BSR 2949

SIGNATURE DATA PROCESSING STUDY FINAL REPORT

Volume II
Equations and Flow Diagrams

By:
Cheryl L. Crawford
David S. Hanson

August 1970

Prepared Under Contract No. NAS 9-9848 by
THE BENDIX CORPORATION AEROSPACE SYSTEMS DIVISION
Ann Arbor, Michigan

Manned Spacecraft Center

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION

CR - 11

BSR 2949

SIGNATURE DATA PROCESSING STUDY
FINAL REPORT

Volume II
Equations and Flow Diagrams

By:
Cheryl L. Crawford
David S. Hanson

August 1970

Distribution of this report is provided in the interest of information exchange and should not be construed as endorsement by NASA of the material presented. Responsibility for the contents resides with the organization that prepared it.

Prepared Under Contract No. NAS 9-9848 by
THE BENDIX CORPORATION AEROSPACE SYSTEMS DIVISION
Ann Arbor, Michigan

Manned Spacecraft Center

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION

CONTENTS

ILLUSTRATIONS

TABLES

SECTION 1

ANALYSIS PROGRAMS

The data simulation, factor analysis, cluster analysis, classification, and information programs represent analytical techniques developed and applied during the study to accomplish signature data processing. The data simulation program provides data from known distributions with which to evaluate techniques and software. The factor analysis program is the product of several years of development and is specifically tailored for analysis of multispectral data. The cluster analysis program employs a new technique for identification of cluster centers in multivariate hyperspace. It can be applied operationally to reduce the need for or to improve the available ground truth.

The classification programs are not intended for classification of data under operational conditions, but for evaluation of the relative performance of various classification techniques. Both these and the information divergence programs provide for several ways of obtaining the probability density function used in the processes. The performance obtained with the different expressions for the density function is compared to assess the relative merits of the available alternatives.

The following sections are intended to provide descriptions of basic features, logic flow, and the underlying mathematics of these analysis programs.

1.1 SIGNATURE DATA SIMULATION PROGRAM

1.1.1 General Description

The Signature Data Simulation Program generates random data, in one of three forms, from distributions which are specified parametrically. Use of the simulated data serves to test program logic, to evaluate the effects of varying program parameters, and to establish the performance of the analytical techniques used by evaluating data from known distributions.

A number of clusters with means and standard deviations for each channel is specified by the user. The correlation matrix of the cluster determines the departure of the equal probability contours of the cluster from sphericity and also the directions of the axis of the equal probability ellipsoids. Three options are available in the program concerning correlation:

1. The data may be uncorrelated

2. The user may specify eigenvalues, determining the relative lengths of the axes and requiring random orientation by the program

3. The user may specify the correlation matrix requiring use of the EIGEN subroutine before data are generated by the main program.

The user may request that the data be generated in parametric form (cluster weight, means, covariance matrix, factor loadings matrix), digitized scanner form, or continuous form.

1.1.2 Flow Chart of the Data Simulation Program

The reference to subroutine EIGEN, made in Figure 1-1 of the Data Simulation Program is satisfied in Section 1.2.2, where the EIGEN flow chart is presented.

KEY TO NOTATION

Main Program, Data Simulation

ix	=	initial value for random number generator
$_8U_8$	=	Hadamard orthogonal matrix
m	=	number of channels
ng	=	number of groups of data to be simulated
Nk_ℓ, $\ell = 1$, ng	=	number of clusters in the ℓ th group
Nob_{ij}, j=1, ng, i=1, Nk_j	=	number of observations in i^{th} cluster of j^{th} group
Nn_ℓ, ℓ=1, ng	=	number of observations in ℓ th group
W_{ij}, i=1, Nk_j, j=1, ng	=	weight of i^{th} cluster of j^{th} group, proportional to number of observations in cluster
$B\ell_i$, i=1, m	=	lower bound of i^{th} channel's scanner range

Bu_i, i=1,m	=	upper bound of ith channel's scanner range
Z_i, i=1,m	=	mean of ith channel
sd_i, i=1,m	=	standard deviation of ith channel
R_{ij}, i=1,m, j=1,m	=	correlation matrix of cluster
$G\ell_i$, i=1,m	=	eigenvalues of correlation matrix
T_{ij}, i=1,m, j=1,m	=	eigenvectors
A_{ij}, i=1,m, j=1,m	=	covariance matrix of cluster
TT_{ij}, i=1,m, j=1,m	=	factor loadings matrix
Ti_{ij}, i=1,m, j=1,m	=	factor deviation matrix
X_i, i=1,m	=	signature in continuous form
Jx_i, i=1,m	=	signature in digitized scanner form.

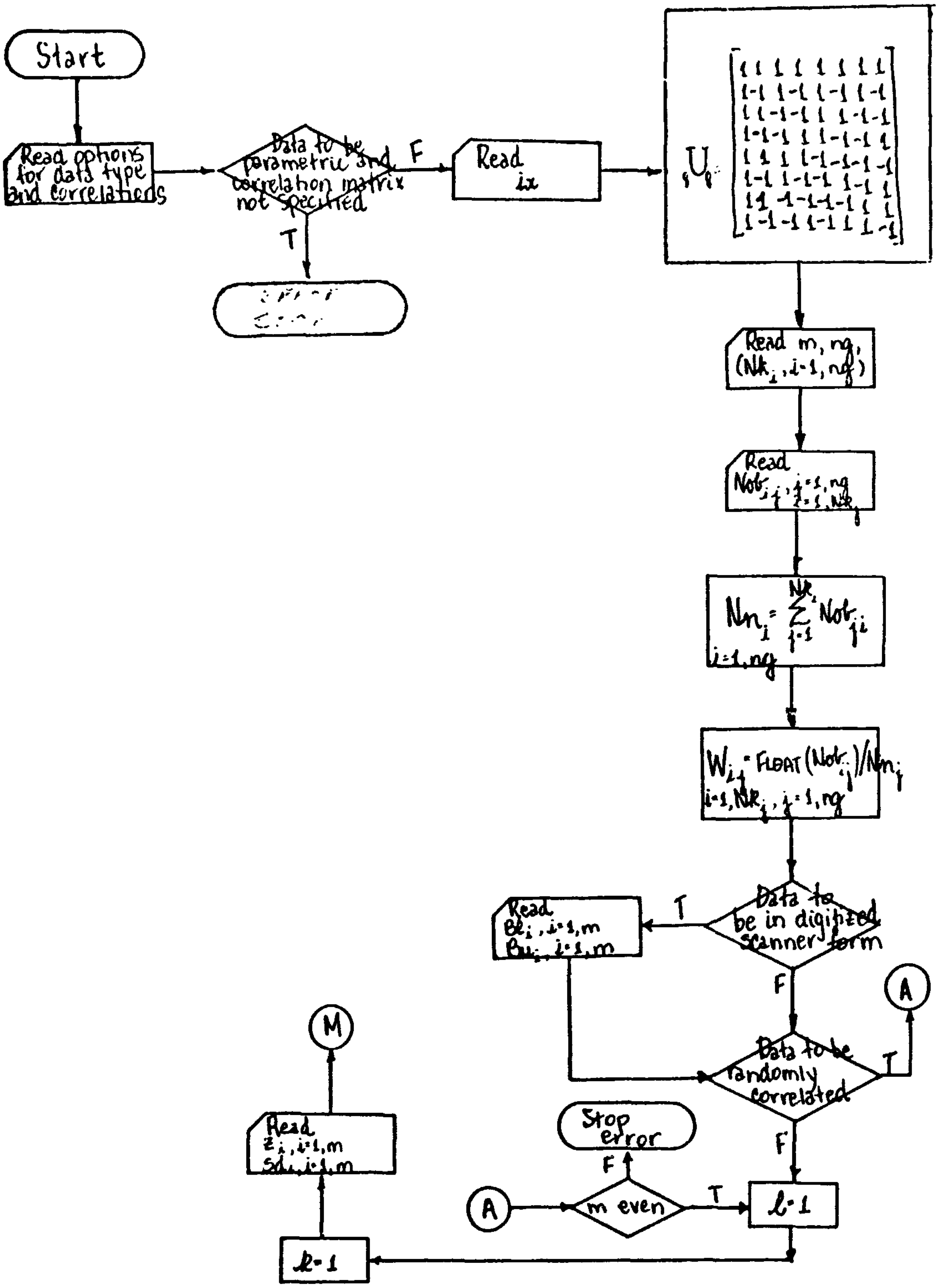

Start
Read options for data type and correlations
Data to be parametric and correlation matrix not specified
F
T
Read ix
$_{i}U_{i}$ =
1 1 1 1 1 1 1 1
1 -1 1 -1 1 -1 1 -1
1 1 -1 -1 1 1 -1 -1
1 -1 -1 1 1 -1 -1 1
1 1 1 1 -1 -1 -1 -1
1 -1 1 -1 -1 1 -1 1
1 1 -1 -1 -1 -1 1 1
1 -1 -1 1 -1 1 1 -1
Read m, ng, (Nk_i, i=1, ng)
Read Nob_{ij}, i=1, ng j=1, Nk_j
$Nn_i = \sum_{j=1}^{Nk_i} Nob_{ji}$ i=1, ng
W_{ij} = FLOAT(Nob_{ij})/Nn_j i=1, Nk_j, j=1, ng
Data to be in digitized scanner form
T
F
Read Bl_i, i=1, m Bu_i, i=1, m
A
Data to be randomly correlated
T
F
M
Read z_i, i=1, m sd_i, i=1, m
Stop error
F
A
m even
T
ℓ=1
k=1

Figure 1-1 Main Program, Data Simulation

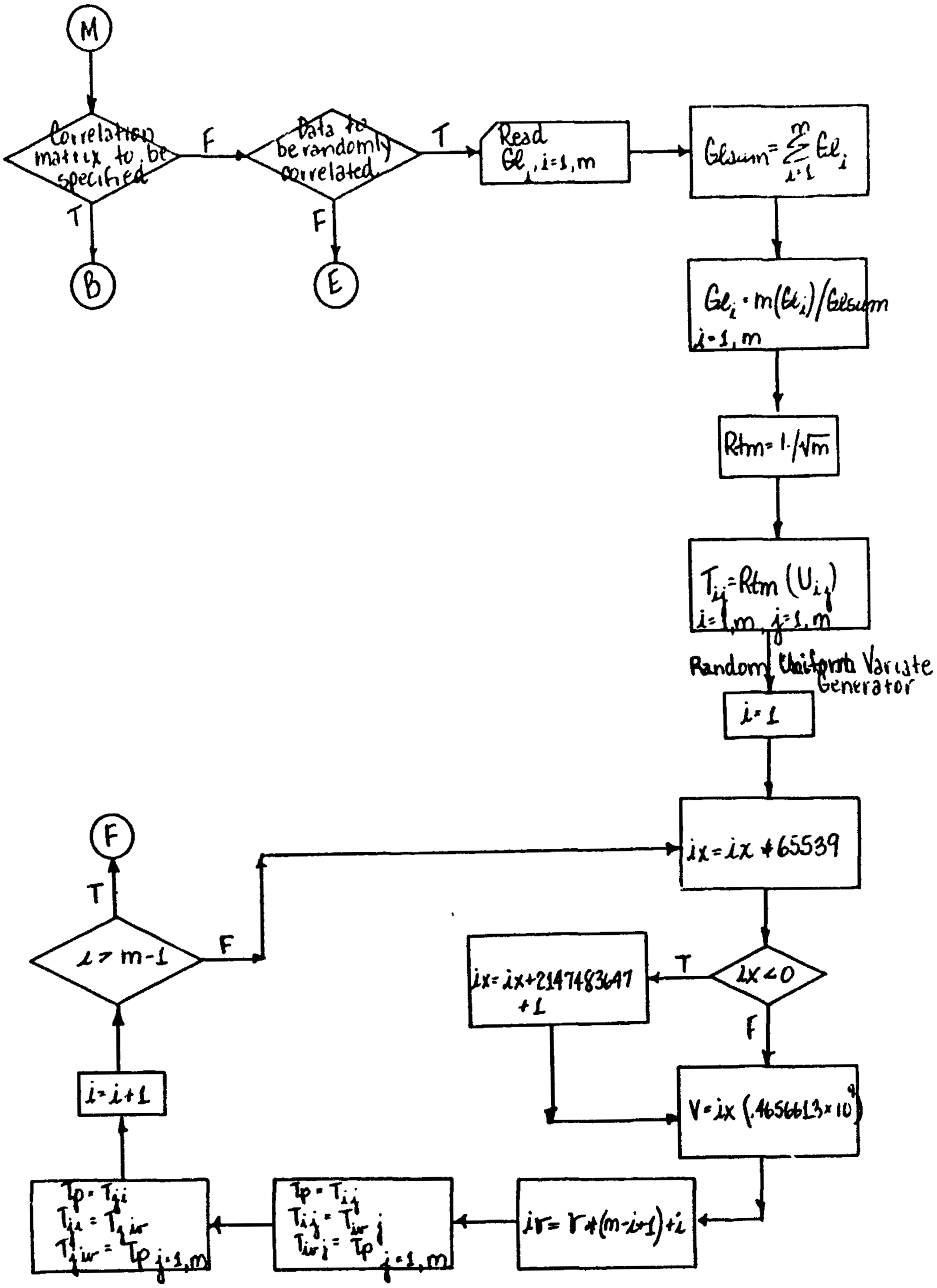


Figure 1-1 (Cont.)

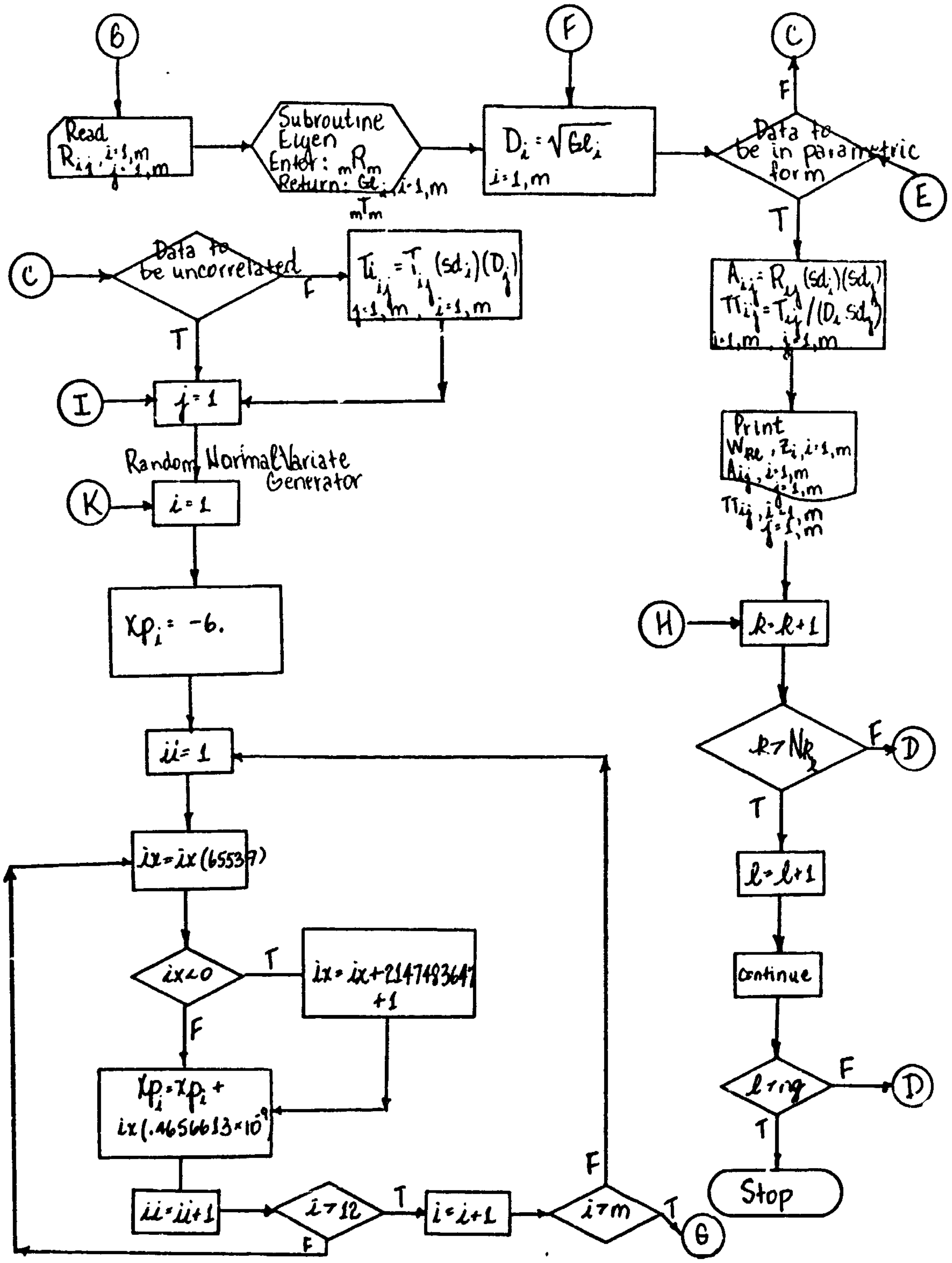


Figure 1-1 (Cont.)

