H Q_x = R_x \tilde{D} R_x^{-1}.$$

Where $\tilde{D} = P D P^T$. Hence the matrix A_k converges indeed to an upper triangular matrix.

4.1 OR algorithm

Each step of the QR algorithm requires the QR factoring of A_k ($k = 1, 2, 3, \dots$). For theoretical purposes it was convenient to think of this in terms of the Schmidt ortho-normalization of the column of A_k . In practice, due to round off error, this commonly gives rise to a Q_k which is by no means orthogonal [[12], p. 243]. In such cases the numerical stability associated with unitary similarity transformation could be lost. In practice the matrix Q_k^H is determined such that

$$Q_k^H A_k = R_k$$

The matrix Q_k^H is determined as the product of plane rotations by Given's method^[6], or by the use of unitary matrices using Householder's method. Here only Householder's triangularization will be discussed briefly.

In the Householder's method, the matrix Q_k^H is obtained as the product of elementary unitary matrices of the form

$$u = I - 2ww^H, \quad (18)$$

where I is identity matrix. Here w is a column vector such that

$$w^H w = 1 \quad (19)$$

Further using (18), we have, $u^H u = (I - 2ww^H)(I - 2ww^H) = I$, that is, u is unitary. Let B denote a (n, n) matrix. Householder's method is a finite iteration consisting of $(n - 1)$ steps. Let B_r be the r^{th} step of Householder's technique to factor B as QR. On the completion of the r^{th} step, B is already upper triangular in its first r columns, we may write

$$B_r = \begin{pmatrix} R_r & v_r \\ \underbrace{0}_{r} & \underbrace{w_{n-r}}_{n-r} \} n-r \end{pmatrix}$$

where R_r is an upper triangular matrix of order r , v_r , and w_{n-r} are of orders as shown in the diagram.

In the $(r+1)^{\text{th}}$ step, we used a matrix defined by (18) and of the form

$$u_r = \begin{pmatrix} I & 0 \\ 0 & I - 2ww^H \end{pmatrix}$$

Thus the vectors w in (19) are constructed so that the $(r+1)^{\text{th}}$ step, the first r components of w are zero. The $(r + 1)^{\text{th}}$ step is given by

$$B_{r+1} = u_r B_r = \begin{pmatrix} R_r & v_r \\ 0 & (I - 2ww^H)w_{n-r} \end{pmatrix}$$

so that only w_{n-r} is modified. w is chosen so that the first column of $(I - 2ww^H)w_{n-r}$ is null except its first element. Therefore, at the end of $(n - 1)^{th}$ steps,

$$u_{n-1} \dots u_1 = B_{n-1}.$$

The product of unitary matrices is unitary and B_{n-1} is upper triangular.

$$u_{n-1} \dots u_1 = Q^H \text{ and } B_{n-1} = R.$$

Therefore, $Q^H B = R$ or $B = QR$ Thus the QR factoring of a matrix B is achieved.

4.2 Rate of convergence

In the QR method, the sub-diagonal elements (i, j) of $A_k (k = 1, 2, 3, \dots)$ converge to zero as fast as $\left(\frac{\lambda_i}{\lambda_j}\right)^k$. The convergence could be slow if the separation of the eigenvalues is poor.

4.3 Application and limitations

The decomposition of A into QR is too laborious (requiring $O(n^3)$ operations) for full matrices. For this reason, the matrix is first transformed into Hessenberg form. It can be shown that the Hessenberg form is invariant with respect to the QR transformation [[12], p.502].

The great benefit the QR with respect to LR transformation is that, because of the use of unitary transformation it is very stable numerically.

