UNIVERSITY OF ILLINOIS

DIGITAL COMPUTER LABORATORY

ILLIAC PROGRAM LIBRARY

Library Routine M 25 - 262

TITLE: Eigenvalues and Eigenvectors of a Symmetric Matrix (SADOI Only)

TYPE: Open

NUMBER OF WORDS: 155 + (R 1) = 164

TEMPORARY STORAGE: 0 - 21

DURATION: $5n^3$ milliseconds per iteration; the number of iterations varies from 4 to 8.

SCALING: The sums of the squares of the elements of the matrix must be less than one half.

ACCURACY: About 8 or 9 decimal places

PRESET PARAMETERS:

S3: Location of symmetric matrix

S4: 00F 00nF n = order of matrix.

S5: 00F 00mF m = 2n if eigenvectors are wanted, otherwise m = n.

S6: 00F 00$\frac{n(n+1)}{2}$ S3

S7: 00F 00n^2S6

SYMBOLIC ADDRESSES USED:

M 25, R 1

METHOD OF USE: The lower off diagonal elements and diagonal elements should be stored consecutively beginning at location S3. If only eigenvalues are desired then only $n(n+1)/2$ memory locations are required for the matrix A. If eigenvectors as well are wanted then an additional n^2 memory locations must be reserved for the eigenvectors.

RESULTS: The off diagonal elements of the original matrix are reduced to close to zero. (See Description of Method and Convergence Criterion). The elements of the orthogonal matrix of eigenvectors are stored consecutively in locations beginning at S6, scaled by 1/2. Thus the matrix

$$\begin{matrix} a_{11} & & \\ a_{21} & a_{22} & \\ \vdots & & \\ a_{n1} & \cdots & a_{nn} \end{matrix}$$

becomes

$$\begin{array}{llll} e_1 & & & \\ 0 & e_2 & & \\ 0 & 0 & e_3 & \\ \vdots & & & \\ 0 \ldots . & 0 & e_n & \end{array}$$ followed by

$k_{11}, k_{12}, \ldots, k_{1n}$
$k_{21}, k_{22}, \ldots, k_{2n}$
$\vdots$
$k_{n1} \ldots\ldots\ldots, k_{nn}$ where if we denote by v_j the jth eigenvector its n components are: $k_{1j}, k_{2j}, \ldots, k_{nj}$, i.e. each column is an eigenvector.

NOTES:

(1) If there is an arithmetic error (see below) two F's will be punched out and the machine will stop on an OF at the right hand side of the 138th word of this routine. The computation can be continued by raising and lowering the white switch.

(2) If the original matrix was scaled down by k, then the eigenvalues are scaled by k.

(3) R 1 is at location 156L.

(4) This program is a revision of M 0 - 141 and replaces it in the library.

(5) Care should be taken so that the user does not exceed the capacity of the Williams Memory. In the complete program version of this routine the maximum size of the matrix A is 21 x 21 if eigenvalues and eigenvectors are desired and 37 x 37 if eigenvalues only are desired. If eigenvalues and eigenvectors are desired then the Williams Memory from S3 to S3 + $n(n+1)/2 + n^2 - 1$ will be reserved for the matrix and the eigenvectors. If eigenvalues only are desired then Williams Memory S3, S3 + $n(n+1)/2 - 1$ will be reserved for the matrix A.

(6) Since $|\Sigma a_{ij}b_{jk}| \leq \Sigma|a_{ij}|\ \Sigma|b_{jk}| = 1$ the results of the successive multiplication of orthogonal matrices remain in scale <u>if round-off is ignored</u>. In the presence of round-off, overflow is prevented by using 1/2 I (the identity matrix), instead of I.