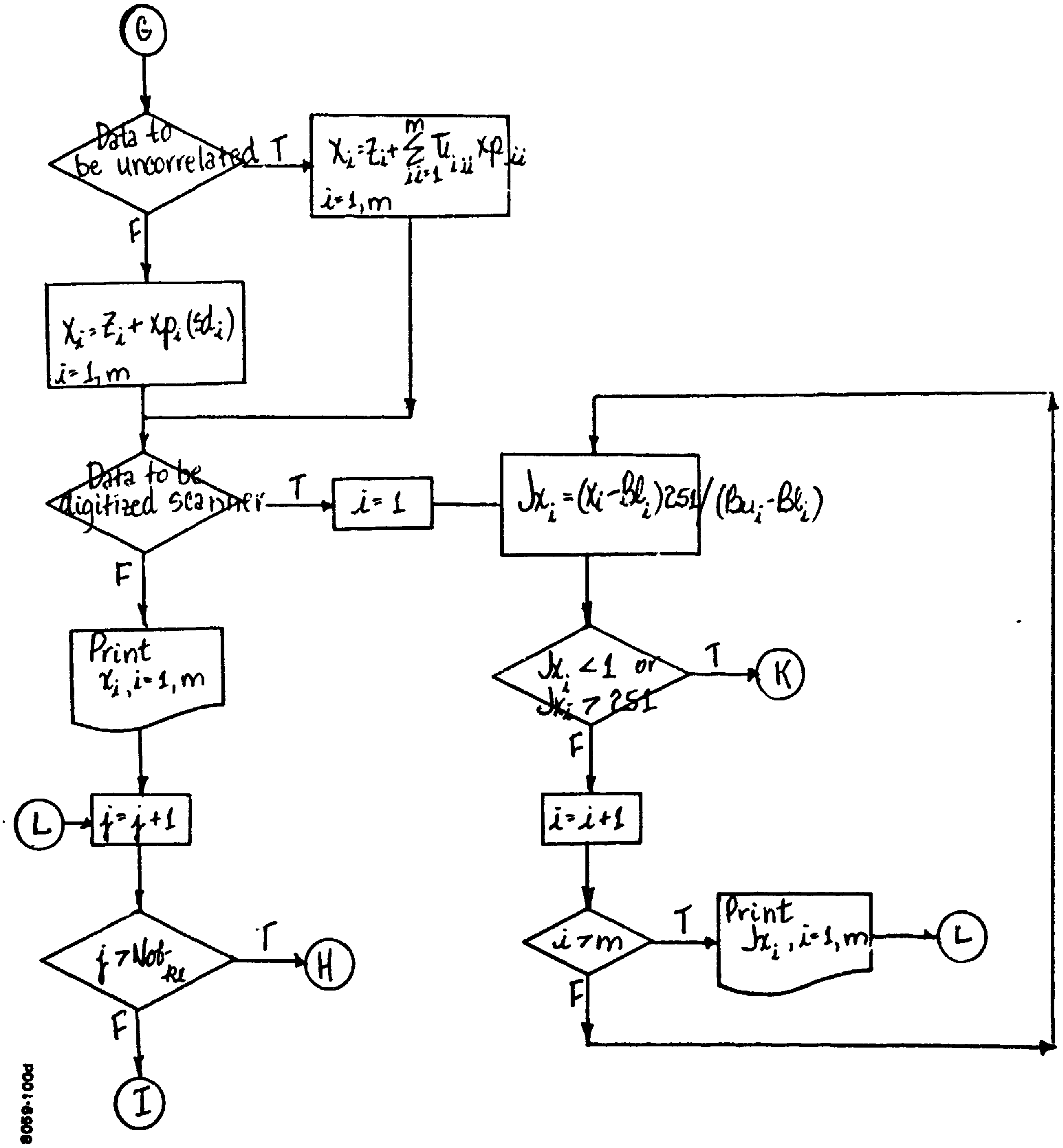


Figure 1-1 (Cont.)

1.2 FACTOR ANALYSIS PROGRAM

The Factor Analysis Program identifi principal components, thereby removing redundancy owing to correlations among variables. The factoring may be done using correlation about the mean, correlation about the origin, covariance about the mean, or covariance about the origin. The number of eigenvalues used in the analysis, and therefore, the number of principal components identified, may be limited in one of three ways. The user may specify directly the number of factors desired, the minimum eigenvalue to remain in the analysis, or the maximum percentage of total variance to be accounted for by the eigenvalues.

Subroutine CORRE reads the n observations of m-tuple raw data and from these data calculates means, standard deviations, and product moment correlation coefficients or covariance matrix, either about the mean or origin. Subroutine EIGEN computes the eigenvalues and eigenvectors of the input covariance or correlation matrix. Subroutine TRACE computes the cummulative proportion of total variance accountedfor by eigenvalues and number of principal components to be kept for analysis on the basis of one of the three limiting options. Subroutine LOAD calculates the coefficients of each factor by multiplying the elements of each normalized eigenvector by the square root of the corresponding eigenvalue.

Input may be in the form of raw data to be read by the CORRE subroutine, or means, standard deviations, eigenvalues, eigenvectors, and factor coefficients may be read by the main program. In either case, the main program handles output and computes the factor score coefficient matrix, channel coefficient settings, and the factor scores.

1.2.1 Factor Analysis Mathematical Description

The factor analysis orthogonal solution is a method of finding coordinate systems in which the variables are uncorrelated. If dependence among the variables is limited to linear (i.e., nonlinear coupling is absent), the transform might also produce statistical independence or something approaching it.

Given a data matrix ${}_mX_n$ consisting of n column vectors, each one an observation, the covariance matrix* of ${}_mX_n$ is:

$${}_mS_m = \frac{1}{n}\ {}_mX_n\ {}_nX_m^T\ .$$

*This applies for covariance about the origin. Covariance about the mean values requires that the data matrix first be properly adjusted.

Since S is a real symmetric positive definite matrix, there exists a transformation U which will transform S into a diagonal matrix D whose diagonal elements are the eigenvalues of S.

$$L = U^T SU = \begin{pmatrix} \lambda_1 0 \cdots\cdots\cdots & 0 \\ 0 \ \lambda_2 \cdots\cdots\cdots & 0 \\ \cdots\cdots\cdots\cdots\cdots & \\ 0 \ 0 \cdots\cdots\cdots\cdots & \lambda_m \end{pmatrix} \qquad (1\text{-}1)$$

The square root of a diagonal matrix can be defined:

$$\sqrt{L} \overset{d}{=} \begin{pmatrix} \sqrt{\lambda_1} \ \ 0 \cdots\cdots & 0 \\ 0 \ \ \sqrt{\lambda_2} \cdots\cdots & 0 \\ \cdots\cdots\cdots\cdots\cdots & \\ \cdots\cdots\cdots\cdots\cdots & \\ 0 \ 0 \cdots\cdots\cdots & \sqrt{\lambda_m} \end{pmatrix} \qquad (1\text{-}2)$$

So that

$$\sqrt{L}\,\sqrt{L} = U^T SU$$

and (1-3)

$$\sqrt{L}^{-1} U^T SU \sqrt{L}^{-1} = I$$

Substituting for S,

$$\sqrt{L}^{-1} U^T X X^T U \sqrt{L}^{-1} = nI \qquad (1\text{-}4)$$

Define A, the factor score coefficient matrix, and F, the factor score matrix:

$$A = \sqrt{L}^{-1} U^T$$

$$F = A X .$$

It follows from these definitions and Equation 1-4, that $FF^T = nI$; i.e., the factor scores are uncorrelated.

If A is nonsingular, its inverse exists and X can be computed from F:

$$X = A^{-1}F \ .$$

If the inverse of A is not defined because A is not square:

$$AX = F$$

$$A^T AX = A^T F$$

$$X = (A^T A)^{-1} A^T F \ .$$

The correlation matrix of ${}_mX_n$,

$${}_mR_m = \sqrt{D}^{-1} \quad S\sqrt{D}^{-1} \quad \text{, where}$$

$$D = \begin{pmatrix} \sigma_1^2 & \cdots\cdots\cdots\cdots & 0 \\ 0 & \sigma_2^2 \cdots\cdots\cdots & 0 \\ \cdot & & \cdot \\ \cdot & & \cdot \\ \cdot & & \cdot \\ 0 & 0 \cdots\cdots\cdots & \sigma_m^2 \end{pmatrix} ,$$

may be diagonalized during the analysis in place of the covariance matrix, S. If analysis is done about the sample mean rather than the origin, ${}_mY_n$, n column vectors of the form $Y_j = X_j - \overline{X}$, j = 1, n, are used in the place of the vectors forming ${}_mX_n$.

1.2.2 Flow Chart of the Factor Analysis Program

The following generalized flow diagram (Figure 1-2) of the factor analysis program is supplemented by detailed Fortran IV statement diagrams of subroutines CORRE, EIGEN, TRACE, and LOAD. (Figures 1-3 to 1-6).

KEY TO NOTATION

Main Program, Principal Component Factor Analysis

n	=	number of observations
k	=	number of factors
m	=	number of variables
$\bar{X}_i$, $i=1, m$	=	mean of ith variable
S_i, $i=1, m$	=	standard deviation of ith variable
${}_mX_n$	=	raw data matrix
λ_j, $j=1, k$	=	jth eigenvalue
${}_kU_m$	=	row-wise eignevectors
d_j, $j=1, k$	=	cumulative proportion of total variance
${}_mB_k$	=	factor matrix for standard data
${}_mC_k$	=	factor matrix for nonstandard data
${}_kA_m$	=	factor score coefficient matrix for standard data
${}_kE_m$	=	factor score coefficient matrix for nonstandard data
${}_kG_m$	=	channel coefficient settings matrix
${}_kF_n$	=	factor scores
$\bar{X}_{f_i}$, $i=1, k$	=	factor score means
${}_mR_m$	=	correlation or covariance matrix.

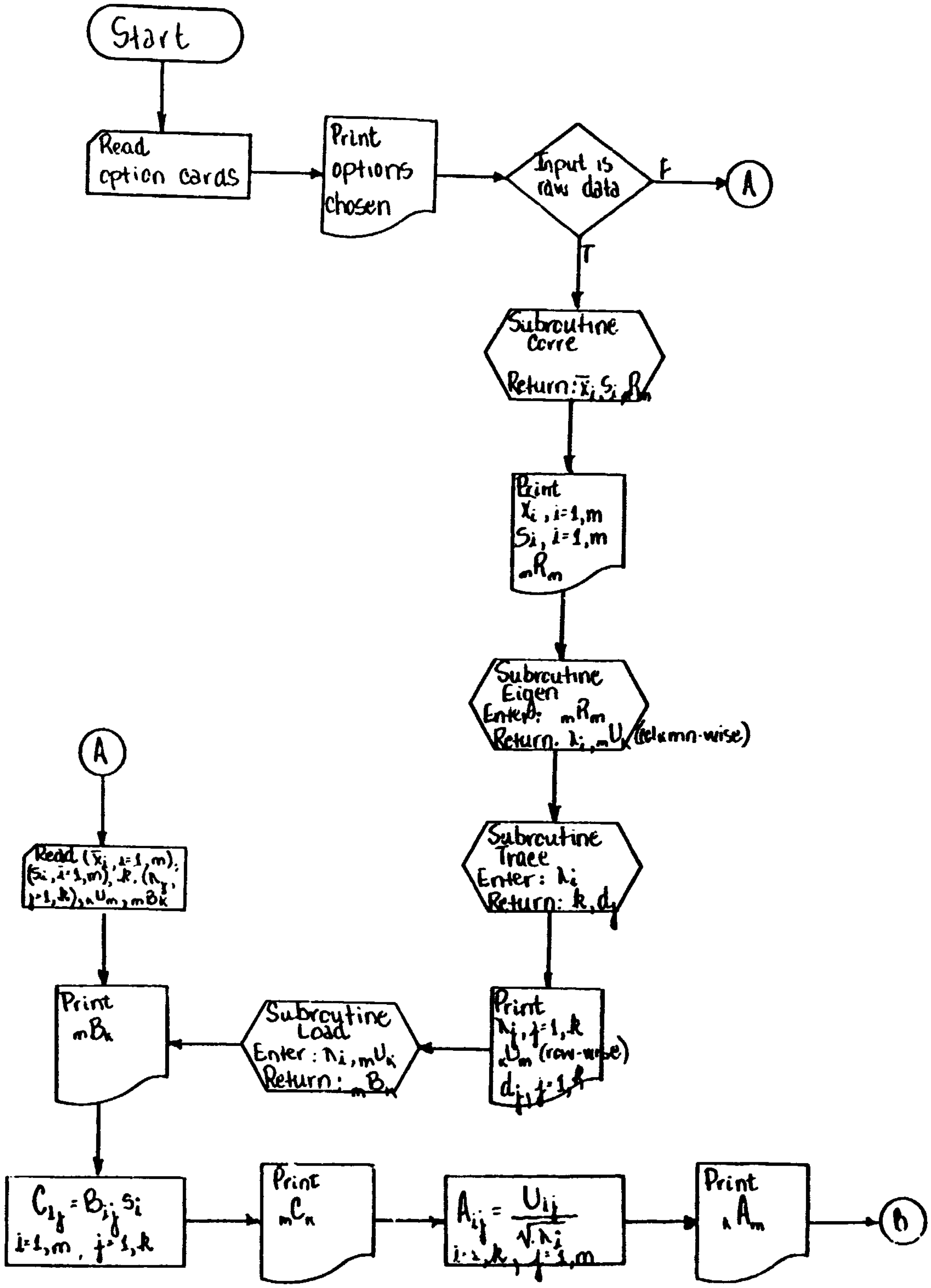

Start
Read option cards
Print options chosen
Input is raw data
F
T
A
Subroutine Corre Return: $\bar{x}_i, S_i, {}_mR_m$
Print $\bar{x}_i$, $i=1,m$ S_i, $i=1,m$ ${}_mR_m$
Subroutine Eigen Enter: ${}_mR_m$ Return: $\lambda_i, {}_mU_k$ (column-wise)
Subroutine Trace Enter: λ_i Return: k, d_j
Print $\lambda_j, j=1,k$ ${}_kU_m$ (row-wise) $d_j, j=1,k$
Subroutine Load Enter: $\lambda_i, {}_mU_k$ Return: ${}_mB_k$
A
Read $(\bar{x}_i, i=1,m)$; $(S_i, i=1,m)$, k, $(\lambda_j, j=1,k)$, ${}_kU_m$, ${}_mB_k$
Print ${}_mB_k$
$C_{ij} = B_{ij} \cdot S_i$ $i=1,m$, $j=1,k$
Print ${}_mC_k$
$A_{ij} = \frac{U_{ij}}{\sqrt{\lambda_i}}$ $i=1,k$, $j=1,m$
Print ${}_kA_m$
B