A considerable economy in the total number of computation can be obtained by using the following technique. If during the course of the iteration, the magnitude of any sub-diagonal element in position $(i, i+1)$ does not exceed a tolerance ε , the eigenvalue problem of A is reduced in approximation to that of matrix A_{11} , A_{22} of order $i, n - i$, respectively, as shown for a matrix A_k .

$$B_r = \begin{pmatrix} A_{11} & & A_{12} \} r \\ 0 & \varepsilon & \\ \underbrace{0}_{r-1} & \underbrace{0}_1 & \underbrace{A_{22}}_{n-r} \} n-r \end{pmatrix}$$

The eigenvalues of A_k are approximated by the eigenvalues of A_{11} , and A_{22} . If the sub-diagonal element (not exceeding ε) happens to be in the position $(n, n - 1)$, then $(A_k)_{nn}$ may be regarded as an eigenvalue and deflation to a Hessenberg matrix of order $(n - 1)$ may be achieved by dropping the last row and column. If it is in the $(n - 1, n - 2)$ position, then the eigenvalue of the $(2, 2)$ matrix in the lower right hand corner may be regarded as the eigenvalues of the original matrix, and deflation to a Hessenberg matrix of order $(n - 2)$ may be achieved [[12], p. 126].

The rate of convergence of the QR method is better by shifting the origin at each iteration, as discussed in connection with the power method. Arithmetic throughout the process can be maintained in real numbers by combining two origin shifts or a pair of complex conjugate origin shifts. An algorithm treating the QR process can be found in [10].

5 Oepomo's iteration for eigenvalues

Let A be an $n \times n$ essentially positive matrix. The algorithm can be used to numerically determine the eigenvalue λ_A with the largest real part and the corresponding positive eigenvector $x[A]$ for essentially positive matrices. This algorithm is based on previous manuscript^[8, 9]. A matrix $A = (a_{ij})$ is an essentially positive matrix if $a_{ij} \geq 0$ for all $i \neq j, 1 \leq i, j \leq n$, and A is irreducible.

Let $x > 0$ be any positive n components vector^[9]. Let

$$z_i(x) = \sum_{r=1, r \neq i} a_{ir} x_r; \quad (20)$$

$$f_i(x) = \frac{(Ax)_i}{x_i} \equiv \sum_{j=1}^n \frac{a_{ij} x_j}{X_i} \quad (i \in N); \quad (21)$$

$$m(x) = \min_{i \in N} f_i(x);$$

$$M(x) = \max_{i \in N} f_i(x); \quad (22)$$

$$\Delta(x) = M(x) - m(x); \quad (23)$$

$$\|x\| = \sum_{i=1}^n x_i. \quad (24)$$

The following theorem is an application of corollary 2.3 from [9] to the design of algorithms using the Perron-Frobenius-Collatz minimax principle for the calculation $x[A]$. Let x^p ($p = 0, 1, 2, \dots$) be a sequence of positive vectors and $x^p = [x_1^p, x_2^p, \dots, x_n^p]^T$.

Theorem 4. *If the sequence $\{x^p\}$ ($p = 0, 1, 2, \dots$) of positive unit vectors is such that either $m(x^p) \rightarrow \Lambda[A] \equiv \lambda_A$ or $M(x^p) \rightarrow \Lambda[A] \equiv \lambda_A$ as $p \rightarrow \infty$ then $x^p \rightarrow x[A] \equiv \xi$. Moreover, the sequence $\{m(x^i)\}$, $\{x^i\}$ are equi-convergent in the sense that an index v and a constant $K > 0$ exist such that $\|x^i - \xi\| < K[\lambda - m(x^i)]$ if $i \geq v$. Similar statements can be expressed if $M(x^i) \rightarrow \Lambda[A] \equiv \lambda_A$ is known.*

Proof ([9], theorem 2.4, and [8], theorem 1). .

5.1 Numerical implication of theorem 4

We will now define a group of sequences, the “decreasing-sequence” which will be defined later.

Decreasing-sequence. Let $Y^r(x)$ ($r = 1, 2, 3, \dots, n$) be an n component vector valued function such that the following equation is valid:

$$Y_i^r(x) = x_i \quad \text{if } i \neq r, \text{ or } \Omega_r(x) \text{ if } i = r. \quad (25)$$

Here, $\Omega_r(x)$ ($r = 1, 2, 3, \dots, n$) are scalar valued functions which are having properties as follows:

(a) $\Omega_r(x)$ is a continuous positive valued functions which maps the set of positive vectors V_+ into a set of real numbers R .

