# THE *RL* ALGORITHM FOR FINDING EIGENVALUES OF A MATRIX

*By* : SYAMAL KUMAR SEN

(*Central Instruments and Services Laboratory, Indian Institute of Science, Bangalore-12 India*)

[Received July 4, 1969]

## 1. ABSTRACT

*In case of any singular or near singular leading submatrix or submatrices* LR *algorithm for finding eigenvalues of a matrix fails or produces inaccurate results. In this context presented in this paper is the* RL *algorithm which, though of exactly same efficiency as that of* LR *algorithm, usually succeeds in those aforesaid cases. A few numerical examples have been worked out as illustrations.*

## 2. INTRODUCTION

Four possible triangular matrices may be thought of for any square matrix. These triangular decompositions spring from the fact that any square matrix has two and only two diagonals, left and right. In any triangular decomposition, the diagonal elements of any of the two triangular matrices must be specified in order to determine them uniquely from a given square matrix. It is convenient here to define a few matrix terms.

(*a*) Triangular matrix is a square matrix having either a left or a right diagonal below or above which all elements are zeros, thus presenting a form of a bilateral triangle of elements, normally non-zero in nature.

(*b*) Lower triangular matrix of left diagonal ($L_l$) is a square matrix, all the elements of which above the left diagonal are zeros.

(*c*) Upper triangular matrix of left diagonal ($R_l$) is a square matrix, all the elements of which below the left diagonal are zeros.

(*d*) Lower triangular matrix of right diagonal ($L_r$) is a square matrix, all the elements of which above the right diagonal are zeros.

(*e*) Upper triangular matrix of right diagonal ($R_r$) is a square matrix, all the elements of which below the right diagonal are zeros.

Since the conventional matrix multiplication is not commutative, the four triangular matrices $L_l$, $R_l$, $L_r$ and $R_r$ have twelve different product combinations taking any two at a time. Out of these, only five product combinations can represent a given square matrix uniquely, they are $L_lR_l$, $R_lL_l$, $L_lR_r$, $L_rL_l$ and $R_rR_l$. The rest of the possible combinations $R_rL_l$, $L_lL_r$, $R_lR_r$, $R_lL_r$, $L_rR_l$, $R_rL_r$ and $L_rR_r$ only give rise to triangular matrices and are, therefore, of no use in presentation of a given square matrix uniquely.

It can be seen that of the five decompositions cited above, the first two can also represent a square matrix uniquely even if they are multiplied in the reverse way. The last three of these, since they fail in this respect, cannot be used for finding eigenvalues of a matrix. Such reverse multiplication of triangular matrices preserve the properties of similarity transformation[1], thus keeping eigen values invariant. These three decompositions ($L_lR_r$, $L_rL_l$ and $R_rR_l$) as also the $R_lL_l$ (*i e*, $RL$) decomposition, have the advantage over the conventional $L_lR_l$ (*i.e.* $LR$)[1,6] decomposition in that for any singular or near singular[3] leading submatrix or submatrices they usually succeed in complete decompositions. Consequently, they can be used safely for the solution of a linear system where the matrix is having one leading submatrix or more singular. It can be mentioned here that the inverse of a triangular matrix can always be determined easily by working out simple recurrence relations using its known elements. Furthermore the $R_lL_l$ decomposition succeeds in finding eigenvalues of a matrix having one or more leading submatrices singular or near singular accurately, where the $L_lR_l$ decomposition fails or produces inaccurate results. Thus $R_lL_l$ decomposition retains its stability even for non-positive definite matrices while $L_lR_l$ may become unsuccessful. The convergence properties[1] of $RL$ algorithm are exactly the same as those of $LR$ algorithm. These properties can also be analysed in the same fashion as that of $LR$ algorithm for those matrices where $LR$ decomposition fails[9].

This paper presents the complete computational recurrence relations for $R_lL_l$ decomposition while those of $L_lR_l$ decomposition find a description in reference[2]. It, moreover, discusses a few examples for illustrations.