Figure 1-2 Main Program, Principle Component Factor Analysis

8059-101e

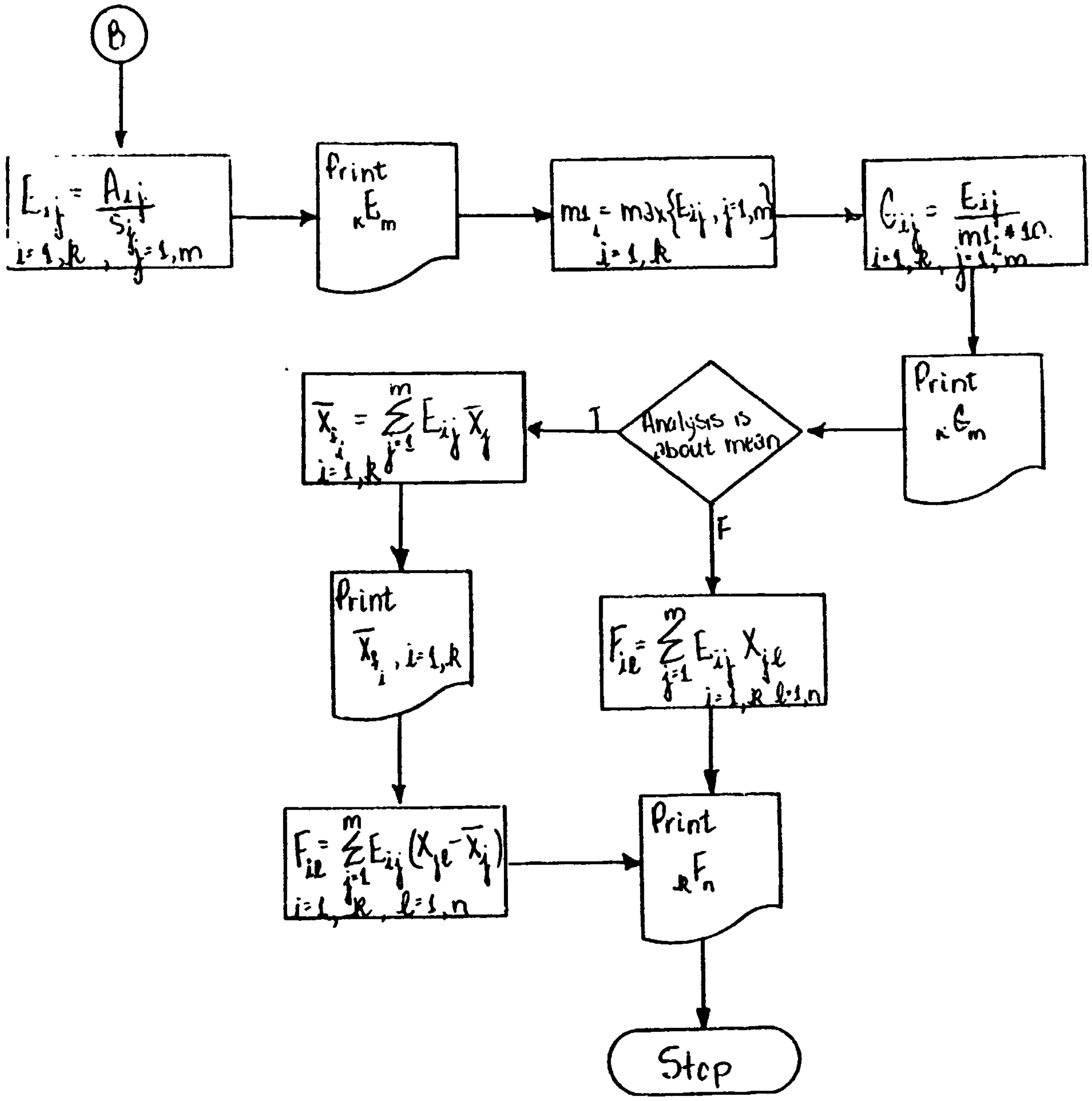
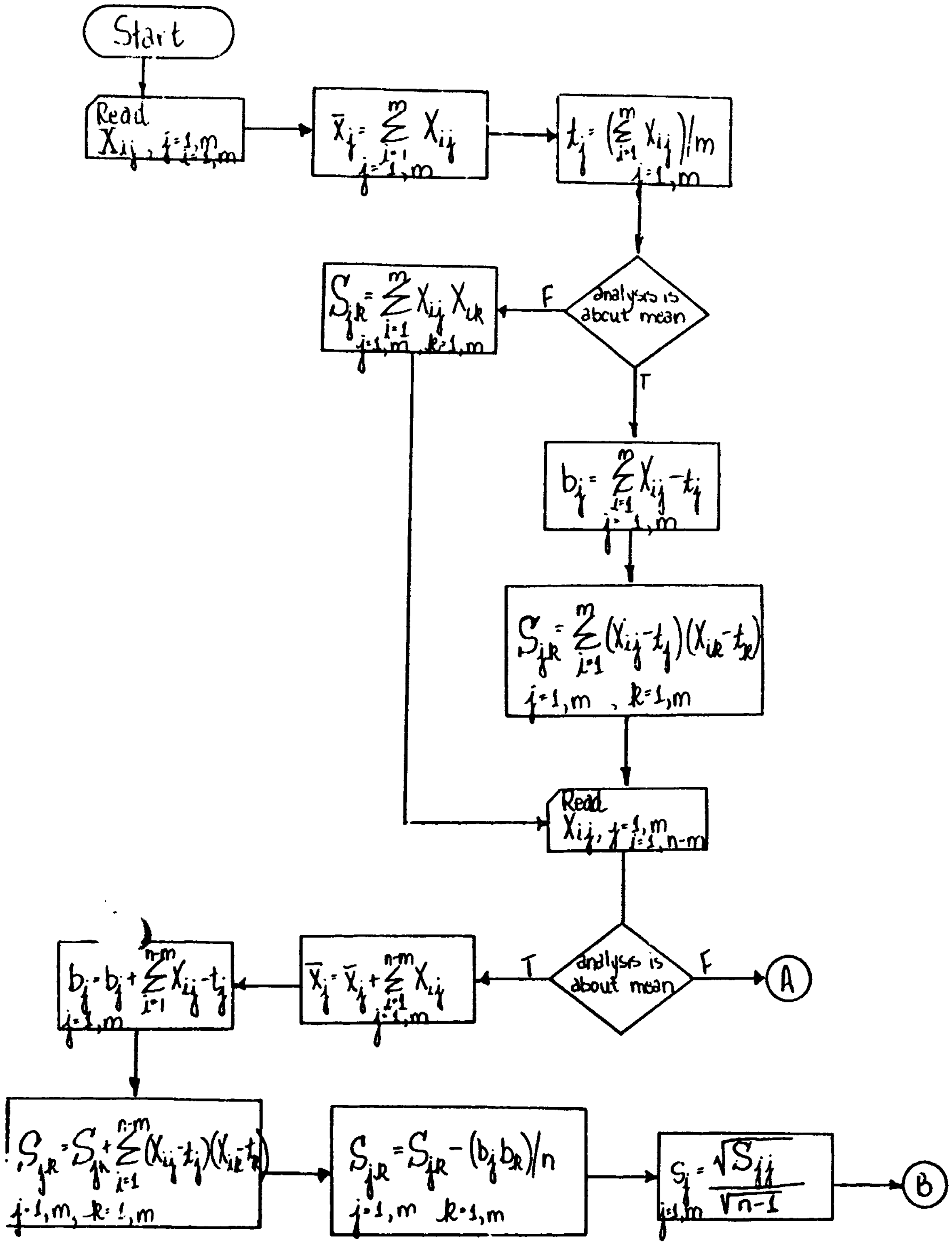

B
$E_{ij} = \frac{A_{ij}}{S_j}$
$i=1,k \quad , \quad j=1,m$
Print
$_kE_m$
$m1_i = \max\{E_{ij}, j=1,m\}$
$i=1,k$
$G_{ij} = \frac{E_{ij}}{m1_i * 10.}$
$i=1,k \quad , \quad j=1,m$
Print
$_kG_m$
Analysis is about mean
T
F
$\bar{X}_{s_i} = \sum_{j=1}^{m} E_{ij} \bar{X}_j$
$i=1,k$
Print
$\bar{X}_{s_i} \; , i=1,k$
$F_{il} = \sum_{j=1}^{m} E_{ij} X_{jl}$
$i=1,k \quad l=1,n$
$F_{il} = \sum_{j=1}^{m} E_{ij} (X_{jl} - \bar{X}_j)$
$i=1,k \quad , \quad l=1,n$
Print
$_kF_n$
Stop

Figure 1-2 (Cont.)

KEY TO NOTATION

Subroutine CORRE

${}_{n}X_{m}$	=	raw data
n	=	number of observations
m	=	number of variables
$\bar{X}_i$, $i=1, m$	=	mean of i^{th} variable
S_i, $i=1, m$	=	standard deviation of i^{th} variable
${}_{m}R_{m}$	=	covariance or correlation matrix.


Start
Read X_{ij}, $j=1,m$ $i=1,m$
$\bar{X}_j = \sum_{i=1}^{m} X_{ij}$ $j=1,m$
$t_j = (\sum_{i=1}^{m} X_{ij})/m$ $j=1,m$
analysis is about mean
F
T
$S_{jk} = \sum_{i=1}^{m} X_{ij} X_{ik}$ $j=1,m$, $k=1,m$
$b_j = \sum_{i=1}^{m} X_{ij} - t_j$ $j=1,m$
$S_{jk} = \sum_{i=1}^{m} (X_{ij}-t_j)(X_{ik}-t_k)$ $j=1,m$, $k=1,m$
Read X_{ij}, $j=1,m$ $i=1,n-m$
analysis is about mean
T
F
A
$\bar{X}_j = \bar{X}_j + \sum_{i=1}^{n-m} X_{ij}$ $j=1,m$
$b_j = b_j + \sum_{i=1}^{n-m} X_{ij} - t_j$ $j=1,m$
$S_{jk} = S_{jk} + \sum_{i=1}^{n-m} (X_{ij}-t_j)(X_{ik}-t_k)$ $j=1,m$, $k=1,m$
$S_{jk} = S_{jk} - (b_j b_k)/n$ $j=1,m$ $k=1,m$
$S_j = \frac{\sqrt{S_{jj}}}{\sqrt{n-1}}$ $j=1,m$
B

8069 102a

Figure 1-3 Subroutine CORRE

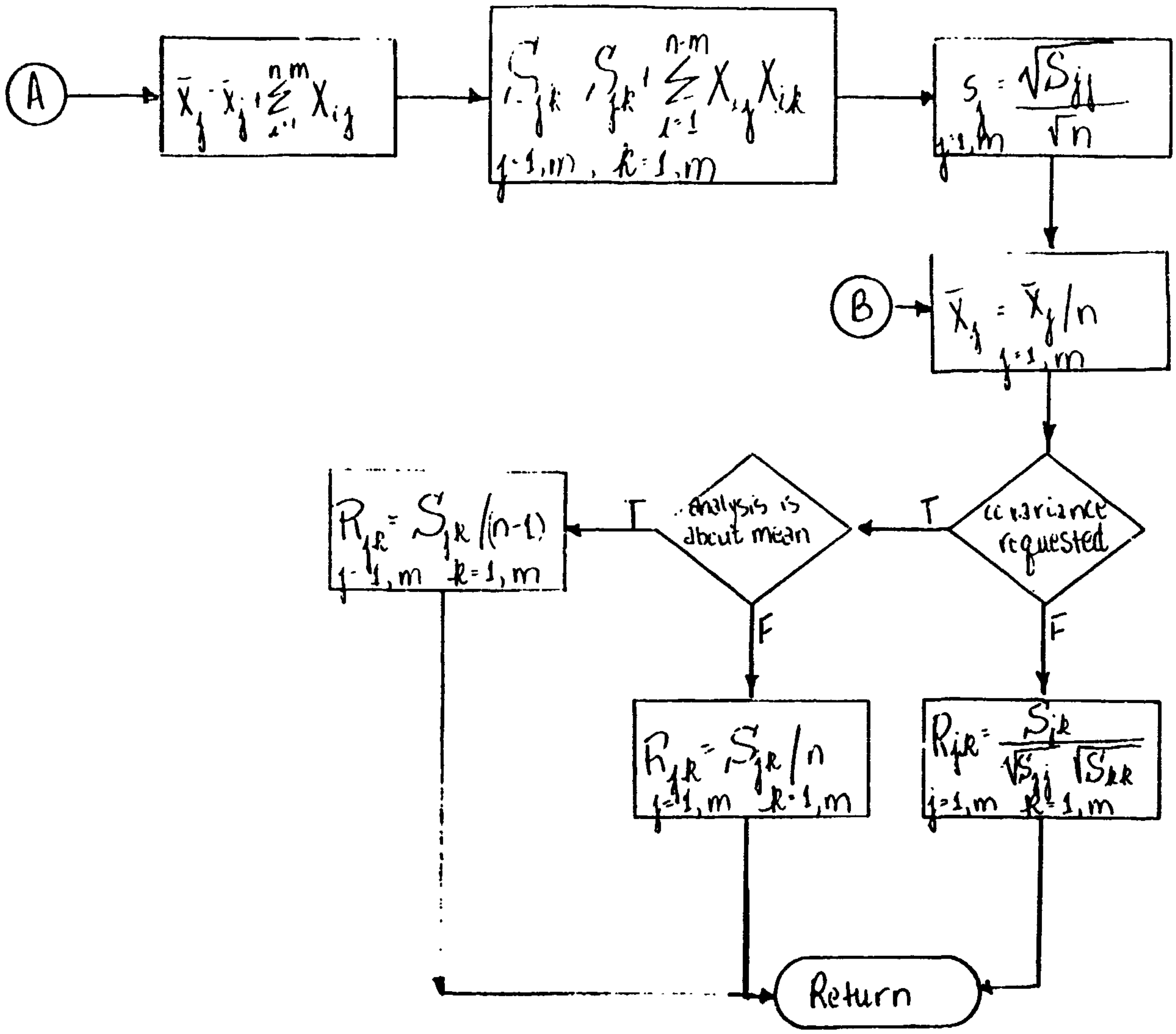
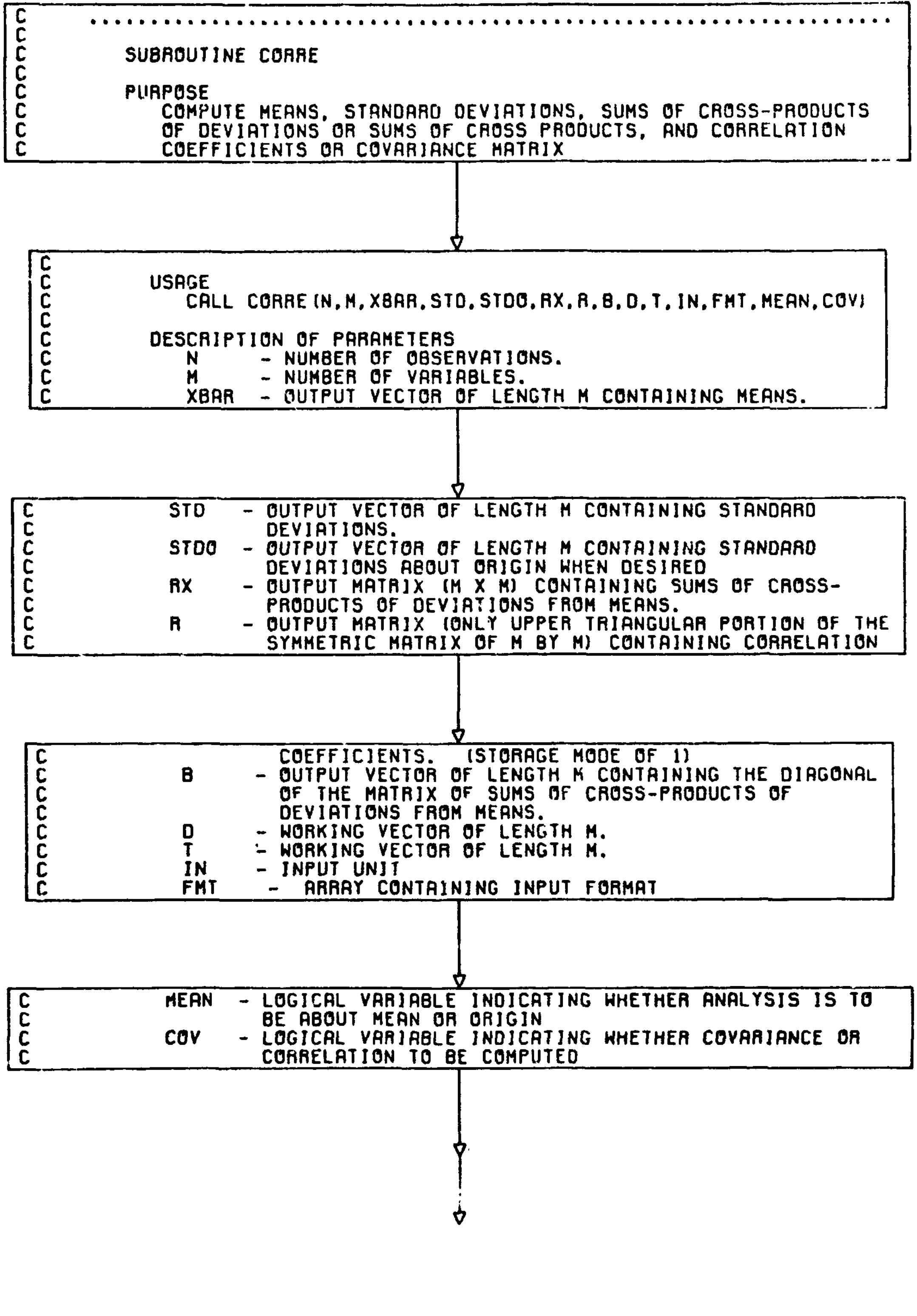