(b) $\Omega_r(x) \leq x_r$.

(c) $f_r(Y^r(x)) \leq M(x)$ (26*)

(d) Equality in (c) may be applicable only if $Y^r(x^{r-1})$ converges to x ; this yields that $f_r(x) = M(x)$.

(e) If for some $x > 0$ $\Omega_r(x) = x_r$ for each $r \in (N = 1, 2, 3, \dots, n)$ then $f_1(x) = f_2(x) = \dots = f_{n-1}(x) = \lambda$. This will imply that $f_n(x) = \lambda$ according to Collatz, hence $x = x[A]$.

Then n -component vector valued function $Y^r(x)$ defined in (6) will be referred to as the Decreasing-functions. A sequence $\{x^p\}$ ($p = 0, 1, 2, 3, \dots$) of positive n -vectors is constructed which satisfy the conditions of theorem 1. The terms of the sequence $\{x^p\}$ are generated by the following recursive formula:

$$x^{p+1} = Y^{p+1}(x^p) \quad (26)$$

Where $Y^k(x) = Y^{k+n}(x)$ ($k = 1, 2, 3, \dots$). If x^0 is given the sequence $\{x^p\}$ is completely defined. x^{p+1} and x^p differ only in the r^{th} component where

$$r \equiv p + 1 \pmod{n} \quad (27)$$

Such a sequence will be called a decreasing maximum ratio sequence or briefly decreasing-sequence.

Corollary 1. *Any decreasing-sequence converges to x_A .*

Proof. see [8], and Theorem 4 on this manuscript.

We will now define a second group of sequences, the “Increasing-sequence”.

Increasing-sequence. Let $y^r(x)(r = 1, 2, 3, \dots, n)$ be an n component vector valued function such that the following equation is valid:

$$\begin{aligned} y_i^r(x) &= x_i \quad \text{if } i \neq r \\ \omega_r(x) & \quad \text{if } i = r \end{aligned} \tag{28}$$

Here $\omega_r(x) (r = 1, 2, 3, \dots, n)$ are scalar valued functions which are having properties as follows:

(f) $\omega_r(x)$ is a continuous positive valued function and bounded in $\sum_r = (x|z_r(x) > 0; x \geq 0)$.

(g) $\omega_r(x) \geq x_r$

(h) $f_r(y^r(x)) \geq m(x)$ (29*)

(i) Equality in h) may be applicable only if $y^r(x^{r-1})$ converges to x ; this yields that $f_r(x) = m(x)$.

(j) If for some $x > 0 \omega_r(x) = x_r$ for each $r \in (N = 1, 2, 3, \dots, n)$ then

$f_1(x) = f_2(x) = \dots = f_{n-1}(x) = \lambda$. This will imply that $f_n(x) = \lambda$ according to Collatz, hence $x = x[A]$.

Then n -component vector valued function $Y^r(x)$ defined in (9) will be referred to as the Decreasing-functions. A sequence $\{x^p\}(p = 0, 1, 2, 3, \dots)$ of positive n -vectors is constructed which satisfy the conditions of theorem 1. The terms of the sequence $\{x^p\}$ are generated by the following recursive formula:

$$x^{p+1} = y^{p+1}(x^p) \tag{29}$$

Where $y^k(x) = y^{k+n}$ and $y(x)$ is as defined in (9)($k = 1, 2, 3, \dots$). If x^0 is given the sequence $\{x^p\}$ is completely defined. Such a sequence will be called an increasing minimum ratio sequence or briefly increasing-sequence.

Corollary 2. Any Increasing-sequence converges to x_A .

Proof. see proof in [8] and Theorem 4 on this manuscript.

Numerical tests indicate that a simultaneous application of the decreasing and increasing sequences will converge faster than either the decreasing or increasing sequence separately. Therefore, we will define a sequence of vectors $\{x^p\}$ which are constructed by alternating methods of the decreasing or increasing type functions.