BRIEF DESCRIPTION OF THE METHOD:

Let A be a symmetric n x n matrix. We define a matrix O_{jk} as follows:

$$O_{jk} = \begin{array}{c} \\ \\ k \\ j \\ \\ \end{array} \begin{array}{c} \begin{array}{ccc} \quad & k & \quad j \end{array} \\ \begin{pmatrix} 1 & & & & \\ & \ddots & \vdots & \vdots & \\ \cdots & & \cos\varphi & -\sin\varphi & \\ \cdots & & \sin\varphi & \cos\varphi & \\ & & & \ddots & \\ & & & & 1 \end{pmatrix} \end{array}$$

where all diagonal elements except O_{jj}, O_{kk} equal 1 and all off diagonal elements except O_{jk} and O_{kj} equal 0.

$$\left\{ \begin{array}{l} O_{jj} = O_{kk} = \cos\varphi, \\ O_{jk} = \sin\varphi; \; O_{kj} = -\sin\varphi \end{array} \right. \qquad j > k$$

It is easily verified that O_{jk} is an orthogonal matrix and hence $O_{jk}^T = O_{jk}^{-1}$. Since A is symmetric then, by a well known theorem of algebra, there exists an orthogonal matrix O such that $O^TAO = D$ a diagonal matrix; in the language of linear algebra "any real quadratic form in n variables assumes the diagonal form, relative to a suitable orthonormal basis." (Birkhoff and MacLane p. 277). This routine constructs the matrix O^T by an iterative process which at the same time reduces A to its diagonal form D. In this iterative procedure the orthogonal matrices O_{jk} play a fundamental role.

First we form $O_{jk}^T AO_{jk} = B$ and note that B differs from A only in the jth and kth rows and jth and kth columns.

$$\left\{ \begin{array}{l} b_{k\ell} = a_{j\ell} \cos\varphi + a_{k\ell} \sin\varphi \\ b_{j\ell} = -a_{k\ell} \sin\varphi + a_{j\ell} \cos\varphi \end{array} \right.$$

$$\left\{ \begin{array}{l} b_{kk} = a_{jj} \cos^2\varphi + 2a_{jk} \cos\varphi \sin\varphi + a_{kk} \sin^2\varphi \\ b_{jj} = a_{jj} + a_{kk} - b_{kk} \end{array} \right.$$ [under an orthogonal transformation $\sum_i a_{ii} = \sum_i b_{ii}$]

$$\left\{ \begin{array}{l} b_{jk} = a_{jk}(\cos^2\varphi - \sin^2\varphi) + (a_{jj} - a_{kk}) \sin\varphi \cos\varphi \\ \quad\;\; = a_{jk} \cos 2\varphi + 1/2 \, (a_{jj} - a_{kk}) \sin 2\varphi \end{array} \right.$$

A necessary and sufficient condition for $b_{jk} = 0$ is that φ satisfies the following equation:

$$\tan 2\varphi = \frac{2a_{jk}}{a_{kk} - a_{jj}}$$

It should be noted that φ is not uniquely determined by this condition as $\varphi + \pi/2$ will also satisfy this equation.

But the important thing is to choose $\sin \varphi$ and $\cos \varphi$ in a manner consistent with the condition

$$\tan 2\varphi = \frac{2a_{jk}}{a_{kk} - a_{jj}}$$

and no matter how this is done the result is to set $b_{jk} = 0$. We have shown that given any non-zero off diagonal element a_{jk} an orthogonal transformation O_{jk} may be constructed such that in the matrix $O_{jk}^T A O' = B$, then $b_{jk} = 0$. If we choose O_{jk} in this way it is easily verified that