## 3. COMPUTATIONAL RECURRENCE RELATIONS

Simple recurrence relations for finding $R_l$ and $L_l$ (from the symbolic relation $A=R_lL_l$) have been derived explicitly from a given square matrix using conventional matrix multiplication rules (*i.e.* defining unit matrix as a left diagonal matrix with its left diagonal elements unity). Consider the square matrix

$$A=\{a_{ij}\}, \quad i=1,\ 2,\ 3,\ \ldots,\ n; \quad j=1,\ 2,\ 3,\ \ldots,\ n$$

and the triangular matrices

$$L=\{l_{ij}\}, \quad i=1,\ 2,\ 3,\ \ldots,\ n; \quad j=1,\ 2,\ 3,\ \ldots,\ i$$

$$R=\{r_{ij}\}, \quad j=1,\ 2,\ 3,\ \ldots,\ n; \quad i=1,\ 2,\ 3,\ \ldots,\ j$$

It can be seen that the sequence of values of the subscripts $i$ and $j$ (to represent matrix elements) carries the following meaning all throughout, though a brief note will be added for easy and quick access to the recurrence relations. The sequence of subscripts $i$ and $j$ in

$$`i=1,\ 2,\ 3,\ \ldots,\ n; \qquad j=1,\ 2,\ 3,\ \ldots,\ n'$$

indicate that $i$ is to be taken as 1 (fixed) first and $j$ is to be varied from 1 to $n$ at an interval of 1. Then $i=2$ (fixed) and $j=1, 2, 3, \ldots, n$. Next $i=3$ (fixed), $j=1, 2, 3, \ldots, n$ and so on.

The square matrix $A$ when expressed as the product of $R_l$ and $L_l$ matrices (*i e.* $A=R_l L_l$) with the left diagonal elements of $L_l$ specified as unity, results in the following recurrence relations.

$$r_{ij}=a_{ij}-\sum_{p=j+1}^{n} r_{ip}\, l_{pj} \qquad (i \leq j,\ l_{ii}=1 \text{ for all } i) \qquad [1.1]$$

$$l_{ij}=(a_{ij}-\sum_{p=i+1}^{n} r_{ip}\, l_{pj})\,/\,r_{ii} \qquad (i>j) \qquad [1.2]$$

$$i=n,\ n-1,\ n-2,\ \ldots,\ 1; \qquad j=n,\ n-1,\ n-2,\ \ldots,\ 1$$

We first take $i=n$ (fixed) and go on varying $j$ from $n$ to 1 at an interval of 1. As a result, we get $r_{nn}$ from the first relation and $l_{n,n-1}$, $l_{n,n-2}$, $l_{n,n-3}$, $\ldots$, $l_{n1}$ from the second relation. Next we take $i=n-1$ (fixed) and vary $j=n$ to 1 as usual at an interval of 1 and find $r_{n-1,n}$, $r_{n-1,n-1}$; $l_{n-1,n-2}$, $l_{n-1,n-3}$, $l_{n-1,n-4}$, $\ldots$, $l_{n-1,1}$ and so on. Lastly we take $i=1$ (fixed) and vary $j$ from $n$ to 1, and consequently, we get $r_{1n}$, $r_{1,n-1}$, $r_{1,n-2}$, $\ldots$, $r_{11}$.

Now we form the reverse product $S=L_l R_l$, the results of which will constitute the first iteration step. The matrix $S$ which is a square one, does not demand any separate storage, the locations of $A$ serve the purpose. In the left hand side of the following recurrence relations we, therefore, write $a_{ij}$ and not $s_{ij}$. This procedure has double advantage. (*a*) $n^2$ locations are saved thus keeping provisions, in the computer memory, for handling bigger matrices. (*b*) Efficiency is increased due to reduction of computer time, thus rendering the program more automatic.

$$a_{ij}=\sum_{p=1}^{j} l_{ip}\, r_{pj}, \qquad (j \leq i) \qquad [2.1]$$

$$a_{ij}=\sum_{p=1}^{i} l_{ip}\, r_{pj}, \qquad (j>i) \qquad [2.2]$$

$$i=1,\ 2,\ 3,\ \ldots,\ n; \quad j=1,\ 2,\ 3,\ \ldots,\ n$$

The subscripts $i$ and $j$ in the above formulae can be varied in any manner we like.