A
$\bar{X}_j = \bar{X}_j + \sum_{i=1}^{n\cdot m} X_{ij}$
$S_{jk} = S_{jk} + \sum_{i=1}^{n\cdot m} X_{ij} X_{ik}$
$j=1,m \;,\; k=1,m$
$s_j = \frac{\sqrt{S_{jj}}}{\sqrt{n}}$
$j=1,m$
B
$\bar{X}_j = \bar{X}_j / n$
$j=1,m$
covariance requested
T
F
analysis is about mean
T
F
$R_{jk} = S_{jk} / (n-1)$
$j=1,m \quad k=1,m$
$R_{jk} = S_{jk} / n$
$j=1,m \quad k=1,m$
$R_{jk} = \frac{S_{jk}}{\sqrt{S_{jj}}\,\sqrt{S_{kk}}}$
$j=1,m \quad k=1,m$
Return

Figure 1-3 (Cont.)



```
C     ..............................................................
C
C        SUBROUTINE CORRE
C
C        PURPOSE
C           COMPUTE MEANS, STANDARD DEVIATIONS, SUMS OF CROSS-PRODUCTS
C           OF DEVIATIONS OR SUMS OF CROSS PRODUCTS, AND CORRELATION
C           COEFFICIENTS OR COVARIANCE MATRIX
```

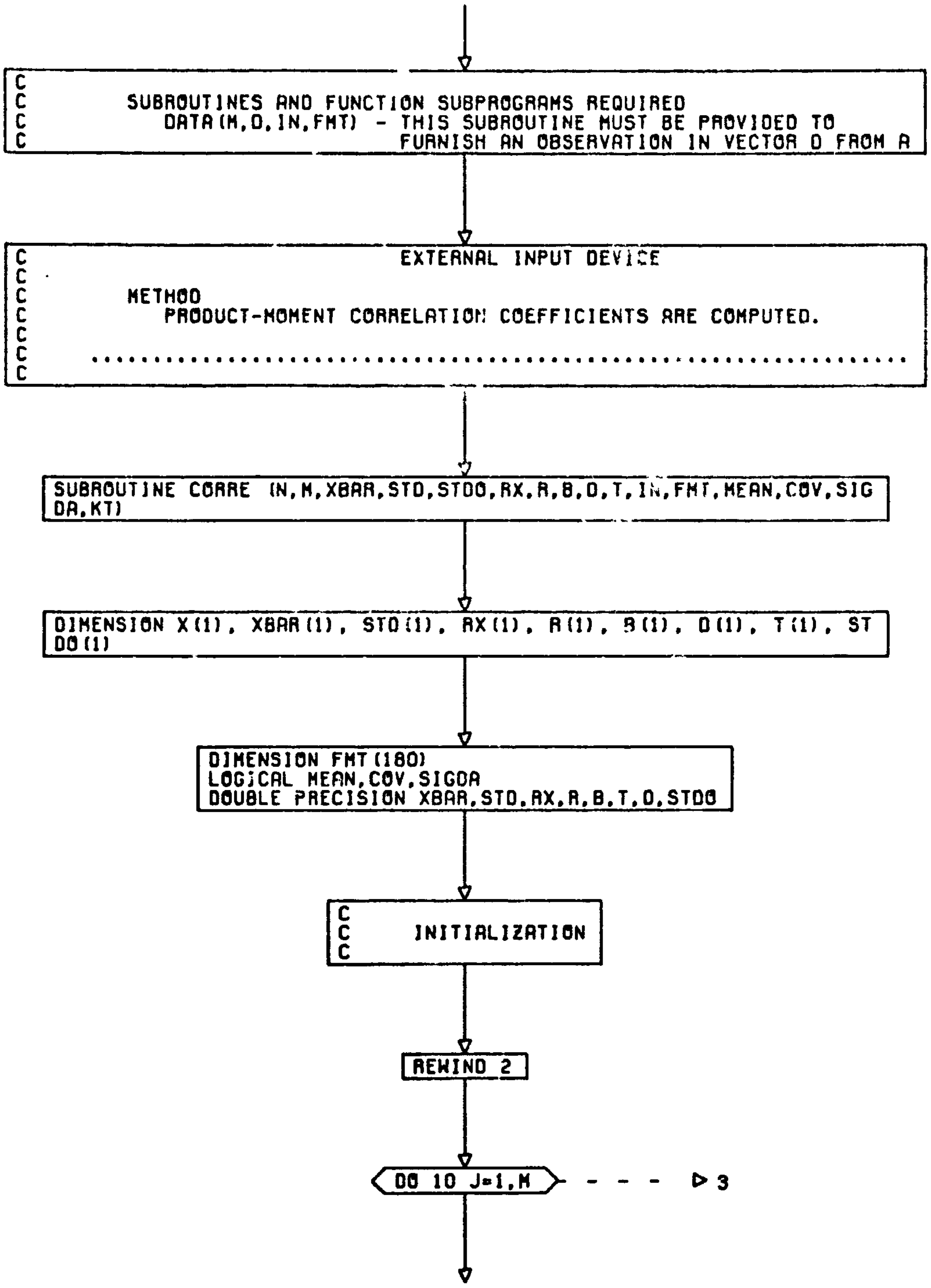
```
C
C        USAGE
C           CALL CORRE(N,M,XBAR,STD,STDO,RX,R,B,D,T,IN,FMT,MEAN,COV)
C
C        DESCRIPTION OF PARAMETERS
C           N      - NUMBER OF OBSERVATIONS.
C           M      - NUMBER OF VARIABLES.
C           XBAR   - OUTPUT VECTOR OF LENGTH M CONTAINING MEANS.
```

```
C           STD    - OUTPUT VECTOR OF LENGTH M CONTAINING STANDARD
C                    DEVIATIONS.
C           STDO   - OUTPUT VECTOR OF LENGTH M CONTAINING STANDARD
C                    DEVIATIONS ABOUT ORIGIN WHEN DESIRED
C           RX     - OUTPUT MATRIX (M X M) CONTAINING SUMS OF CROSS-
C                    PRODUCTS OF DEVIATIONS FROM MEANS.
C           R      - OUTPUT MATRIX (ONLY UPPER TRIANGULAR PORTION OF THE
C                    SYMMETRIC MATRIX OF M BY M) CONTAINING CORRELATION
```

```
C                    COEFFICIENTS.  (STORAGE MODE OF 1)
C           B      - OUTPUT VECTOR OF LENGTH M CONTAINING THE DIAGONAL
C                    OF THE MATRIX OF SUMS OF CROSS-PRODUCTS OF
C                    DEVIATIONS FROM MEANS.
C           D      - WORKING VECTOR OF LENGTH M.
C           T      - WORKING VECTOR OF LENGTH M.
C           IN     - INPUT UNIT
C           FMT     -  ARRAY CONTAINING INPUT FORMAT
```

```
C           MEAN   - LOGICAL VARIABLE INDICATING WHETHER ANALYSIS IS TO
C                    BE ABOUT MEAN OR ORIGIN
C           COV    - LOGICAL VARIABLE INDICATING WHETHER COVARIANCE OR
C                    CORRELATION TO BE COMPUTED
```

Figure 1-3 (Cont.)



```
C
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C           DATA(M,D,IN,FMT) - THIS SUBROUTINE MUST BE PROVIDED TO
C                              FURNISH AN OBSERVATION IN VECTOR D FROM A
```

```
C                              EXTERNAL INPUT DEVICE
C
C        METHOD
C           PRODUCT-MOMENT CORRELATION COEFFICIENTS ARE COMPUTED.
C
C
C     ..................................................................
C
```

```
SUBROUTINE CORRE (N,M,XBAR,STD,STDD,RX,R,B,D,T,IN,FMT,MEAN,COV,SIG
DA,KT)
```

```
DIMENSION X(1), XBAR(1), STD(1), RX(1), R(1), B(1), D(1), T(1), ST
DD(1)
```

```
DIMENSION FMT(180)
LOGICAL MEAN,COV,SIGDA
DOUBLE PRECISION XBAR,STD,RX,R,B,T,D,STDD
```

```
C
C     INITIALIZATION
C
```

```
REWIND 2
```

```
DO 10 J=1,M - - - ▷ 3
```

Figure 1-3 (Cont.)

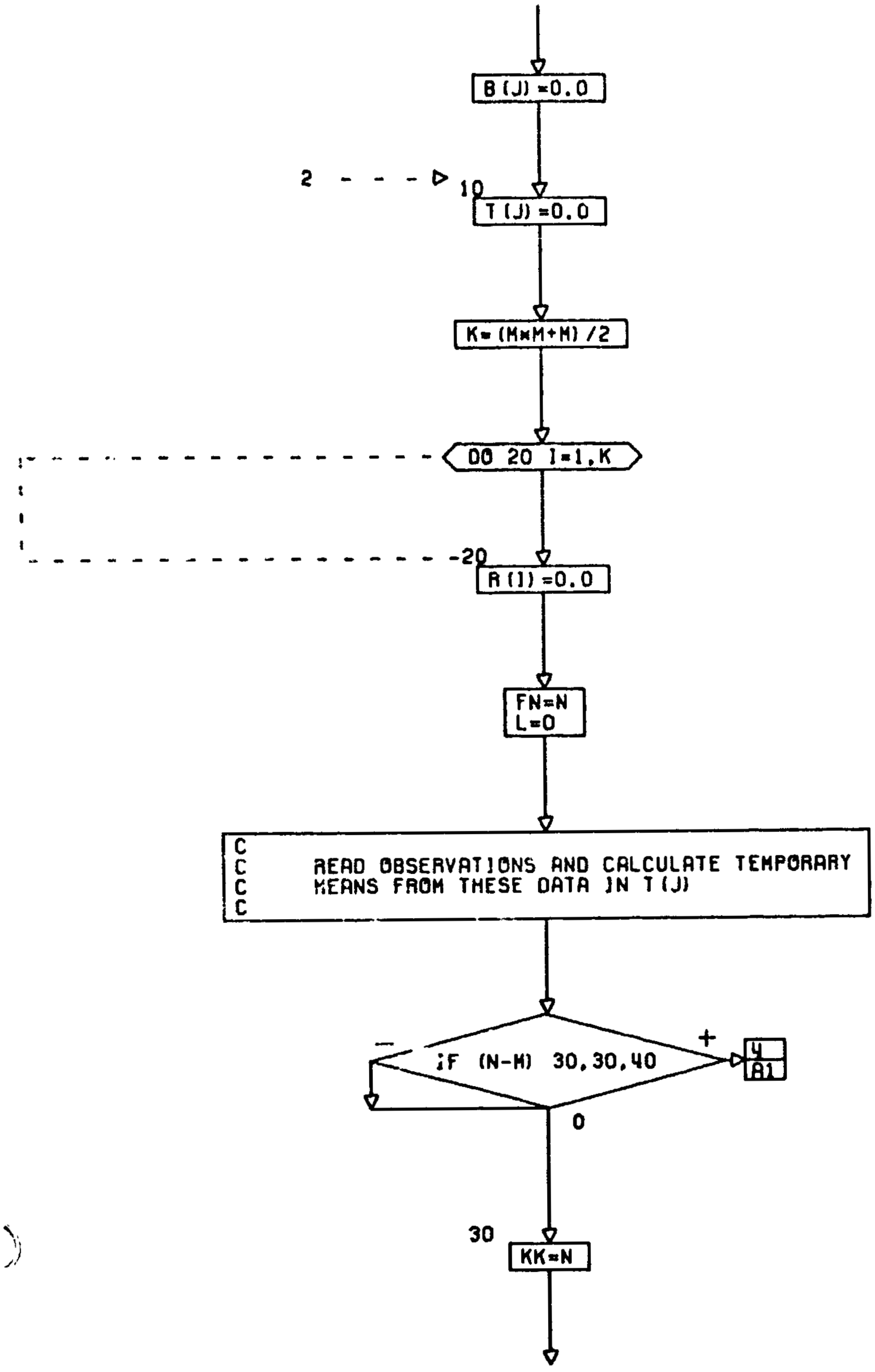


Figure 1-3 (Cont.)