$\sum_{r \neq s} b_{rs}^2 = \sum_{r \neq s} a_{rs}^2 - 2a_{jk}^2$ i.e. the sums of the squares of the off diagonal elements are reduced by a positive amount $2a_{jk}^2$. This is easily verified as follows: for $\ell \neq j, k$ $b_{k\ell}^2 + b_{j\ell}^2 = a_{k\ell}^2 + a_{j\ell}^2$. Therefore,

$$\sum_{r \neq s} b_{rs}^2 = \sum_{r \neq s} a_{rs}^2 - 2a_{jk}^2.$$

The method is therefore to select in some way a sequence of $a_{jk} \neq 0$ $(j \neq k)$ and reduce these in succession to zero via the method described above. After a finite number of these rotations the off diagonal elements will be sufficiently small (zero inside the machine since the sum of their squares is constantly being reduced by a positive amount) and the process has converged. Let $\bar{D}$ represent the matrix in the machine to which A has been reduced and let D be the matrix obtained from $\bar{D}$ by setting all off diagonal elements to zero. Then if we denote by $\bar{\lambda}_j$ the eigenvalues of $\bar{D}$ and by λ_j the eigenvalues of D then by a well known theorem of Courant $|\lambda_j - \bar{\lambda}_j| < |D - \bar{D}|$ where $|D - \bar{D}|$ refers to the matrix norm.

If v is an eigenvector with respect to D, its representation with respect to A must be found. $O^T A O v = \lambda v$ implies (multiplying on the left by O) that $A(Ov) = \lambda(Ov)$; hence if v is an eigenvector with respect to D, Ov is an eigenvector with respect to A. Since (1, 0, ..., 0), (0, 1, 0, ..., 0), ..., (0, ..., 0, 1) are the eigenvectors with respect to D, it follows that the columns of the matrix 0 represent the eigenvectors with respect to A. Thus if we want the eigenvectors all we need do is form $T_{i+1} = T_i B_i$ (where $T_0 = I$) after each iteration and

$$B_m^T \ldots B_1^T A\, B_1\, B_2 \ldots B_m = D.$$

We note in passing that the passage from $\bar{D}$ to D is not without its dangers as the following simple example shows. Let $A = \begin{pmatrix} 1 & \epsilon^{1/2}/2 \\ \epsilon^{1/2}/2 & 1 \end{pmatrix}$ and $v = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$ then the sum of the squares of the off diagonal elements of A is $\epsilon/2$ but the eigenvectors of A are not $\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$ but $\frac{1}{\sqrt{2}}\begin{pmatrix} 1 & -1 \\ 1 & 1 \end{pmatrix}$. For a detailed discussion of these and other matters relating to this method see the paper by H.H. Goldstine, F.J. Murray, and J. Von Neumann: The Jacobi Method for Real Symmetric Matrices, Journal of the ACM, volume 6, January, 1959, no. 1.

DESCRIPTION OF NUMERICAL OPERATIONS PERFORMED:

When the routine begins, the original symmetric matrix $A = A_1$ is stored in the consecutive memory locations, S3, S3 + 1, ..., etc. and if the eigenvectors are desired the unit matrix, T_0, is generated and stored in the Williams Memory beginning at S6 and ending at S6 + n^2 - 1. After executing the ith transformation each element of A_{i+1} occupies the memory locations previously occupied by the corresponding element of A_i and the same applies to T_{i+1} and T_i. Actually, the machine changes only those elements affected by the matrix multiplication and leaves the remaining elements unaltered. The ith transformation, reducing element a_{jk} to zero, will alter only the elements of A_i in the jth and kth columns and rows and will alter only the elements of T_i in the jth and kth columns. This means we start with a pair of elements, a_{k1}, a_{j1}, and work our way across the kth and jth rows until we reach the diagonal and then go down the kth and jth columns until we reach the last row of the matrix T_i (if eigenvectors are desired; otherwise, we go until the last row of the matrix A_i is reached). In order to keep track of our progress we use ℓ as a tally and when ℓ - S5 is positive we are through. The elements are transformed according to the equations

(1) $a_{k\ell}^{(i+1)} = a_{k\ell}^{(i)} \cos\varphi + a_{j\ell}^{(i)} \sin\varphi$