We will describe a sequence of n steps which generate $x^{j+1} \dots x^{j+n}(j = 0, n, 2n, \dots)$ in an iteration. If the decreasing ($y^r(x), r = 1, 2, \dots, n$) functions are used to generate the n terms of the sequence $\{x^p\}$ during iteration as defined in (8) then we will say that the iteration is in decreasing mode. Similarly the iteration is in the increasing mode if increasing functions are used as defined in (11). Successive terms of the sequence $\{x^p\}$ can be defined recursively in the following respects:

$$x^{k+1} = Y^{k+1}(x^k) \quad \text{for } k = 0, 1, 2, \dots \tag{30}$$

$$\text{or } x^{k+1} = y^{k+1}(x^k) \quad \text{for } k = 0, 1, 2, \dots \tag{31}$$

Where $k = 0$ corresponds to the input vector. The first iteration could be either in the increasing or in the decreasing mode.

We also define the sequence of real number $\{t_l\} \{T_l\}$ as follows:

$$t_0 = m(x^0). \quad T_0 = M(x^0).$$

At the end of each iteration we consider the following inequalities:

$$m(x^{n \bullet l}) \geq t_{l-1} \tag{32}$$

$$\text{and } M(x^{n \bullet l}) \leq T_{l-1} \tag{33}$$

Where $(l = 1, 2, 3 \dots)$ are indexes of the iteration. If inequalities (34) and (35) are met, we may set

$$t_l = m(x^{n \bullet l}), \quad T_l = M(x^{n \bullet l}) \quad (34)$$

and the mode of the next $(l + 1)^{st}$ iteration will be different from the l^{st} iteration, i.e. if the l^{st} iteration is an increasing sequence then the $(l + 1)^{st}$ iteration is a decreasing sequence or vice versa. If either inequality (34-1) or (34-2) is not satisfied then the mode or direction of the $(l + 1)^{st}$ iteration is the same as that of the iteration and we set:

$$t_l = t_{l-1} \quad T_l = T_{l-1}$$

A sequence having the above-mentioned properties will be called Oepomo's alternating sequence iteration.

Corollary 3. Any Oepomo's alternating sequence iteration converges to x_A .

Proof. see Corollaries 1 and 2, and Theorem 4 on this manuscript.

Corollary 1, 2, and 3 described above lay the foundation of the procedure of an iterative algorithm for the determination of the positive eigenvector of essentially positive matrices. The choice of the functions $\Omega_r(x)$, $\omega_r(x)$ is open but is subject to the restrictions specified in connection with the decreasing and increasing sequences. In theorem 5 and theorem 6 which follow, a possible choice for $\Omega_r(x)$ and $\omega_r(x)$ is given.

Theorem 5. Let $H_r(x)$ ($r = 1, 2, 3, \dots, n$) denote continuous, positive valued functions which map the set of positive vectors V_+ into a set of real number R such that

$$m(x) \leq H_r(x) \leq M(x) \quad (35)$$

and equality may hold on either side of (15) only if $m(x) = M(x) = \lambda_A$. For $r \in N$ ($N = 1, 2, 3, \dots, n$), let

$$\Omega_r(x) = \begin{cases} x_r & \text{if } f_r \geq H_r \\ \frac{z_r}{H_r - a_{rr}} & \text{if } f_r < H_r \end{cases} \quad (36)$$

Where $H_r \equiv H_r(x)$, $f_r \equiv f_r(x)$, and all notations are defined in (1, 2, 3, 4, and 5). Then the functions $\Omega_r(x)$ (together with a starting vector x^0) define a decreasing sequence.

Proof. see [8].

Theorem 6. Let h_r ($r = 1, 2, 3, \dots, n$) denotes a continuous bounded function mapping $\sum_r \rightarrow R$, such that

$$m(x) \leq h_r(x) \leq M(x) \quad (37)$$

and equality may hold on either side of (23) only if $m(x) = M(x) = \lambda_A$, (\sum_r has been introduced in equation (20)). We further define

$$w_r(x) = \begin{cases} x_r & \text{if } f_r \leq H_r \\ \frac{z_r}{h_r - a_{rr}} & \text{if } \frac{z_r}{\|x\|} + a_{rr} < h_r < f_r \\ \|x\| & \text{otherwise.} \end{cases}$$

Then the function w_r (together with an $x^0 > 0$) define an increasing sequence.