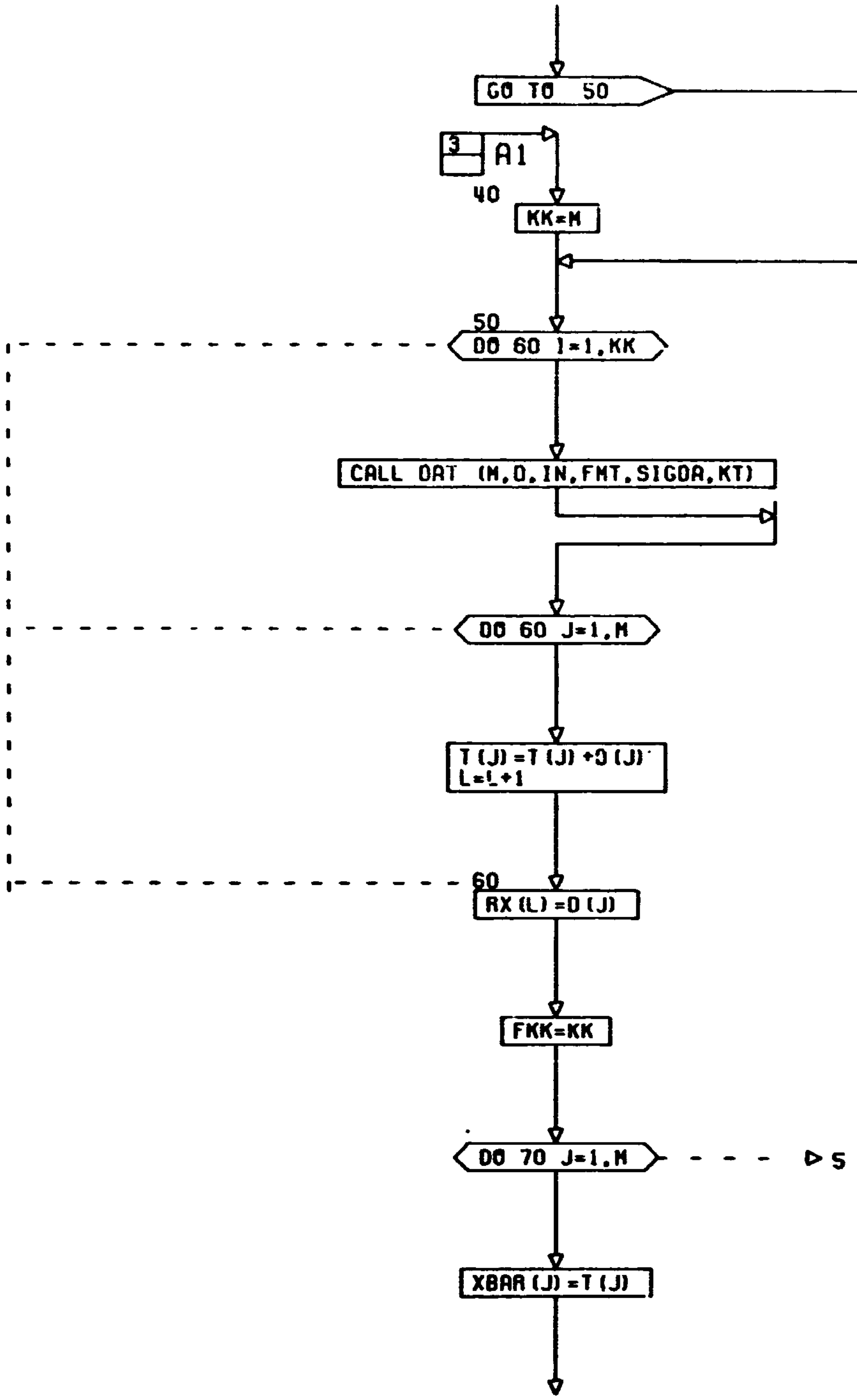
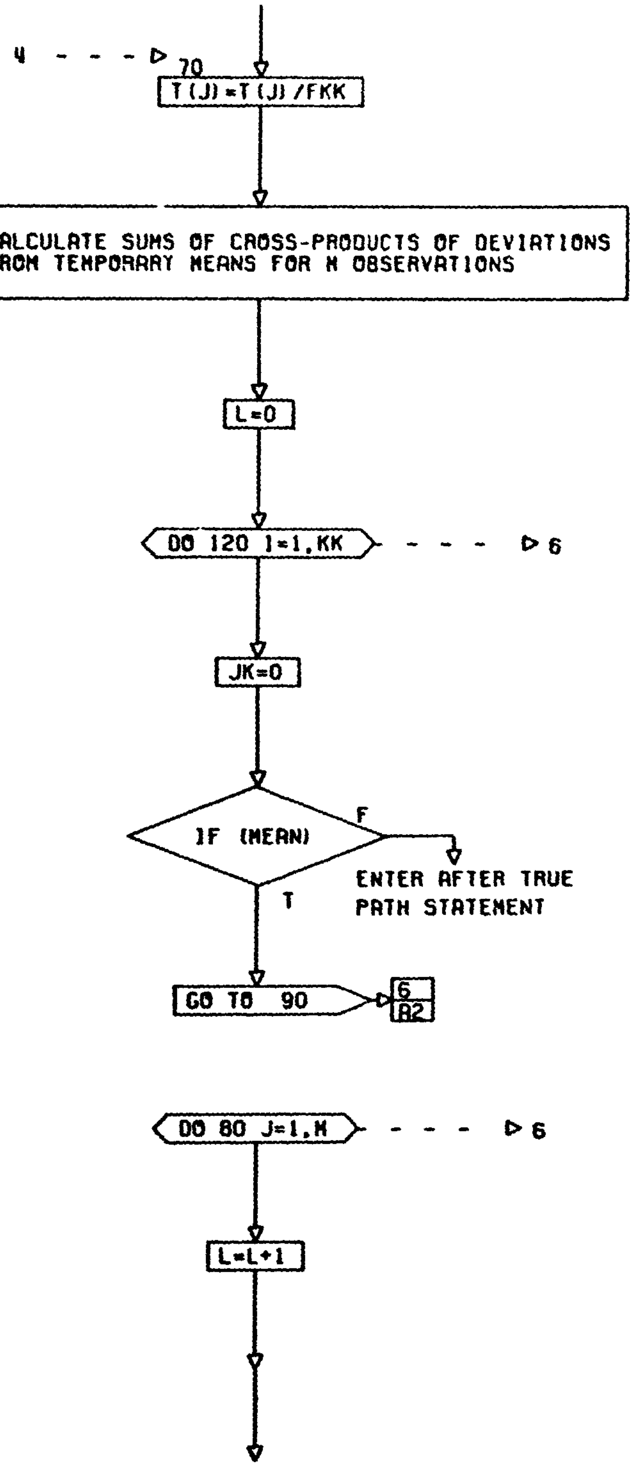
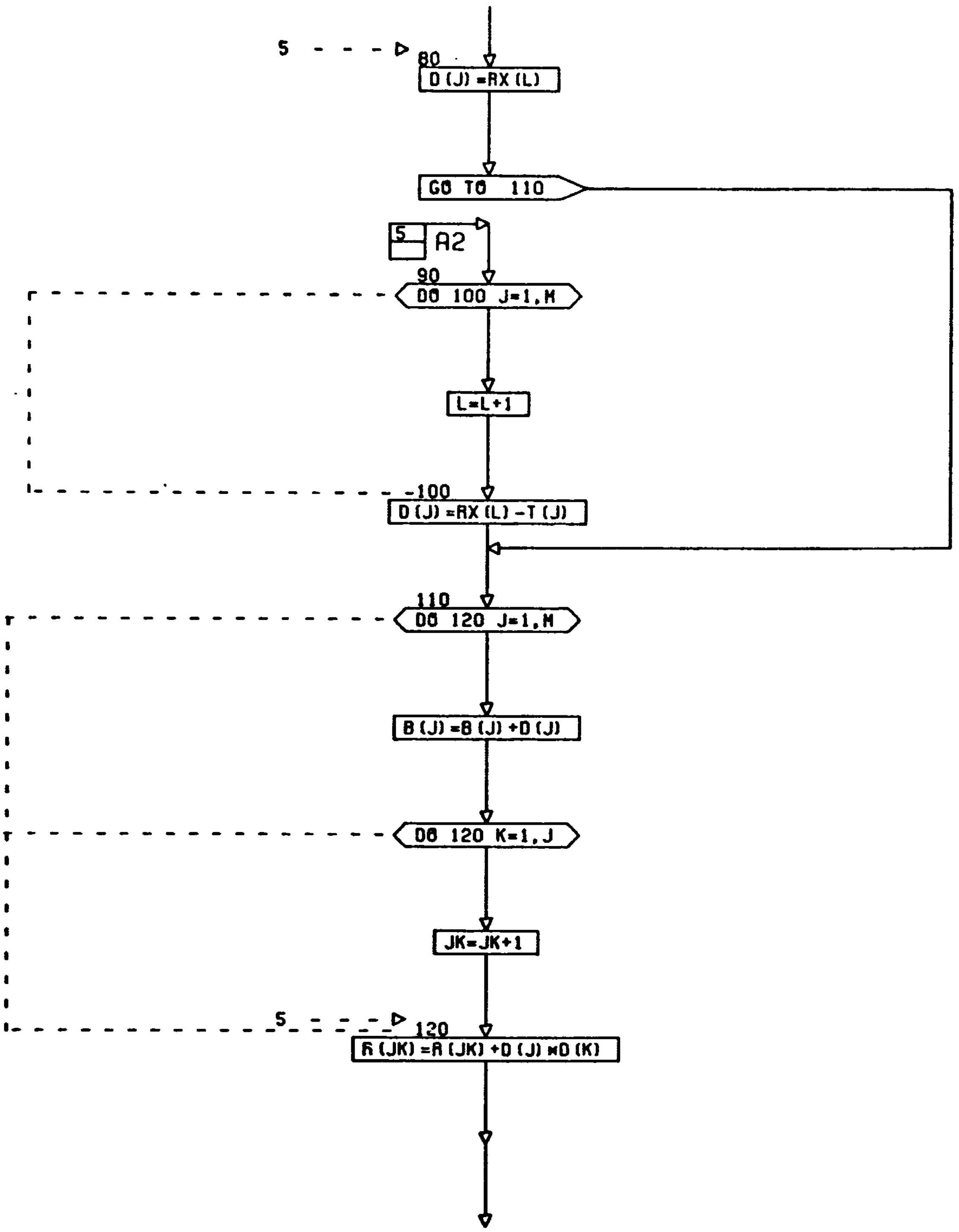


Figure 1-3 (Cont.)



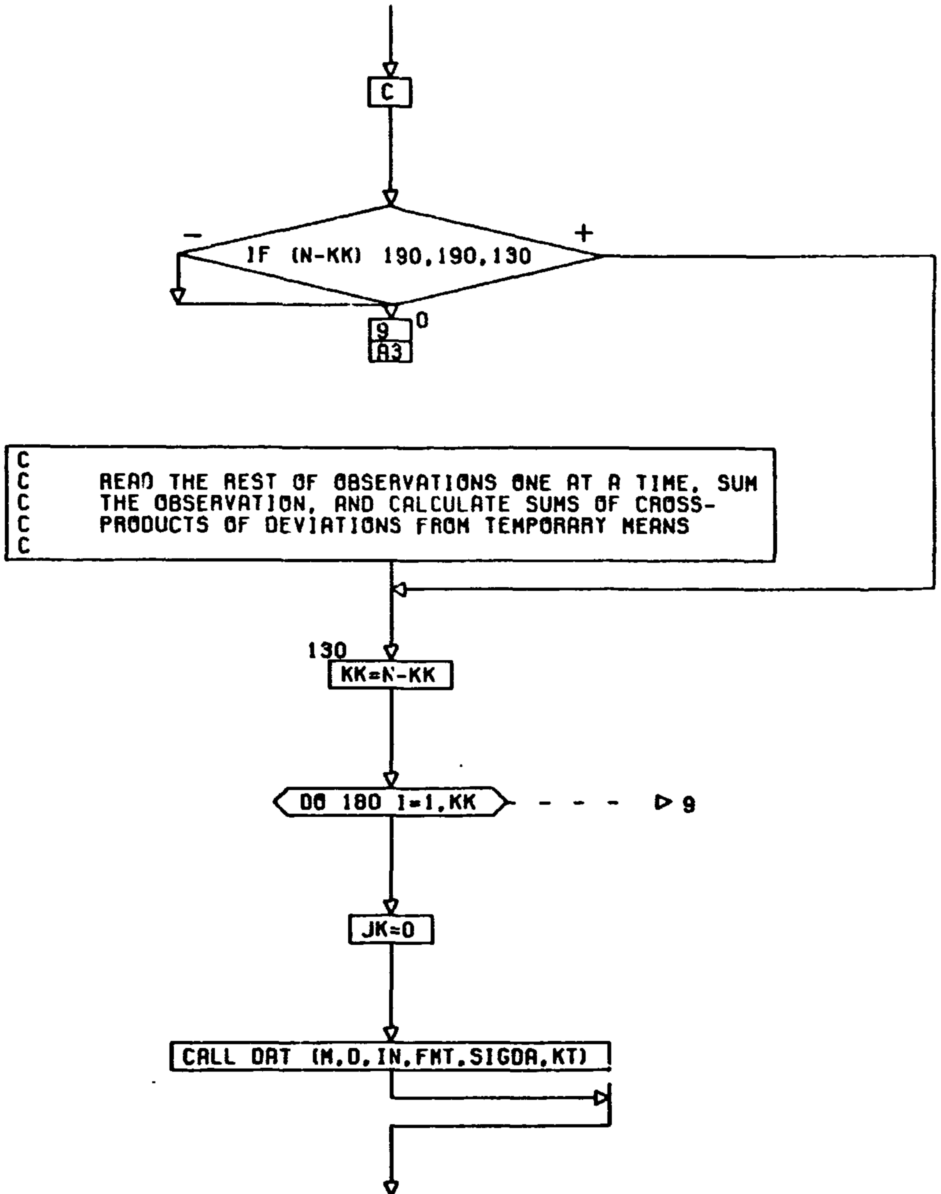
8059-102g

Figure 1-3 (Cont.)



8059-102h

Figure 1-3 (Cont.)


C
IF (N-KK) 190,190,130
−
+
0
9
A3
C
C READ THE REST OF OBSERVATIONS ONE AT A TIME, SUM
C THE OBSERVATION, AND CALCULATE SUMS OF CROSS-
C PRODUCTS OF DEVIATIONS FROM TEMPORARY MEANS
C
130
KK=N-KK
DO 180 I=1,KK
9
JK=0
CALL DAT (M,D,IN,FMT,SIGDA,KT)
IF (MEAN)
F
ENTER AFTER TRUE
PATH STATEMENT
T

8059-1021

Figure 1-3 (Cont.)

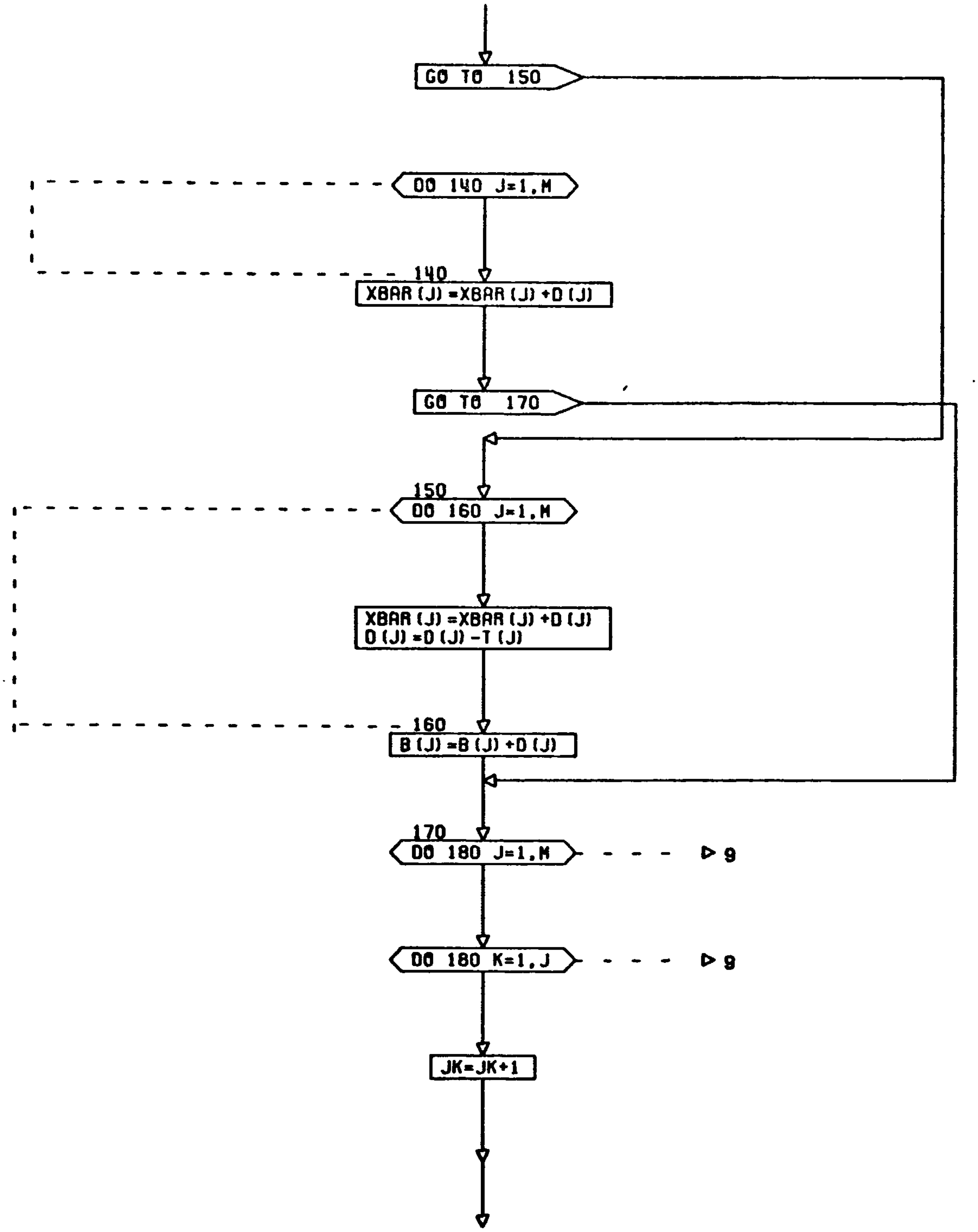


Figure 1-3 (Cont.)