(2) $a_{j\ell}^{(i+1)} = - a_{k\ell}^{(i)} \sin\varphi + a_{j\ell}^{(i)} \cos\varphi$

where $a^{i}_{k\ell} = a^{i}_{\ell k}$; $a^{i}_{j\ell} = a^{i}_{\ell j}$. However, there are three special cases which do not use these equations and since these elements have already been computed incorrectly we merely write the correct values over the incorrect ones. We set $a^{i+1}_{jk} = 0$ and transform the diagonal elements a^{i}_{jj} and a^{i}_{kk} according to the equations

(3) $a_{kk}^{(i+1)} = a_{kk}^{(i)} \cos^2\varphi + a_{jk}^{(i)} \sin 2\varphi + a_{jj}^{(i)} \sin^2\varphi$

(4) $a_{jj}^{(i+1)} = a_{kk}^{(i)} + a_{jj}^{(i)} - a_{kk}^{(i+1)}$.

The elements a_{jk} for successive transformations are selected in consecutive order along successive rows, i.e., we use a_{21} in matrix A_1, a_{31} in matrix A_2, etc. Of course, if $a_{jk} = 0$ already we do not rotate but go on to $a_{j,k+1}$. We define one iteration to be the $(n^2 - n)/2$ transformations required to reduce each off diagonal element to zero once. It should be stated that once an element is reduced to zero it will not, in general, remain zero during subsequent transformations. However, the sum of the squares of the off diagonal elements will be decreased each time by an amount equal to $2a^2_{jk}$ and thus will become small (zero inside the machine), after a finite number of transformations. The program can be divided roughly into two parts:

(1) Computation of $\sin\varphi$ and $\cos\varphi$:

We always choose $\cos\varphi > 0$. Define $a = 2a_{jk}$; $b = a_{kk} - a_{jj}$.

Case 1: $|a| < |b|$ then $m = \tan 2\varphi$

$$\frac{m}{2} = \frac{a}{2b}$$

$$S = \frac{1}{4}$$

$$\frac{1}{2}\cos 2\varphi = \frac{S}{\sqrt{\frac{1}{4} + \frac{m^2}{4}}} \qquad \sin\varphi\cos\varphi = \frac{m}{2}\cos\varphi$$

Case 2: $|a| \geq |b|$ then $m = \cot 2\varphi$

$$\frac{m}{2} = \frac{b}{2a} \qquad S = \frac{m}{4}$$

$$\frac{1}{2}\cos 2\varphi = \frac{|S|}{\sqrt{\frac{1}{4} + \frac{m^2}{4}}}$$

Case 2a: $S < 0$, $r = -\frac{1}{4}$

Case 2b: $S \geq 0$, $r = \frac{1}{4}$

$$\sin\varphi\cos\varphi = \frac{r}{\sqrt{\frac{1}{4} + \frac{m^2}{4}}}$$

In both cases $\cos\varphi = \sqrt{\frac{1}{2} + \frac{1}{2}\cos 2\varphi}$

$$\sin\varphi = [\sin\varphi\cos\varphi]/\cos\varphi$$

(2) Matrix multiplication:

This has already been mentioned. However, we need to describe the method of obtaining the addresses of the various elements involved in the matrix multiplication. In order to simplify the discussion we make a slight change of notation - we begin our count from 0 instead of 1; i.e., we denote an element of the first row by $a_{0\ell}$ instead of $a_{1\ell}$ and similarly an element from the first column by $a_{\ell 0}$ instead of $a_{\ell 1}$.

First of all once a_{jk} is chosen we need to compute (k, 0); (k, k); (j, 0); (j, j) where these denote the addresses of a_{k0}; a_{kk}; a_{j0}; a_{jj}.

$(k, 0) = \frac{1}{2}(k^2 + k) + S3$

$(k, k) = (k, 0) + k$

$(j, 0) = \frac{1}{2}(j^2 + j) + S3$

$(j, j) = (j, 0) + j$.