In the second iteration the transformed matrix $\{a_{ij}\}$ will be treated in the same fashion as it has been done for the first iteration with the original matrix $\{a_{ij}\}$. This process will continue till the continuously transformed matrix $\{a_{ij}\}$ becomes almost an upper triangular matrix or $L_l$ matrix becomes nearly an identity matrix.

## 4. Results

Calculations in all the examples are carried out in 8 dit floating point arithmetic. Due to non-availability of higher precision computer and limited computational facilities available to us numerical calculations with larger precision and with larger order of the matrix could not be carried out. Results are retained correct upto 6 significant figures all throughout. Trace check has been performed to determine the accuracy of the eigenvalues occured in the diagonal of the transformed matrix.

*Example* 1. A matrix having a leading submatrix of order 2 singular.

$$A=\begin{pmatrix} 2 & 4 & 3 & 2 \\ 3 & 6 & 5 & 2 \\ 2 & 5 & 2 & -3 \\ 4 & 5 & 14 & 14 \end{pmatrix}$$

The above matrix fails to oblige *LR* algorithm, since in course of decomposition $r_{22}$ turns out to be zero, and as a result $l_{32}$ and $l_{42}$ cannot be determined. But according to *RL* algorithm

$$A\rightarrow\begin{pmatrix} -.290119\times10^{-1} & .192781\times10^{1} & -.201269\times10^{1} & .200000\times10^{1} \\ .217004\times10^{-27} & .148215\times10^{1} & -.874429\times10^{0} & .285631\times10^{1} \\ .851964\times10^{-38} & .212549\times10^{-10} & .798597\times10^{1} & .177384\times10^{1} \\ .613817\times10^{-42} & .262017\times10^{-14} & .196926\times10^{-2} & .145609\times10^{-2} \end{pmatrix}, \text{ 16 passes}$$

The result is correct at least upto 5 decimal places.

*Example* 2 A matrix of order 4 having leading submatrices of order 2 and 3 singular. This is another case where *LR* algorithm fails.

$$A=\begin{pmatrix} 2 & 4 & 3 & 2 \\ 3 & 6 & 5 & 2 \\ 2 & 4 & 3 & -3 \\ 4 & 5 & 14 & 14 \end{pmatrix}$$

$$A \rightarrow \begin{bmatrix} .506501\times10^{-1} & .236057\times10^{1} & -.318246\times10^{1} & .200000\times10^{1} \\ .100152\times10^{-28} & .249444\times10^{1} & -.235665\times10^{1} & .273792\times10^{1} \\ .447542\times10^{-37} & .638611\times10^{-8} & .852114\times10^{1} & .107837\times10^{2} \\ .275574\times10^{-40} & .310213\times10^{-11} & .716619\times10^{-2} & .139338\times10^{2} \end{bmatrix}, \text{ 17 passes}$$

The result is correct at least upto 4 decimal places.

*Example* 3. A singular matrix of order 4 (the elements of the 4th row are just double the corresponding elements of the 1st row)

$$A = \begin{bmatrix} 1 & 5 & 3 & 7 \\ 2 & 4 & 1 & 6 \\ 3 & 1 & -2 & 3 \\ 2 & 10 & 6 & 14 \end{bmatrix}$$

$$A \rightarrow \begin{bmatrix} -.279397\times10^{-8} & -.439421\times10^{-1} & .727635\times10^{0} & .700000\times10^{1} \\ .000000\times10^{0} & .201548\times10^{0} & -.119029\times10^{1} & .366667\times10^{1} \\ .000000\times10^{0} & .299380\times10^{-1} & -.384547\times10^{1} & -.476015\times10^{0} \\ .000000\times10^{0} & .126704\times10^{-32} & -.527113\times10^{-11} & .206439\times10^{2} \end{bmatrix}, \text{ 17 passes}$$

The result is correct at least upto 4 decimal places.

*Example* 4. A near singular matrix of order 4. All the elements in this matrix are same as those of the previous example save for the (4,2)th element which is here 9 99.

$$A = \begin{bmatrix} 1 & 5 & 3 & 7 \\ 2 & 4 & 1 & 6 \\ 3 & 1 & -2 & 3 \\ 4 & 9.99 & 6 & 14 \end{bmatrix}$$

$$A = \begin{bmatrix} .123948\times10^{-2} & -.405618\times10^{-1} & .727502\times10^{0} & .700000\times10^{1} \\ -.861738\times10^{-39} & .203285\times10^{0} & -.118814\times10^{1} & .368799\times10^{1} \\ -.172921\times10^{-58} & .333843\times10^{-21} & -.384552\times10^{1} & -.475664\times10^{0} \\ .674497\times10^{-71} & .147903\times10^{-32} & -.528014\times10^{-11} & .206410\times10^{2} \end{bmatrix}, \text{ 17 passes}$$

The result is correct at least upto 5 decimal places.