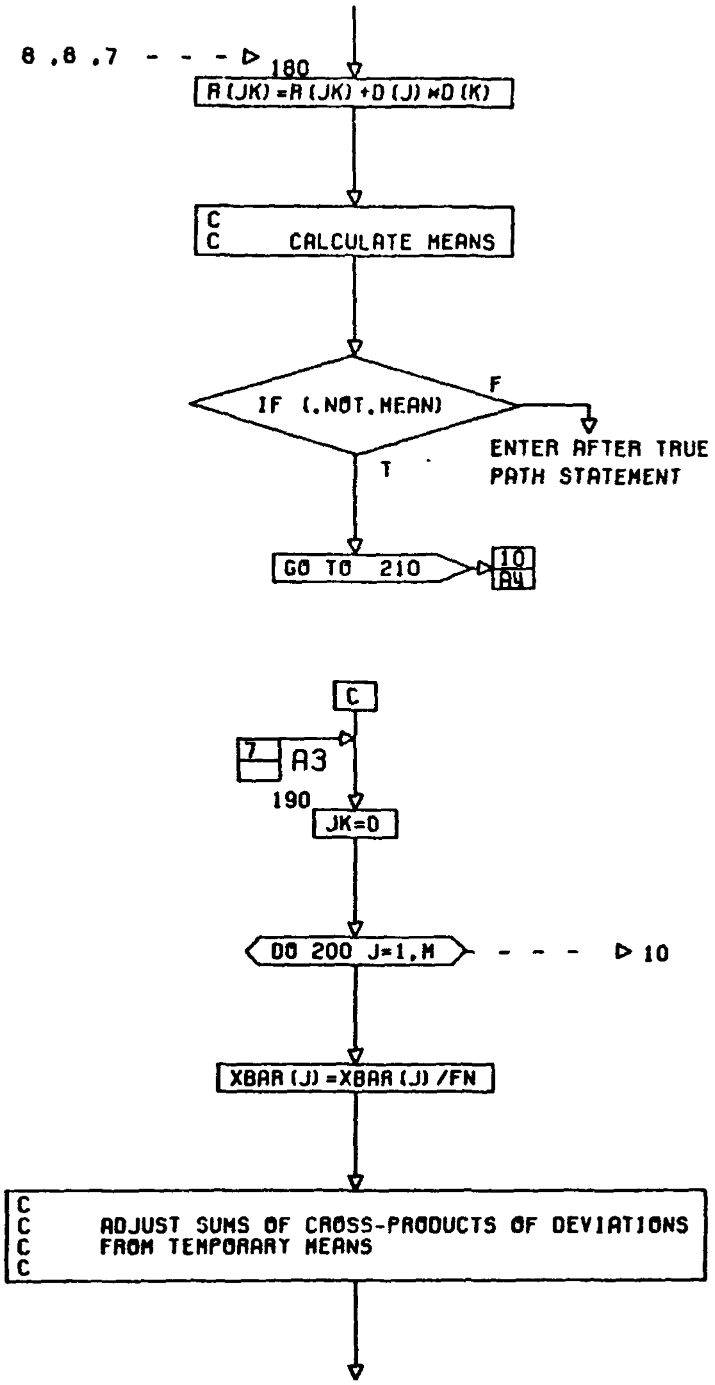
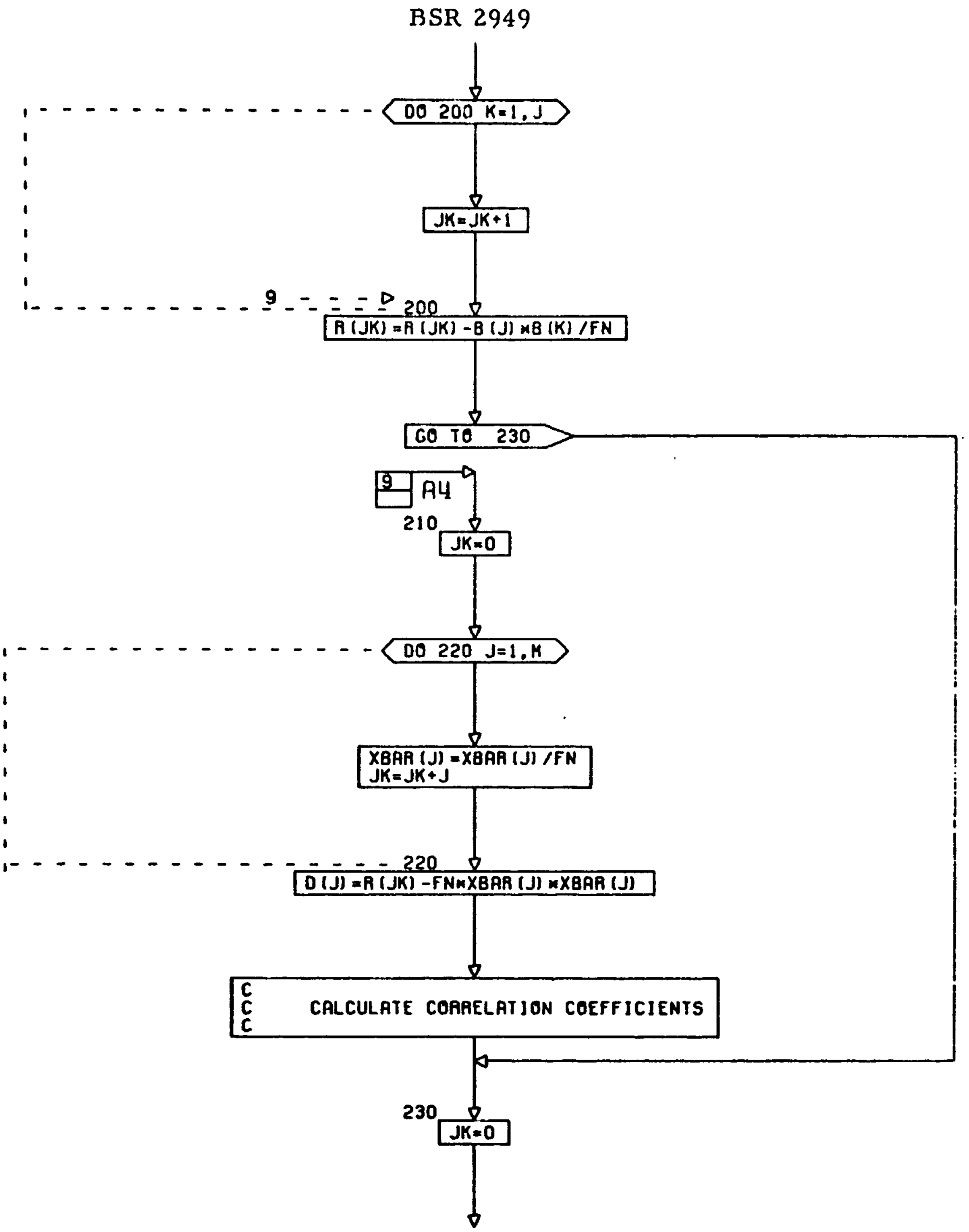
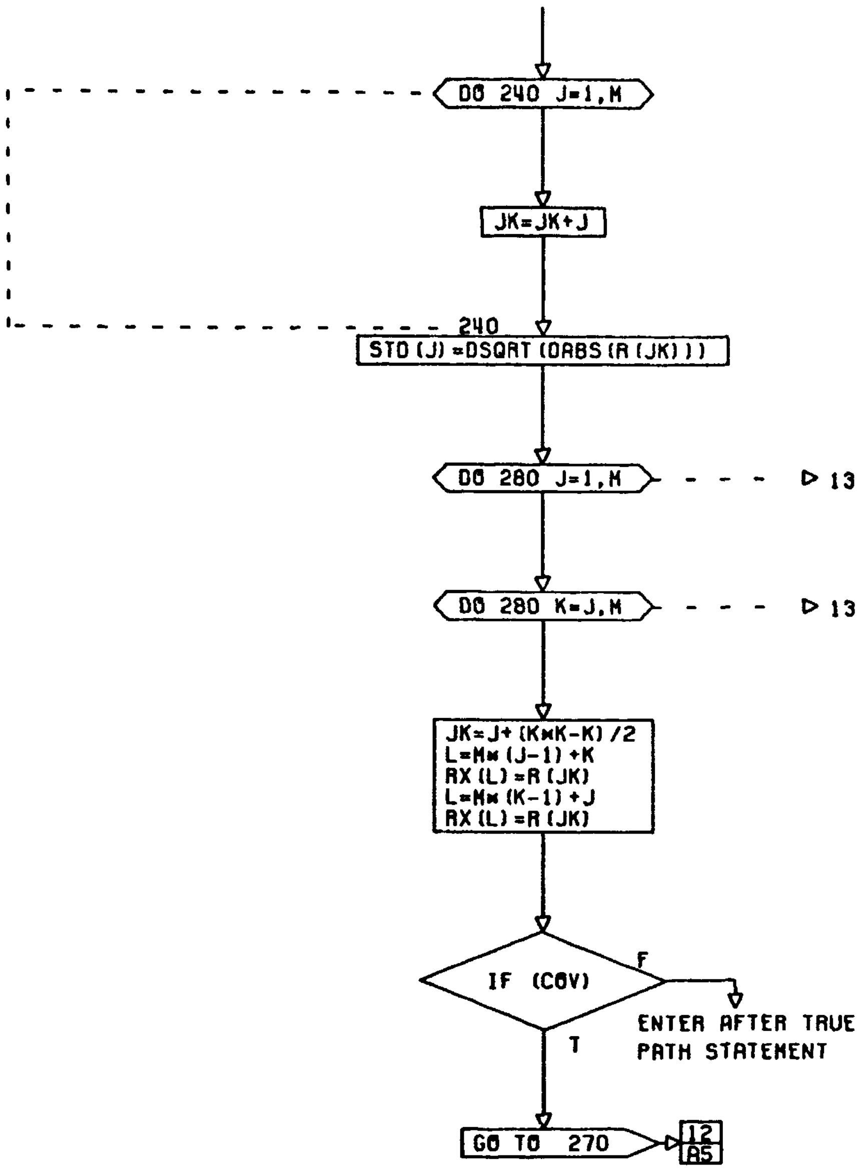


Figure 1-3 (Cont.)



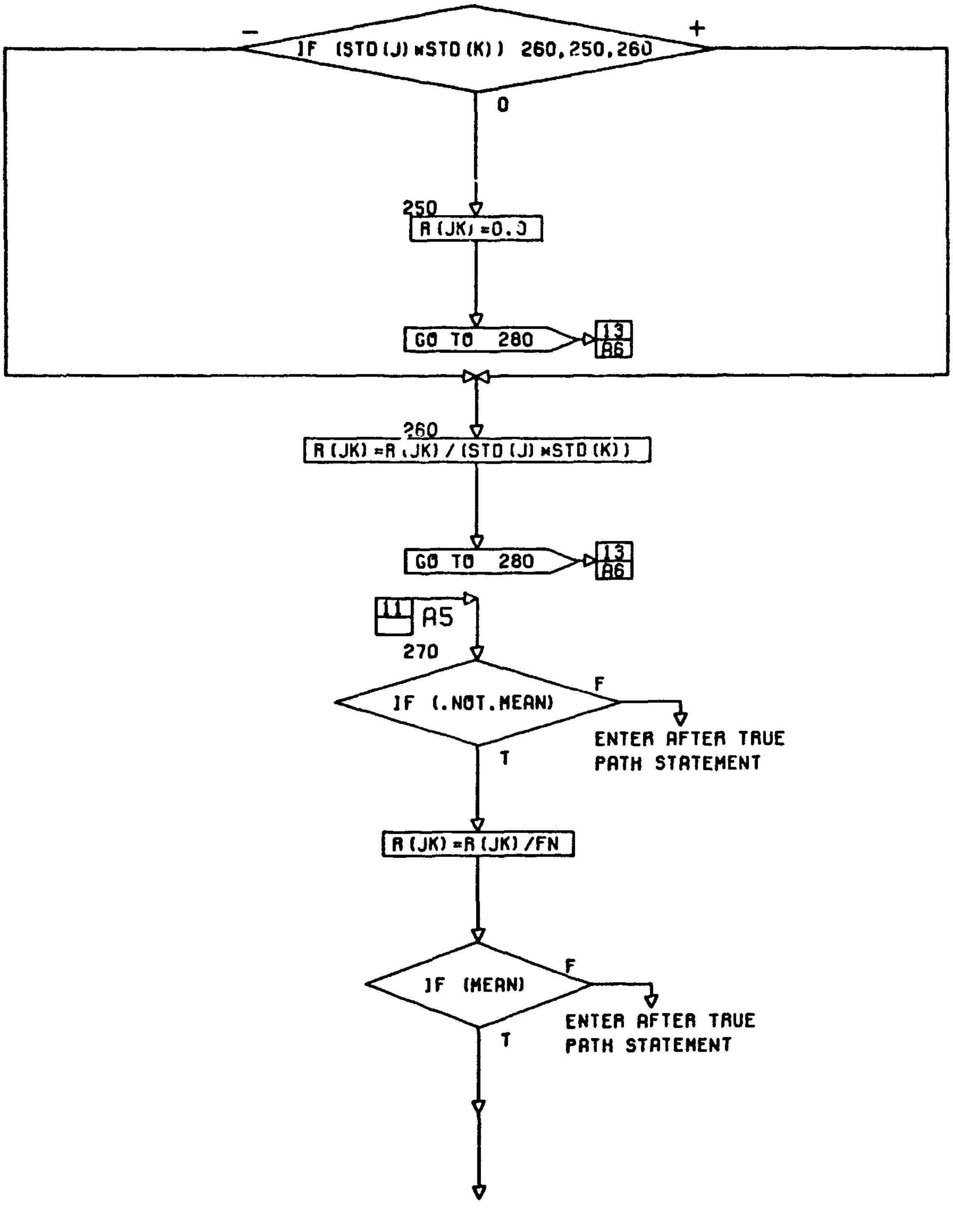
8059-1021

Figure 1-3 (Cont.)



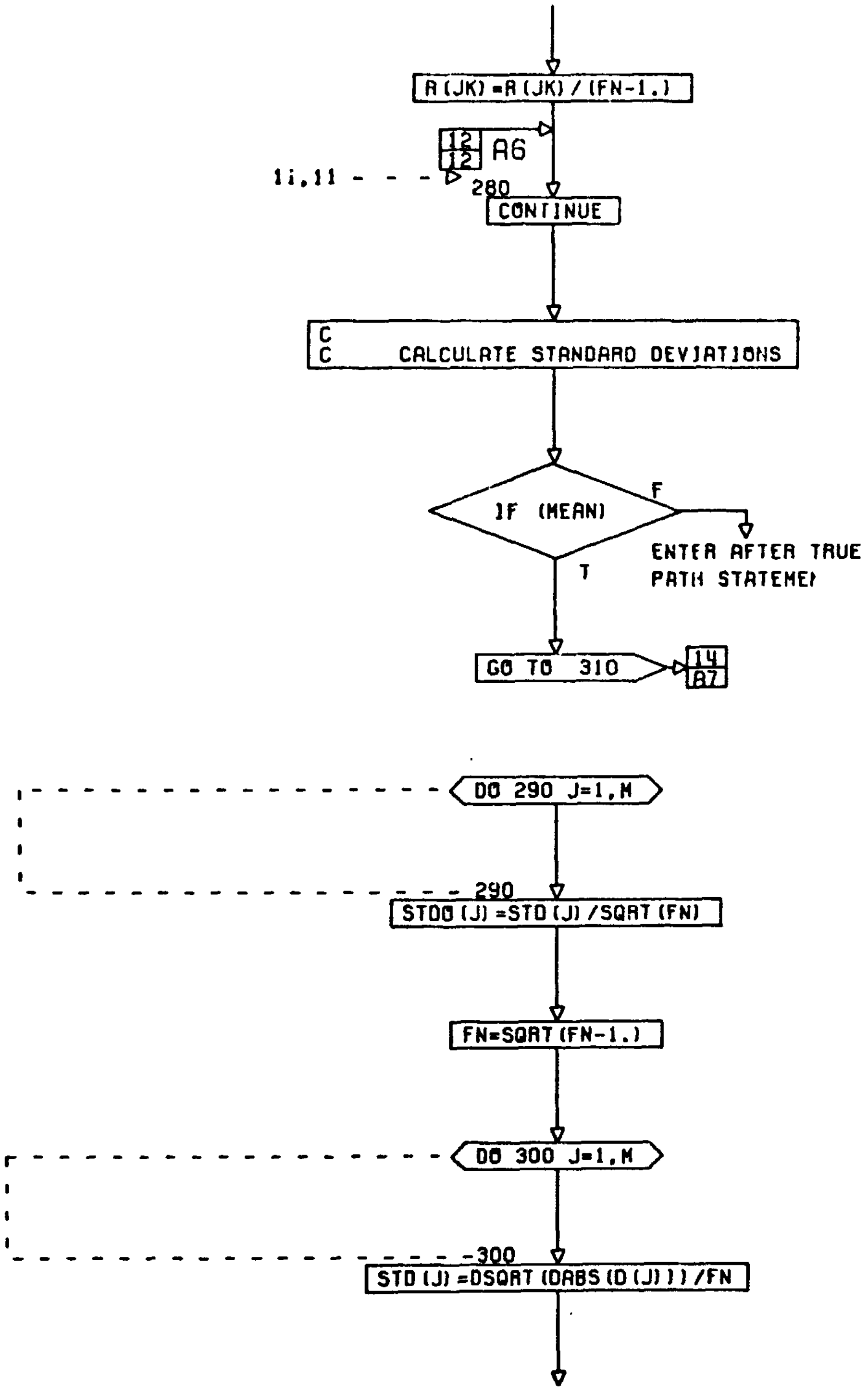
8059-102m

Figure 1-3 (Cont.)


−
IF (STD(J)*STD(K)) 260,250,260
+
0
250
R(JK)=0.0
GO TO 280
13
A6
260
R(JK)=R(JK)/(STD(J)*STD(K))
GO TO 280
13
A6
11
A5
270
IF (.NOT.MEAN)
F
ENTER AFTER TRUE PATH STATEMENT
T
R(JK)=R(JK)/FN
IF (MEAN)
F
ENTER AFTER TRUE PATH STATEMENT
T

8059-102m

Figure 1-3 (Cont.)



8059-1020

Figure 1-3 (Cont.)