These addresses are planted in the orders which carry out the operations indicated in equations (1), (2), (3), (4) above. Now (k, ℓ) and (j, ℓ) $\ell = 0, 1, \ldots, S5$ should be increased by 1 as one moves across the kth and jth rows until the diagonal elements are reached. Then they should be increased by ℓ as one moves down the jth and kth columns until the last row of A is reached, i.e., ℓ = n - 1. Then if eigenvalues are desired they should be increased by n until ℓ = S5. We handle this by storing two increments in the address portions of 130 (M 25). The increment in the left hand address is used to move along the path starting at (k, 0) and the increment in the right hand address is used to move along the path starting at (j, 0). The increments are not always the same since one path reaches the diagonal sooner than the other. The determination of the increments requires that our tally ℓ (in location 11F) be compared with k and j so as to change the increment from 1 to ℓ along each path. It is also necessary to compare ℓ with n so as to change the increments from ℓ to n. Finally we compare ℓ with S5 in order to get out of this loop. As was mentioned earlier we take care of the three special cases ($a_{jj}^{(i+1)}$, $a_{kk}^{(i+1)}$, $a_{jk}^{(i+1)}$) after we leave the loop.

DESCRIPTION OF CONVERGENCE TEST AND ARITHMETIC TEST:

If we define

$$N^2 = \sum_{j,k=1}^{n} a^2_{jk}, \quad E = \sum_{j\neq k} a^2_{jk}, \quad S = \sum_{j\geq k} a^2_{jk},$$

then we have

$$S = N^2 - (\tfrac{1}{2})\, E. \tag{1}$$

Under an orthogonal transformation N^2 remains invariant and E is reduced by an amount equal to twice the square of the element which goes to zero under the transformation, i.e.

$$E_{i+1} = E_i - 2a^2_{j'k'}.$$

Thus using (1),

$$S_{i+1} = N^2 - (\tfrac{1}{2})\, E_{i+1}$$

$$= (N^2 - \tfrac{1}{2}\, E_i) + a^2_{j'k'}$$

$$= S_i + a^2_{j'k'} \tag{2}$$

and we see that $\left\{ S_i \right\}$ forms a monotone-increasing sequence which approaches N^2 as E approaches zero. This is the basis of our convergence test.

Since we form S at the end of each iteration (defined as $\frac{1}{2}$ $(n^2 - n)$ orthogonal transformations) then

$$S^{(i+1)} = S^{(i)} + W^{(i+1)} \tag{3}$$

where $W^{(i+1)}$ means the sum of the squares of the elements which are reduced to zero by each of the $(1/2)$ $(n^2 - n)$ transformations. Equation (3) gives us our convergence test and also provides the means to test the accuracy of our computation. We test the quantity $S^{(i)} - S^{(i+1)}$. If it is negative we know that the process has not converged and we then look at the quantity

$$\mu - |S^{(i)} - S^{(i+1)} + W^{(i+1)}|$$

where μ is our tolerance. If this result is negative the machine stops on an OF order from the right hand side of 138L. It should be mentioned that these quantities have been computed using double precision. When $S^{(i)} - S^{(i+1)}$ becomes positive the process has converged.

DATE April 27, 1959
PROGRAMMED BY W.a. Rosenkrantz
APPROVED BY J N Snyder

lgr

LOCATION	ORDER	NOTES PAGE 1
	00 K(M25)	
0	41 17F	
	41 18F	
1	41 19F	
	L5 15(M25)	Compute eigenvectors?
2	L0 10L	≥ 0 No!
	36 23L	< 0 Yes!
3	22 3L	
	41 S6	Begin clearing
4	F5 3L	n^2 memory locations
	40 3L	for 1/2 unit matrix
5	L0 11L	
	32 3L	≥ 0 Not done
6	L5 20L	
	40 S6	Put 1/2 down the
7	F5 6L	diagonal, thus
	L4 12L	get 1/2 I
8	40 6L	
	L0 13L	
9	36 23L	
	26 6L	
10	00 S5	
	00 S5	
11	K2 3L	
	41 S7	
12	00 F	
	00 S4	
13	L5 20L	
	40 S7	
14	00 1F	
	00 1F	
15	00 S4	
	00 S4	
16	00 S3	
	00 S3	
17	80 S5	
	00 S5	