Proof. see [8].

5.2 The requirements of functions $h_r(x)$, and $H_r(x)$

The functions $H_r(x)$, and $h_r(x)$ can be selected in many means. The following are a few of the possible choices:

(a)

$$h_r(x) = \frac{1}{2}[M(x) + m(x)] = \mu(x); \quad H_r(x) = \frac{1}{2}[M^*(x) + m(x)] = \mu^*(x)$$

where $r \in N(N = 1, 2, 3, \dots, n)$ and $M^*(x) = \min(m(x), M_1)$ and M_1 is an upper estimate of the eigenvalue λ_A , e.g., $M_1 = M(x^0)$. In [3], $m(x)$ is defined as $\min_{i \in k} R_k(x)$ and $M(x)$ is defined as $\max_{i \in k} R_k(x)$.

(b) For full matrices a reasonable choice for $H_r(x)$, and $h_r(x)$ are derived from the arithmetic mean of the f'_i s.

$$H_r(x) = \sigma(x) = \frac{1}{n} \sum_{i=1}^n f_i(x); \quad \text{and} \quad h_r(x) = \sigma^*(x) = \frac{1}{n} \sum_{i=1}^n f_i^*(x)$$

where $f_i^*(x) = \min(f_i(x), M_1)$.

(c) Another simple choice for $h_r(x)$; $H_r(x)$ is a weighted arithmetic mean of f'_i s:

$$h_r(x) = H_r(x) = \frac{\sum_{i=1}^n f_i(x)x_i}{\|x\|} = v(x)$$

$v(x)$ can also be defined in the following mean:

$$v(x) = \frac{\sum_{i=1}^n b_i x_i}{\|x\|} \tag{38}$$

where $b_i = \sum_{j=1}^n a_{ij}$

5.3 Algorithm

A step of the alternating sequence iteration algorithm consists in modifying a single component x_r of x . As a result $z_i, f_i, \|x\|, v(x)$ will have to be calculated at each iteration. Calculating $z_i, \|x\|$ from their definition in equations (20), and (24) will be referred as recalculating. A considerable reduction of calculation can be accomplished if instead of recalculating these terms are merely updated according to the following steps:

$$\begin{aligned} \|x^{p+1}\| &= \|x^p\| + (x_r^{p+1} - x_r^p) \\ z_i^{p+1} &= z_i^p + a_{ir}(x_r^{p+1} - x_r^p) \quad I = 1, 2, 3, \dots, n \\ v(x^{p+1}) &= v(x^p) + b_r(x_r^{p+1} - x_r^p) \end{aligned} \tag{39}$$

These steps will be referred to as the updating iteration. The updating equations can be obtained easily from equations (20) through (24). To prevent the accumulation of round off errors, after a number of iterations the variables will have to be recalculated instead of updating. If we are working in a double precision, our previous experiences indicate that it is more than sufficient to recalculate after every twenty five iterations.

5.3.1 Over-relaxation method

From various choices for functions $H_r(x)$, $h_r(x)$ and $v(x)$ as indicated in equation (32) seems to give a rapid convergence at least for full matrices. The purpose of this section is to present a variant of equation (39) by introducing the over-relaxation technique. We consider the following equation

$$h_r(x) = (1 - \gamma)f_r(x) + (\gamma)v(x) \tag{40}$$

As it well known, for several suitable values for γ , is the over-relaxation factor, and $1 \leq \gamma < 2$ Equation (35) may be useful in case of banded matrices. The over-relaxation method contains the following cases:

- (a) $\gamma = 1$ for simultaneous over-relaxation method, and
- (b) $1 < \gamma < 2$ for over-relaxation method.

5.3.2 Error vector

In all methods the quantity $\Delta(x) = M(x) - m(x)$ as indicated in (4) is used as a measure of accuracy.