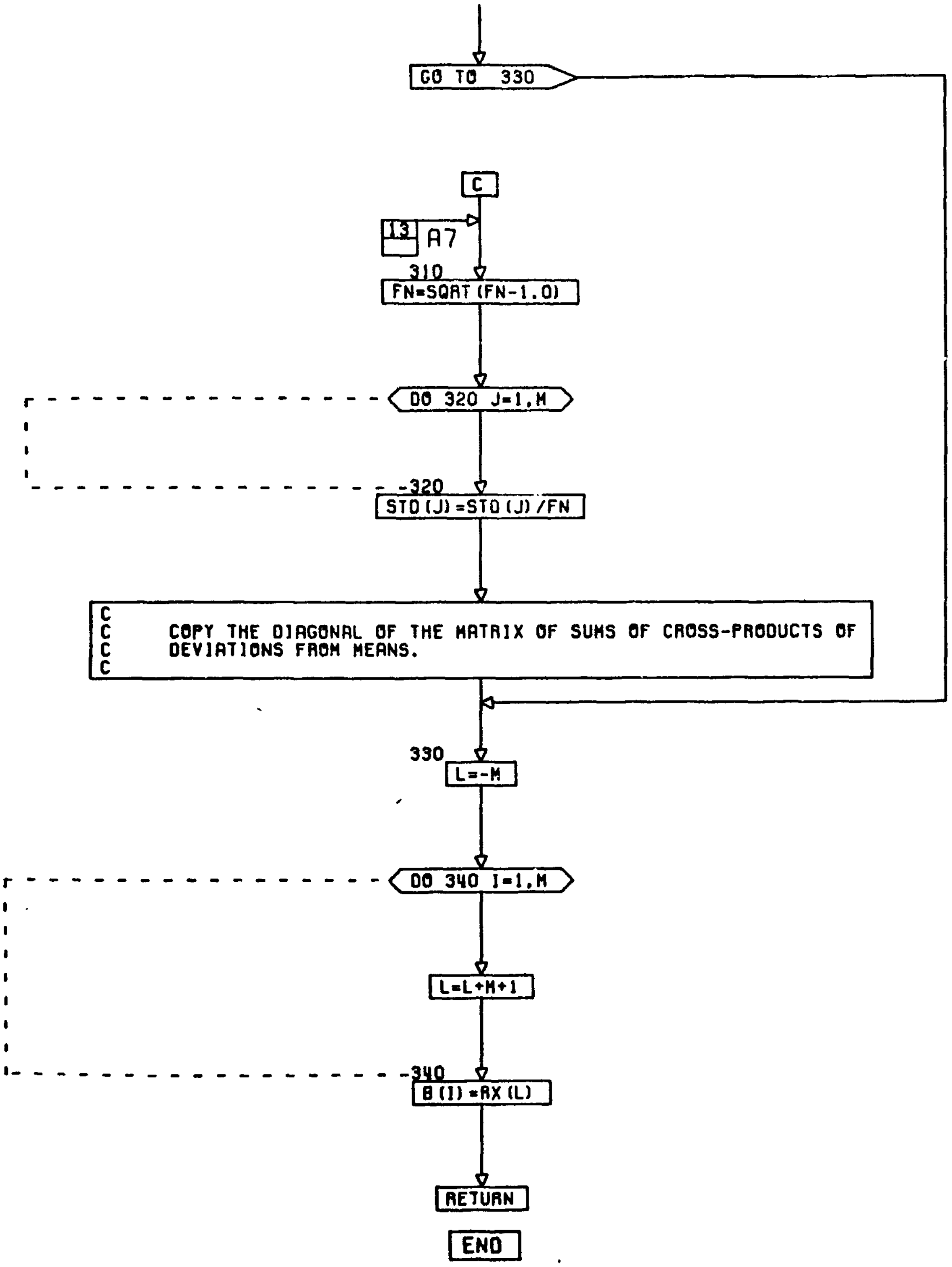


Figure 1-3 (Cont.)

KEY TO NOTATION

<u>Subroutine EIGEN</u>

n = order of input matrix (corresponds to m of main program)

${}_nA_n$ = input correlation or covariance matrix, destroyed during computation; eigenvalues are placed in diagonal as output

${}_nR_n$ = eigenvectors, columnwise

V_I = initial norm

V_f = final norm

t = threshold

ind = indicator that off-diagonal elements found larger than present threshold.

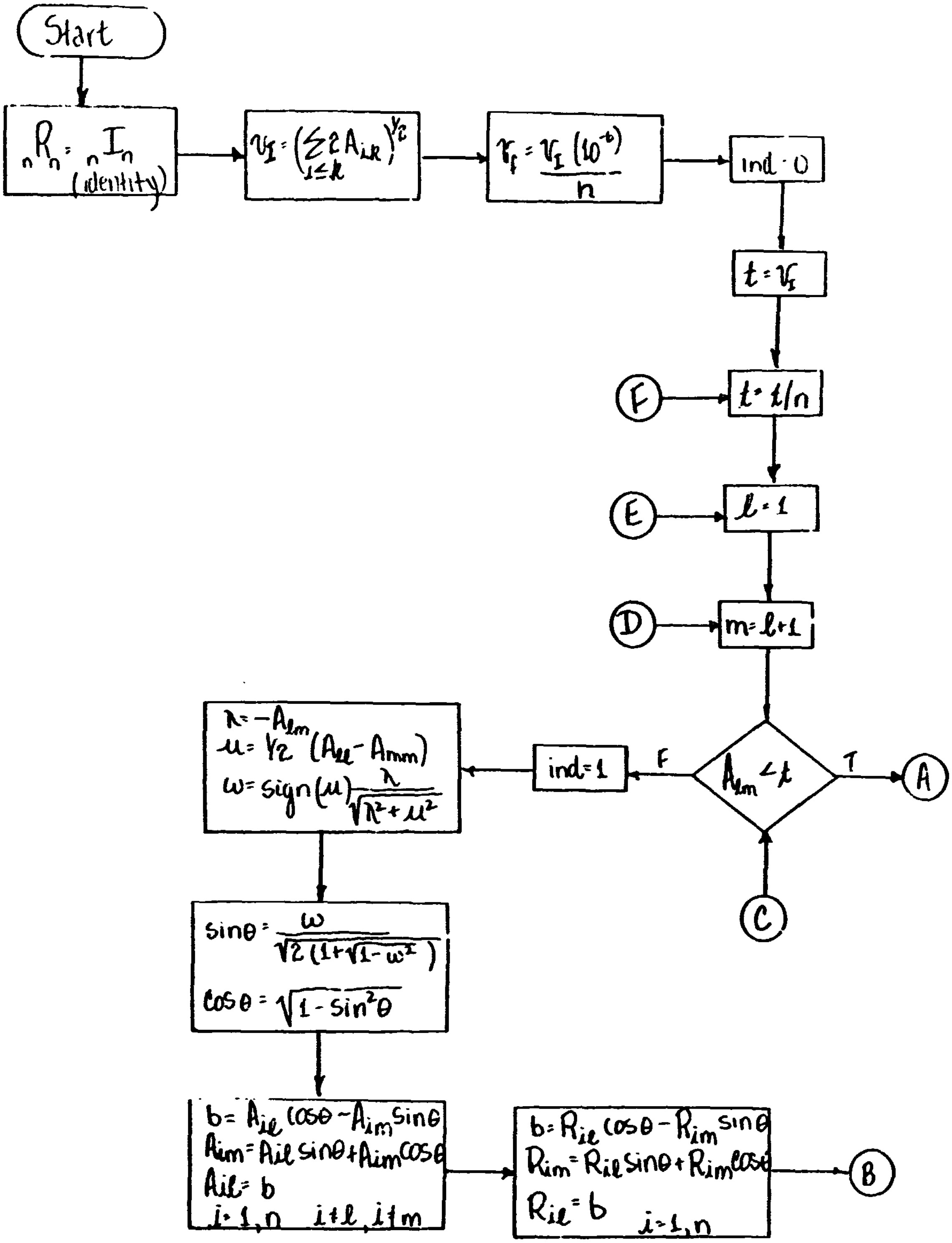
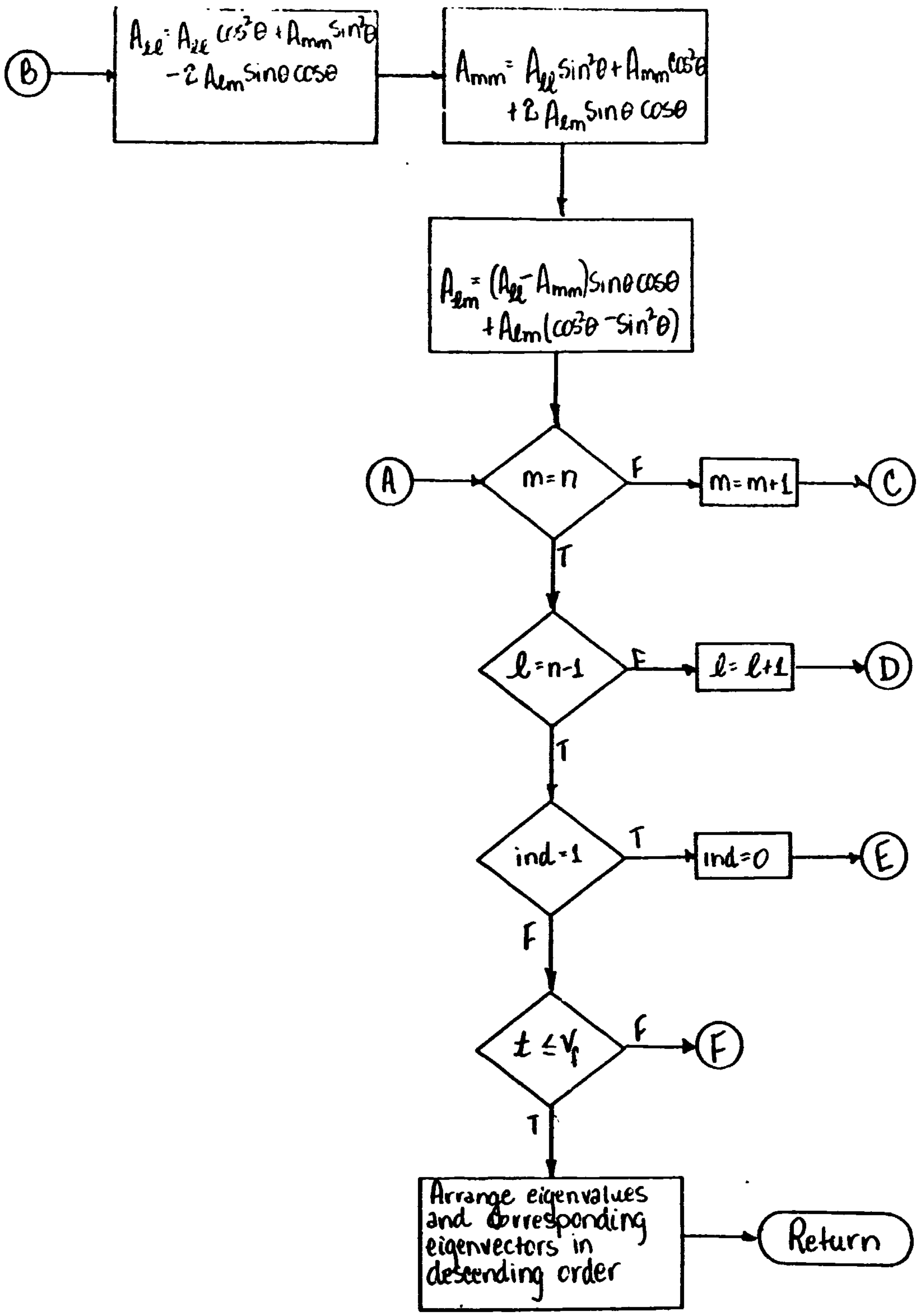


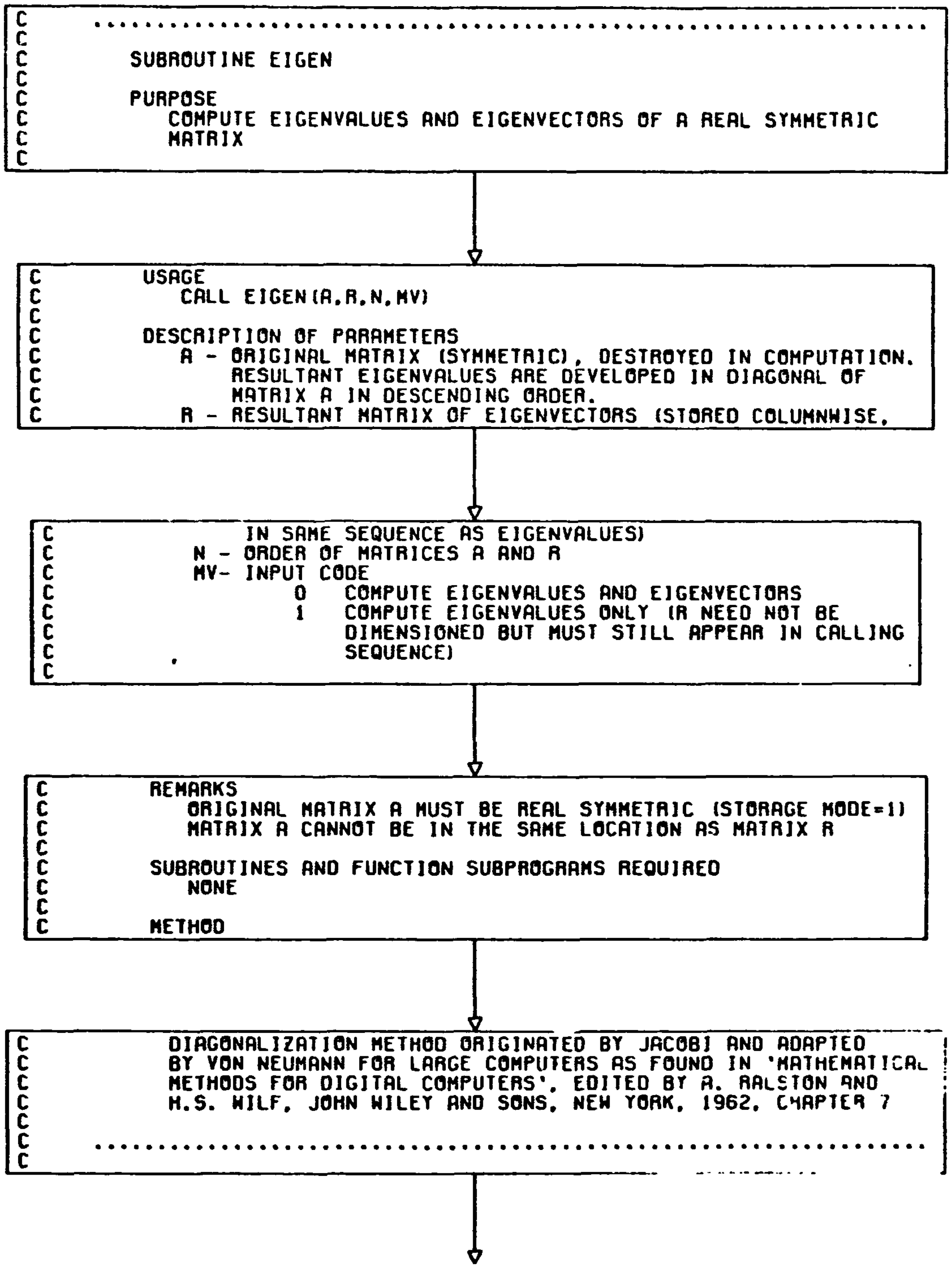
Figure 1-4 Subroutine EIGEN

8059-103a


B
$A_{\ell\ell} = A_{\ell\ell}\cos^2\theta + A_{mm}\sin^2\theta - 2A_{\ell m}\sin\theta\cos\theta$
$A_{mm} = A_{\ell\ell}\sin^2\theta + A_{mm}\cos^2\theta + 2A_{\ell m}\sin\theta\cos\theta$
$A_{\ell m} = (A_{\ell\ell} - A_{mm})\sin\theta\cos\theta + A_{\ell m}(\cos^2\theta - \sin^2\theta)$
A
m=n
F
m=m+1
C
T
ℓ=n-1
F
ℓ=ℓ+1
D
T
ind=1
T
ind=0
E
F
$t \le V_f$
F
F
T
Arrange eigenvalues and corresponding eigenvectors in descending order
Return

8089-103b

Figure 1-4 (Cont.)



```
C     ...............................................................
C
C        SUBROUTINE EIGEN
C
C        PURPOSE
C           COMPUTE EIGENVALUES AND EIGENVECTORS OF A REAL SYMMETRIC
C           MATRIX
C
```

```
C        USAGE
C           CALL EIGEN(A,R,N,MV)
C
C        DESCRIPTION OF PARAMETERS
C           A - ORIGINAL MATRIX (SYMMETRIC), DESTROYED IN COMPUTATION.
C               RESULTANT EIGENVALUES ARE DEVELOPED IN DIAGONAL OF
C               MATRIX A IN DESCENDING ORDER.
C           R - RESULTANT MATRIX OF EIGENVECTORS (STORED COLUMNWISE,
```

```
C               IN SAME SEQUENCE AS EIGENVALUES)
C           N - ORDER OF MATRICES A AND R
C           MV- INPUT CODE
C                   0   COMPUTE EIGENVALUES AND EIGENVECTORS
C                   1   COMPUTE EIGENVALUES ONLY (R NEED NOT BE
C                       DIMENSIONED BUT MUST STILL APPEAR IN CALLING
C                       SEQUENCE)
C
```