LOCATION	ORDER	NOTES	PAGE 2
18	20 F		
	00 F		
19	00 F		
	00 F		
20	40 F		
	00 F		
21	00 F		
	00 F		
22	J0 S6		
	74 S6		
	00 K		
0	F5 19(M25)		Advance iterations
	40 19(M25)		counter
1	41 20F		
	41 21F		
2	41 4F		Set j = 0
	41 5F		Set k = 0 (from 99)
3	L5 4F		
	L4 14(M25)		Step j till j = n
4	40 4F		
	L0 15(M25)		
5	32 100L		
	50 5F	from 99'	
6	L5 5F		
	74 5F		
7	00 38F		$(k, 0) = \frac{1}{2} (k^2 + k) + S3$
	42 21(M25)		
8	00 20F		
	46 21(M25)		
9	L5 21(M25)		
	L4 16(M25)		
10	46 130L		
	L4 5F		$(k, 0) + k = (k, k)$
11	42 29L		
	46 94L		

LOCATION	ORDER	NOTES	
12	42 95L		
	50 4F		
13	L5 4F		
	74 4F		
14	00 38F		
	42 21(M25)		$(j, 0) = \frac{1}{2}(j^2 + j) + S3$
15	00 20F		
	46 21(M25)		
16	L5 21(M25)		
	L4 16(M25)		
17	42 130L		
	L4 4F		$(j, j) = (j, 0) + j$
18	46 96L		
	42 28L		
19	L0 4F		
	L4 5F	form (j, k)	
20	42 21L		
	42 22L		
21	42 96L		
	L3 F	by 20	$a_{jk} = 0$?
22	36 97L		
	L5 F	by 20'	
23	40 7F		
	L5 21F		
24	50 7F		
	74 7F		
25	L4 20F		
	40 20F		
26	S5 F		
	40 21F		
27	L5 7F		
	00 1F		
28	40 7F		$2a_{jk} = a$
	L5 F	by 18'	
29	40 9F		a
	L5 F	by 11	

LOCATION	ORDER	NOTES	
30	40 8F		
	L0 9F		$a_{kk} - a_{jj} = b$
31	40 10F		
	L7 7F		$\|a\| - \|b\|$
32	L2 10F		
	50 19F		
33	36 37L		≥ 0 Use cot 2 φ
	L5 7F		< 0 Use tan 2 φ
34	66 10F		
	S5 F		
35	10 1F		
	40 11F		
36	L5 18(M25)		
	26 40L		
37	L5 10F		
	66 7F		
38	S5 F		
	10 1F		
39	40 11F		
	26 124L		
40	40 12F		
	50 11F		
41	75 11F		
	14 18(M25)		$\sqrt{\frac{1}{4} + \frac{m^2}{4}}$ in 14 F
42	40 1F		
	S5 F		
43	40 F		
	50 43L		
44	22 (R1)		
	40 14F		
45	L3 12F		
	50 19F		
46	66 14F		
	S1 F		
47	40 19F		
	22 118L		

LOCATION	ORDER		NOTES
48	40 1F		$\sqrt{\frac{1}{2} + \frac{1}{2} \cos 2\varphi} = \cos \varphi$
	41 F		
49	22 49L		
	50 49L		
50	22 (R1)		$\cos \varphi$ in 2F
	L5 12F		
51	L0 18(M25)		
	32 56L		
52	L5 12F		
	36 54L		
53	L1 18(M25)		
	22 54L		
54	L5 18(M25)		
	50 19F		
55	66 14F		
	S5 F		
56	22 58L		
	19 1F		
57	50 11F		
	74 13F		
58	00 1F		
	40 14F		
59	66 2F		
	S5 F		
60	40 3F		$\sin \varphi$ in 3F
	41 11F		Clear tally counter
61	27 70L		
	L5 11F		
62	L0 15(M25)		
	36 100L		
63	L4 15(M25)		
	40 131L		
64	L5 5F		
	L0 11F		
65	32 67L		
	L5 4F		