5.4 Discussions

Before we go any further the following issues should be understood. Are both eigenvalues and eigenvectors required to be calculated, or are eigenvalues by itself enough? Is only the absolute largest eigenvalue of interest? Does the matrix have special characteristics, such as real symmetric, essentially positive and so on? If all eigenvalues and eigenvectors are required then this algorithm can not be used;

If a matrix (A) is essentially positive and the positive eigenvector (x_A) and the corresponding eigenvalue (λ_A) are of particular interest, then the algorithm can be used. Each step of the algorithm requires $n^2 + 0(n)$ computations, if the parameters are chosen for the best rate of convergence. It is possible to assume that in half the steps practically no computations are needed, resulting thereby in $\frac{n^2}{2} + 0(n)$ computations for each iteration. As previously stated, after some iteration the variables will have to be recalculated instead of updating. Recalculations need $n^2 + 0(n)$ additional computations. If the computations are performed in double precision, recalculation will not have to be performed so often. As a result recalculations do not increase the total number of computation significantly.

For our numerical comparisons all three algorithms, Power, Oepomo, and QR methods, were tried to solve eigenvalue of the following matrices:

(a) All three algorithms were used to estimate the eigenvalue of Hilbert matrices of various orders. Let H_n be a Hilbert matrix of order n . The elements of Hilbert matrix are defined according to the following relation:

$$a_{ij} = \frac{1}{(i+j-1)} \quad 1 \leq i, j \leq n.$$

The results of the 3 methods can be seen in Tab. 1 ~ 3.

Table 1. Hilbert Matrix H_{40} Power Method

Operations	$\Delta(x)$	$\log \Delta(x)$
1640	1.423	0.3528
4921	1.41×10^{-1}	-1.958
8200	1.39×10^{-2}	-4.275
11490	1.3441×10^{-3}	-6.611
14764	1.296×10^{-4}	-8.949
18040	1.25×10^{-5}	-11.288
21322	1.208×10^{-6}	-13.626
24600	1.116×10^{-7}	-15.965
27880	1.123×10^{-8}	-18.304

Table 2. Hilbert Matrix H_{40} Oepomo Method

Operations	$\Delta(x)$	$\log \Delta(x)$
4800	1.016×10^{-1}	-2.288
9600	3.71×10^{-3}	-5.597
14400	8.549×10^{-5}	-9.368
19200	3.068×10^{-6}	-12.693
24000	7.061×10^{-8}	-16.467
28800	2.536×10^{-9}	-19.289
29700	9.977×10^{-10}	-20.628

(b) We would like to find the efficiency of the three algorithms, when a matrix had eigenvalues of nearly the same modulus. So it was decided to pick a matrix of order n that was almost cyclic (c_n). Consider the below mentioned matrix

$$\begin{bmatrix} A_{1,1} & A_{1,2} \\ A_{2,1} & A_{2,2} \end{bmatrix}$$

The elements of $A_{1,2}$ and $A_{2,1}$ were defined as follows:

$$a_{i,j} = \frac{1}{i+j-1}, A_{1,2} \text{ is a } (8, 12) \text{ matrix, and } A_{2,1} \text{ is a } (12, 8) \text{ matrix.}$$

The elements of $A_{1,1}$ and $A_{2,2}$ were defined in the following respects:

$$a_{i,j} = \frac{10^{-2}}{i+j-1}, A_{1,1} \text{ is a } (12, 12) \text{ matrix, and } A_{2,2} \text{ is a } (8, 8) \text{ matrix.}$$

If the elements of $A_{1,1}$ and $A_{2,2}$ were replaced by zero then the matrix would be nearly cyclic.