*Example* 5. Hilbert's matrix of order 3 (elements fed correct upto 8 decimal places). Hilbert's matrices are excellent examples of near singular[3] matrices of positive definite nature. The singularity is more pronouneed as we increase the order.

$$A = \begin{bmatrix} .268734 \times 10^{-2} & .258623 \times 10^{-1} & .333333 \times 10^{0} \\ .658049 \times 10^{-27} & .122327 \times 10^{0} & .211368 \times 10^{1} \\ .115379 \times 10^{-43} & .287538 \times 10^{-17} & .140832 \times 10^{1} \end{bmatrix}. \ 17 \text{ passes}$$

The result is correct at least upto 5 decimal places.

*Example* 6. Hilbert's matrix of order 4 (elements fed correct upto 8 decimal places).

$$A = \begin{bmatrix} .966976 \times 10^{-4} & .932890 \times 10^{-3} & .136802 \times 10^{-1} & .250000 \times 10^{0} \\ .826989 \times 10^{-31} & .673827 \times 10^{-2} & .148315 \times 10^{0} & .301496 \times 10^{1} \\ .547841 \times 10^{-52} & .670001 \times 10^{-23} & .169141 \times 10^{0} & .463292 \times 10^{1} \\ .263925 \times 10^{-67} & .359048 \times 10^{-38} & .122134 \times 10^{-15} & .150021 \times 10^{1} \end{bmatrix}, \ 17 \text{ passes}$$

The result is correct at least upto 5 decimal places.

In both the above Hilbert's matrices *LR* algorithm als› produces the identical results. In fact the *LR* algorithm is always stable for positive definite matrices.

## 5. CONCLUSION

We also worked out other typical examples like real matrices with one pair of complex roots, stochastic matrices, Hilbert's matrices of higher order, matrices having double latent roots and disorder of latent roots. In all these cases the results of the *RL* algorithm did not differ from those of the *LR* algorithm.

We can, in fact, deduce the relationship between *LR* and *RL* algorithams as follows:

Let $J$ be a matrix with unity in the right diagonal, above and below which the elements are zeros and let

$$A = R'L'$$

then

$$R' = JLJ$$

$$L' = JRJ$$

where $JAJ = LR$ is the standard triangular factorization of $JAJ$.

Proof: $JAJ = JR'L'J = (JR'J)\ (JL'J) = LR$ by uniqueness.

We can thus, conclude that the *RL* algorithm, though immune to some vanishing leading minors, fails when some trailing minors vanish.

## 6. ACKNOWLEDGMENT

The author wishes to express his gratitude to Prof. P. L. Bhatnagar and to Dr. S. Dhawan, Director for constant encouragement.

## REFERENCES

1. H. Rutishauser .. .. "Solution of eigenvalue of problems with the LR-transformation." National Bureau of Standards, Applied Mathematics Series, 1958, **49**, 47.

2. Syamal Kumar Sen .. .. *J. Indian Inst. Sci.*, 1968, **50**, 210.

3. ———, .. .. *J. Indian Inst. Sci.*, 1967, **49**, 37.

4. ———, .. .. *J Indian Inst. Sci.*, 1968, **50**, 37.

5. J. H. Wilkinson .. .. The Algebraic Eigenvalue Problem, Clarendon Press, Oxford, 1965, 47.

6. L. Fox .. .. An Introduction to Numerical Linear Algebra, Clarendon Press, Oxford, 1964, 130.

7. J. G. F. Francis .. .. *Computer J.*, 1961, **4**, 265,

8. ———, .. .. *Ibid*, 1961, **4**, 332.

9. E. Bodewig .. .. Matrix Calculus (second edition), Interscience publishers. New York, North Holland Publisning Company, Amesterdam, 1959, 140.