LOCATION	ORDER	NOTES	
66	L0 11F		
	32 68L		
67	22 69L		
	L5 14(M25)		
68	46 131L		
	L5 14(M25)		
69	42 131L		
	L5 131L		
70	L4 130L		
	40 130L		
71	46 74L		
	46 78L		Plant (k, ℓ)
72	46 83L		
	42 76L		
73	42 80L		Plant (j, ℓ)
	42 81L		
74	50 F	by 71	
	7J 2F		$a_{k\ell} \cos\varphi + a_{j\ell} \sin\varphi$
75	40 10F		
	S5 F		
76	50 3F		
	74 F	by 72'	
77	L4 10F		
	40 F		
78	50 F	by 71'	
	79 3F		
79	40 10F		$a_{j\ell} \cos\varphi - a_{k\ell} \sin\varphi$
	S5 F		
80	50 2F		
	74 F	by 73	
81	L4 10F		
	40 F	by 73'	
82	22 82L		
	L5 F		
83	40 F	by 72	
	L5 11F		

LOCATION	ORDER		NOTES
84	L4 14(M25)		Advance ℓ
	40 11F		
85	L0 17(M25)		ℓ = S5?
	32 61L		≥ 0 No
86	LJ 13F		< 0 Compute
	40 F		
87	L9 13F		$a_{jj}^{(i+1)}$; $a_{kk}^{(i+1)}$
	40 1F		
88	50 7F		and set $a_{jk}^{(i+1)} = 0$
	7J 14F		
89	40 10F		
	S5 F		
90	50 8F		
	74 F		
91	L4 10F		
	40 10F		
92	S5 F		
	50 9F		
93	74 1F		
	L4 10F		
94	40 F	by 11'	
	L5 8F		
95	L4 9F		
	L0 F	by 12	
96	40 F	by 18	
	41 F	by 21	
97	L5 5F		Advance k
	L4 14(M25)		
98	40 5F		
	L0 4F		k = j?
99	32 2L		
	22 5L		
100	27 63L		
	41 15F	from 5	
101	41 16F		
	L5 16(M25)		

LOCATION	ORDER	NOTES	PAGE 8
102	42 104L		
	46 104L		
103	22 103L		
	L5 16F		
104	50 F	by 102	
	74 F		
105	L4 15F		Form $\sum_{j \geq k} a_{jk}^2 = S^{(i+1)}$
	40 15F		
106	S5 F		
	40 16F		
107	L5 104L		
	L4 14(M25)		
108	40 104L		
	L0 22(M25)		
109	32 103L		
	L5 18F		
110	L0 16F		
	10 39F		
111	L4 17F		
	L0 15F		
112	36 9(R1)		$S^{(i)} - S^{(i+1)}$
	L5 128L		If ≥ 0 then process has converged } Bypass arithmetic test on 1st iteration
113	26 126L		
	19 34F		
114	L2 F		
	36 116L		Arithmetic fail stop
115	92 904F		
	OF F		
116	L5 15F		
	40 17F		
117	L5 16F		
	40 18F		
118	26 L		
	L5 13F		
119	L0 123L		
	36 121L		

LOCATION	ORDER	NOTES	PAGE 9
120	LJ 13F		
	26 48L		Overflow test
121	LJ 123L		
	40 2F		
122	22 50L		
	00 F		
123	3L 4095F		
	LL 4095F		
124	F5 11F	from 39'	
	10 1F		
125	26 40L		
	00 F		
126	40 112L	from 113	
	L5 129L		
127	40 113L		
	26 116L		
128	36 9(R1)		
	L4 20F		
129	40 F		
	19 34F		
130	00 F		
	00 F		
131	00 F		
	00 F		
	(R1) 00K		Square Root Routine