Table 3. Hilbert Matrix H_{40} QR Method

Operations	$\Delta(x)$	$\log \Delta(x)$
17400	2.423×10^{-2}	-3.21
23400	1.007×10^{-4}	-9.202
26600	8.263×10^{-9}	-18.614
332000	2.422×10^{-2}	-3.722
442000	1.0096×10^{-4}	-9.212
506000	8.239×10^{-9}	-18.615

Table 4. Almost Cyclic Matrix C_{40} Power Method

Operations	$\Delta(x)$	$\log \Delta(x)$
3780	3.091×10^{-1}	-1.174
7980	2.089×10^{-1}	-1.566
1188	1.421×10^{-1}	-1.950
16390	9.706×10^{-2}	-2.334
20580	6.6381×10^{-2}	-2.714
24780	4.5461×10^{-2}	-3.092
28990	3.111×10^{-2}	-3.478
33190	2.1341×10^{-2}	-3.848
37380	1.4651×10^{-2}	-4.228

Table 5. Almost Cyclic Matrix C_{40} Oepomo Method

Operations	$\Delta(x)$	$\log \Delta(x)$
1400	2.76×10^{-2}	-3.589
2800	7.607×10^{-4}	-7.187
4200	2.293×10^{-5}	-10.699
5600	6.3492×10^{-7}	-14.277
7000	1.9246×10^{-8}	-17.774
8400	5.3279×10^{-10}	-21.358

Table 6. Almost Cyclic Matrix C_{40} QR Method

Operations	$\Delta(x)$	$\log \Delta(x)$
2400	3.091×10^{-1}	-6.622
3260	2.089×10^{-1}	-8.308
4000	1.421×10^{-1}	-14.178
44680	9.706×10^{-2}	-21.798
21340	6.6381×10^{-2}	-6.620
29340	4.5461×10^{-2}	-8.30892
37300	3.111×10^{-2}	-14.178
41340	2.1341×10^{-2}	-21.798

For comparisons, the results of those 3 methods can be seen in Tab. 4 ~ 6.

(c) Introducing a proper shift of origin could speed up the convergence of power algorithm^[8]. So it was decided to try to that kind of matrix such that by introducing a shift of origin would not help the speed of convergence. Such a matrix of order $n(Q_n)$ can be given by the following relations

$$a_{i,j} = \max \left[\frac{n-i}{n}, \frac{n-j}{n} \right] \text{ for } 1 \leq i, j \leq n,$$

$$\&a_{i,i} = \left(\frac{n-i}{n} \right) - \left(\frac{50 \left(i - \frac{n+1}{2} \right)}{n} \right) \text{ for } 1 \leq i \leq n.$$

The results of our tests are indicated in Tab. 7 ~ 9.

Table 7. Matrix Q_{40} Power Method

Operations	$\Delta(x)$	$\log \Delta(x)$
4200	6.506×10^3	8.788
3780	1.0023×10^2	4.608
7990	6.196	1.824
12190	1.065	0.0609
16390	2.1967×10^{-1}	-1.518
20580	4.985×10^{-2}	-2.988
24790	1.160×10^{-2}	-4.458
28990	2.703×10^{-3}	-5.916
33190	6.249×10^{-3}	-7.372

We will here assume that we are interested in the positive eigenvector and the corresponding eigenvalue of the essentially positive matrix. From our trials, it is obvious that in all three cases the rate of convergence of our new algorithm is better or at least as fast as the power^[4]. The QR^[13] algorithm converges very slowly in the last two cases, when the separation between the eigenvalues is poor. Let us consider the results of case b, when the matrix is nearly cyclic. For a cyclic matrix there are some eigenvalues of equal modulus, and so

Table 8. Matrix Q_{40} Oepomo Method

Operations	$\Delta(x)$	$\log \Delta(x)$
2400	3.105	1.134
3260	2.089×10^{-4}	-8.472
4000	4.778×10^{-7}	-14.558
44680	3.124×10^{-10}	-21.879

Table 9. Matrix Q_{40} QR Method

Operations	$\Delta(x)$	$\log \Delta(x)$
4260	$1.429 \times 10^{+1}$	2.6591
5260	1.322×10^{-2}	-4.329
5870	4.664×10^{-5}	-9.9722
41300	$1.4294 \times 10^{+1}$	2.6588
51300	1.324×10^{-2}	-4.3262
55300	4.668×10^{-10}	-9.9722

for a matrix that is “near cyclic” it is plausible to assume the separation between the modules is very poor. The new algorithm takes about 5700 multiplication and divisions to reach an accuracy of 8 digits; which is about 5 times the computations of the power algorithm and the QR algorithm reach an accuracy of 2 digits and 4 digits respectively. We should remember that the QR algorithm is not specifically designed to calculate just one eigenvalue; therefore, a comparable efficiency cannot be expected. Thus from our recent experience we can conclude that the new method shows a good speed of convergence even when the separation of the eigenvalues is poor. However in the case of banded matrices the new algorithm converges slowly. The new algorithm was tried on various banded matrices arising from finite difference approximation to boundary value problems of ordinary differential equations. A computer code was written specially for banded matrices, to take advantage of the large number of zero elements in a banded matrix. We will here summarize the results of our computer runs with the following (20, 20) matrix

$$\begin{aligned}
 a_{ii} &= -2 & \text{if } 1 \leq i \leq n. \\
 a_{i+1,i} = a_{i,i+1} &= 1 & \text{if } 1 \leq i \leq n-1. \\
 a_{i,j} &= 0 & \text{otherwise.}
 \end{aligned} \tag{41}$$

The over relaxation method as described in (35), was tried on the above-mentioned matrix with values of γ ranging from 1 to 1.99. The speed convergence did not show a remarkable dependence of γ . An 8 digit of accuracy was obtained in 168 iterations for $\gamma = 1.73$ whereas for full matrices the new algorithm gave a 9 digit of accuracy in 21 steps.

We will now return our attention to full matrices. Let R_n be a matrix (of order n) with pseudo-random entries. The new algorithm and the power algorithm were tried on each family of matrices (R_n, C_n, H_n) of order $n = 20, 40, \text{ and } 80$. The speed of convergence is almost the same for the two algorithms remembering that each iteration step of the power algorithm requires about twice as many computations. Within one algorithm it is somewhat surprising that the number of iteration steps required for as given accuracy hardly depend on the order of the Hilbert matrix at all.

References

- [1] B. Carnahan, H. Luther, J. Wilkes. *Applied Numerical Methods*. John Wiley, Inc., New York, 1969.
- [2] L. Collatz. Einschliessungssatz fuer die charakteristischen Zahlen von Matrizen. *Math. Z.*, 1942-1943, **48**: 221–226.
- [3] L. Elsner. Verfahren zur Berechnung des Spektralradius nichtnegativer irreducibler Matrizen II. *Computing*, 1972, **9**: 69–73.
- [4] D. Faddeev, V. Faddeeva. *Computational Methods of Linear Algebra*. W. H. Freeman and Company, San Fransisco and London, 1973.

- [5] J. Francis. The QR Transformation. *Comp. Jour.*, 1961, **4**: 265–271.
- [6] C. Froberg. *Introduction to Numerical Analysis*. Addison-Wesley, Inc., 1965. Reading, Massachusetts.
- [7] W. Givens. Computation of Plane Unitary Rotations Transforming a General Matrix to Triangular Form. *J. Soc. Industr. Appl. Math Math.*, 1958, **9**: 26–50.
- [8] T. Oepomo. Oepomo's Algorithm for Computing Largest Real Part Eigenvalues of Essentially Positive Matrices: Collatz & Perron-Frobenius Approach. To be submitted to ELA.
- [9] T. Oepomo. A Contribution to Collatz's Eigenvalue Inclusion Theorem for Nonnegative Irreducible Matrices. *ELA.*, 2003, **10**: 31–45.
- [10] A. Ruhe. Eigenvalue of a Complex Matrix by the QR Method. *BIT.*, 1966, **6**(4): 350–358.
- [11] R. Varga. *Matrix Iterative Analysis*. Prentice-Hall, Englewood Cliffs, NJ, 1962.
- [12] J. Wilkinson. *The Algebraic Eigenvalue Problem*. Clarendon Press, Oxford, 1965.
- [13] J. Wilkinson. Convergence of LR, QR and Related Algorithms. *Comp. Jour.*, 1966, **8**: 77.