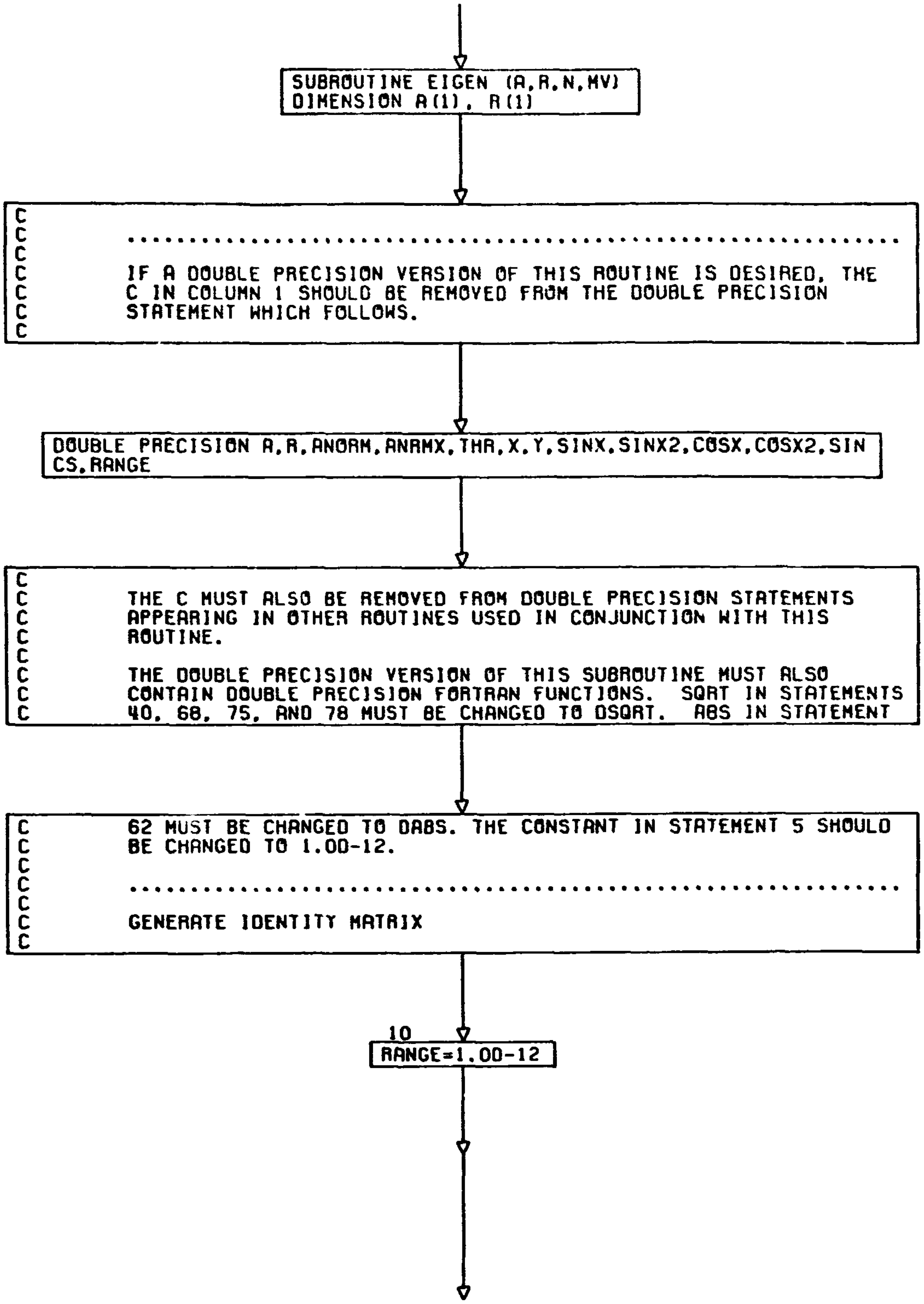
```
C        REMARKS
C           ORIGINAL MATRIX A MUST BE REAL SYMMETRIC (STORAGE MODE=1)
C           MATRIX A CANNOT BE IN THE SAME LOCATION AS MATRIX R
C
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C           NONE
C
C        METHOD
```

```
C           DIAGONALIZATION METHOD ORIGINATED BY JACOBI AND ADAPTED
C           BY VON NEUMANN FOR LARGE COMPUTERS AS FOUND IN 'MATHEMATICAL
C           METHODS FOR DIGITAL COMPUTERS', EDITED BY A. RALSTON AND
C           H.S. WILF, JOHN WILEY AND SONS, NEW YORK, 1962, CHAPTER 7
C
C     ...............................................................
C
```

CONT. ON PG _

Figure 1-4 (Cont.)

8069-103c



```
      SUBROUTINE EIGEN (A,R,N,MV)
      DIMENSION A(1), R(1)
```

```
C
C        ...............................................................
C
C        IF A DOUBLE PRECISION VERSION OF THIS ROUTINE IS DESIRED, THE
C        C IN COLUMN 1 SHOULD BE REMOVED FROM THE DOUBLE PRECISION
C        STATEMENT WHICH FOLLOWS.
C
```

```
      DOUBLE PRECISION A,R,ANORM,ANRMX,THR,X,Y,SINX,SINX2,COSX,COSX2,SIN
     CS,RANGE
```

```
C
C        THE C MUST ALSO BE REMOVED FROM DOUBLE PRECISION STATEMENTS
C        APPEARING IN OTHER ROUTINES USED IN CONJUNCTION WITH THIS
C        ROUTINE.
C
C        THE DOUBLE PRECISION VERSION OF THIS SUBROUTINE MUST ALSO
C        CONTAIN DOUBLE PRECISION FORTRAN FUNCTIONS.  SQRT IN STATEMENTS
C        40, 68, 75, AND 78 MUST BE CHANGED TO DSQRT.  ABS IN STATEMENT
```

```
C        62 MUST BE CHANGED TO DABS. THE CONSTANT IN STATEMENT 5 SHOULD
C        BE CHANGED TO 1.0D-12.
C
C        ...............................................................
C
C        GENERATE IDENTITY MATRIX
C
```

```
   10 RANGE=1.0D-12
```

CONT. ON PG 3

Figure 1-4 (Cont.)

BSR 2949

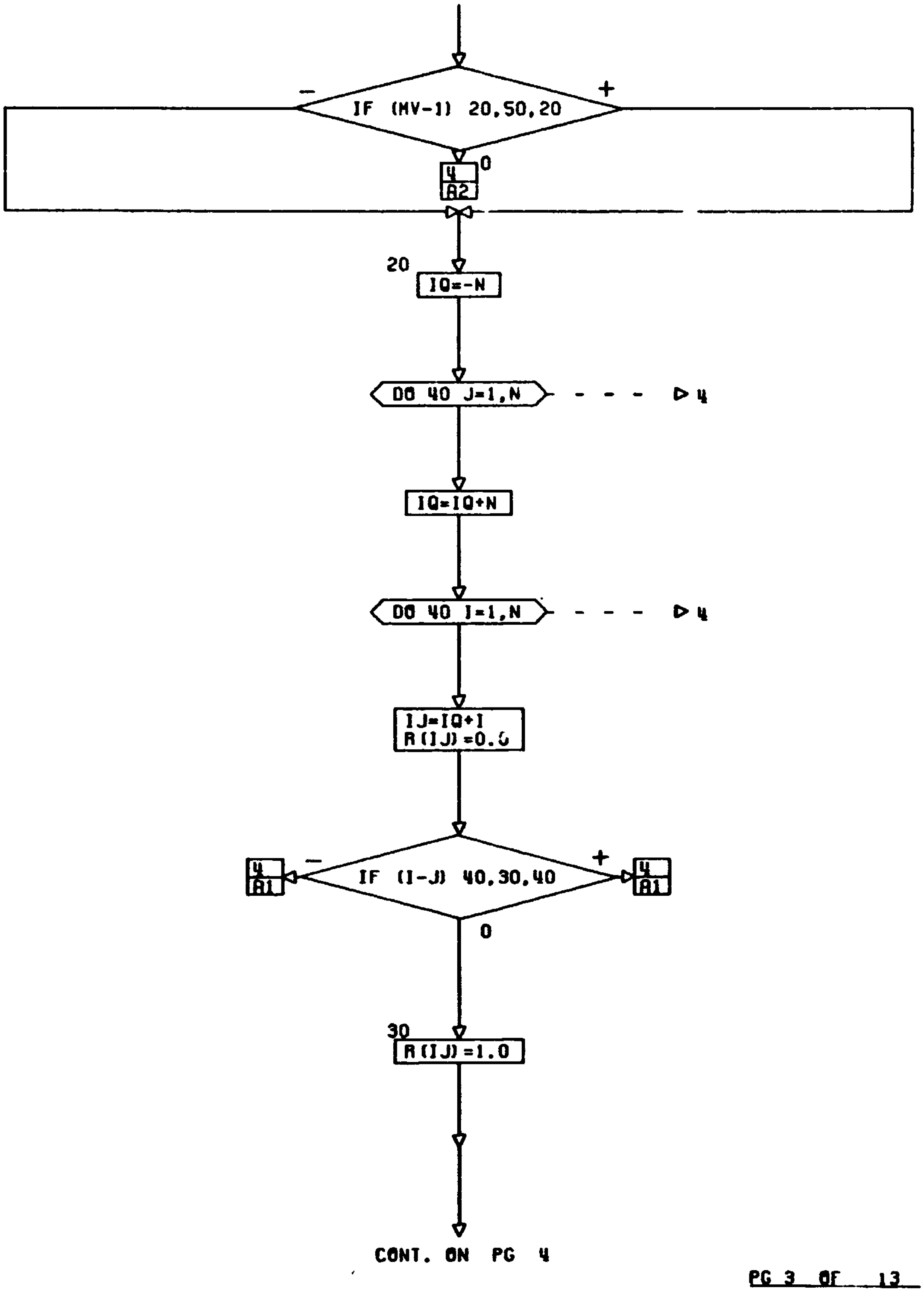
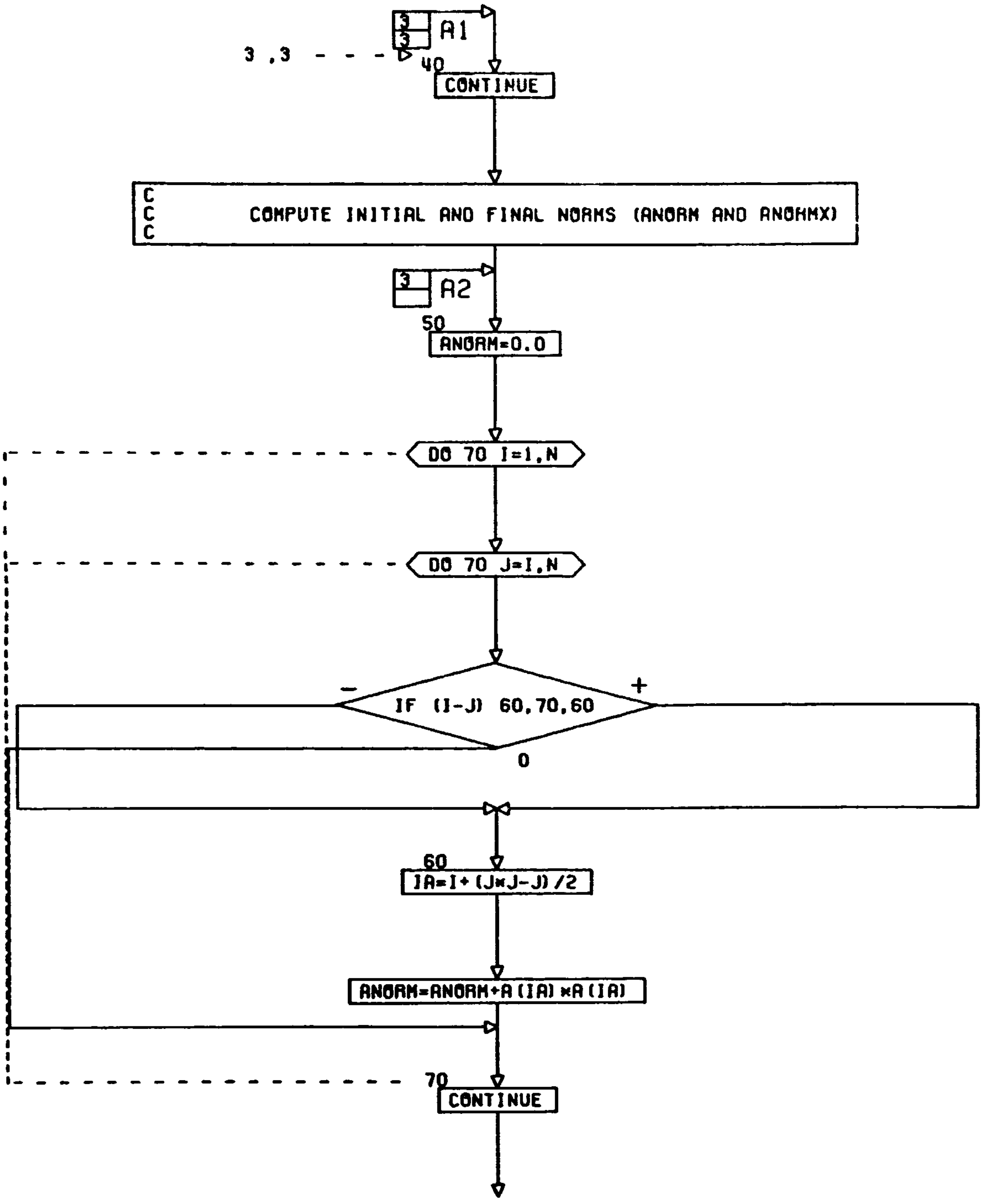


Figure 1-4 (Cont.)

8059-103e



CONT. ON PG 5

PG 4 OF 13

8059-1031

Figure 1-4 (Cont.)

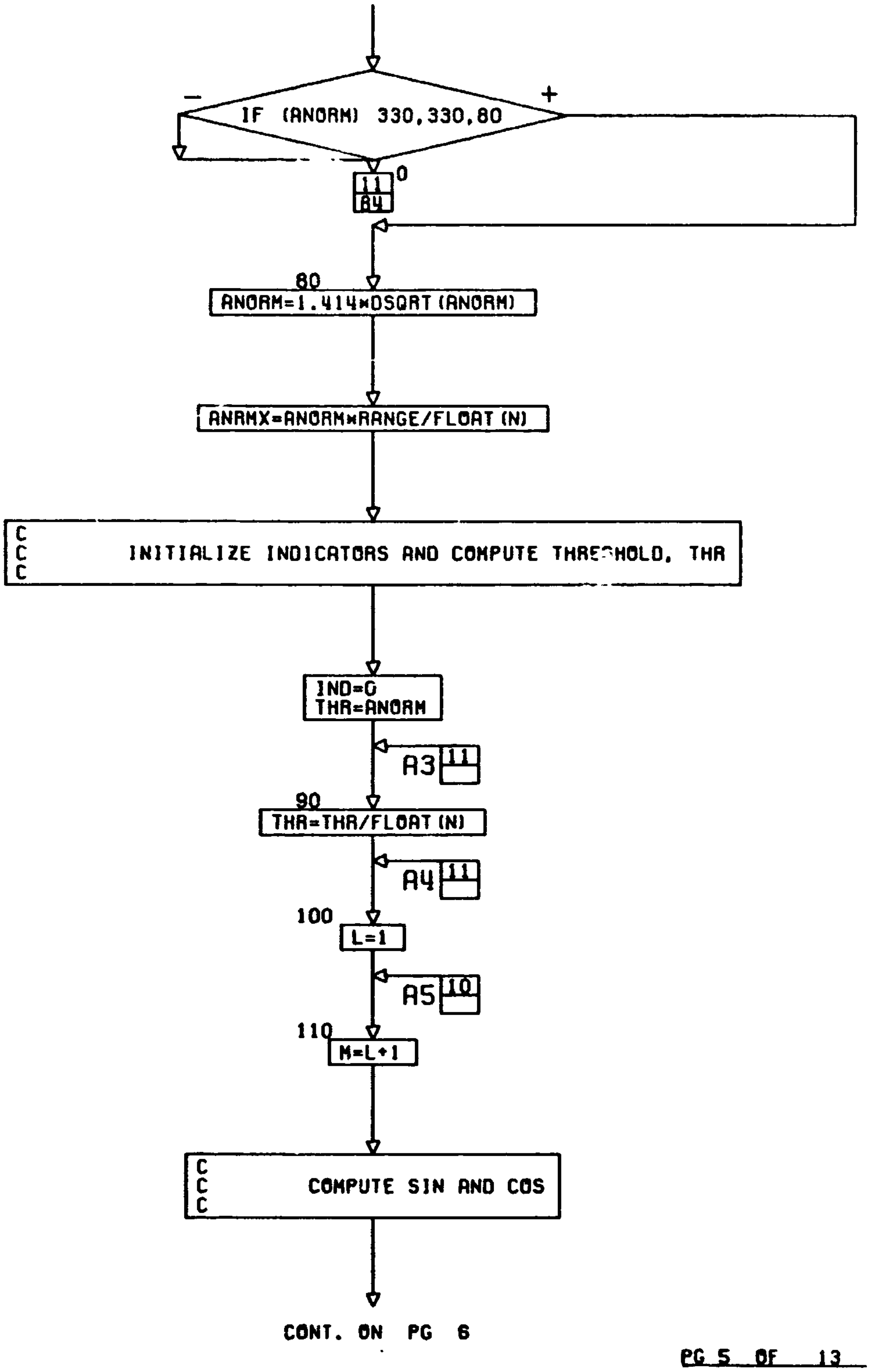
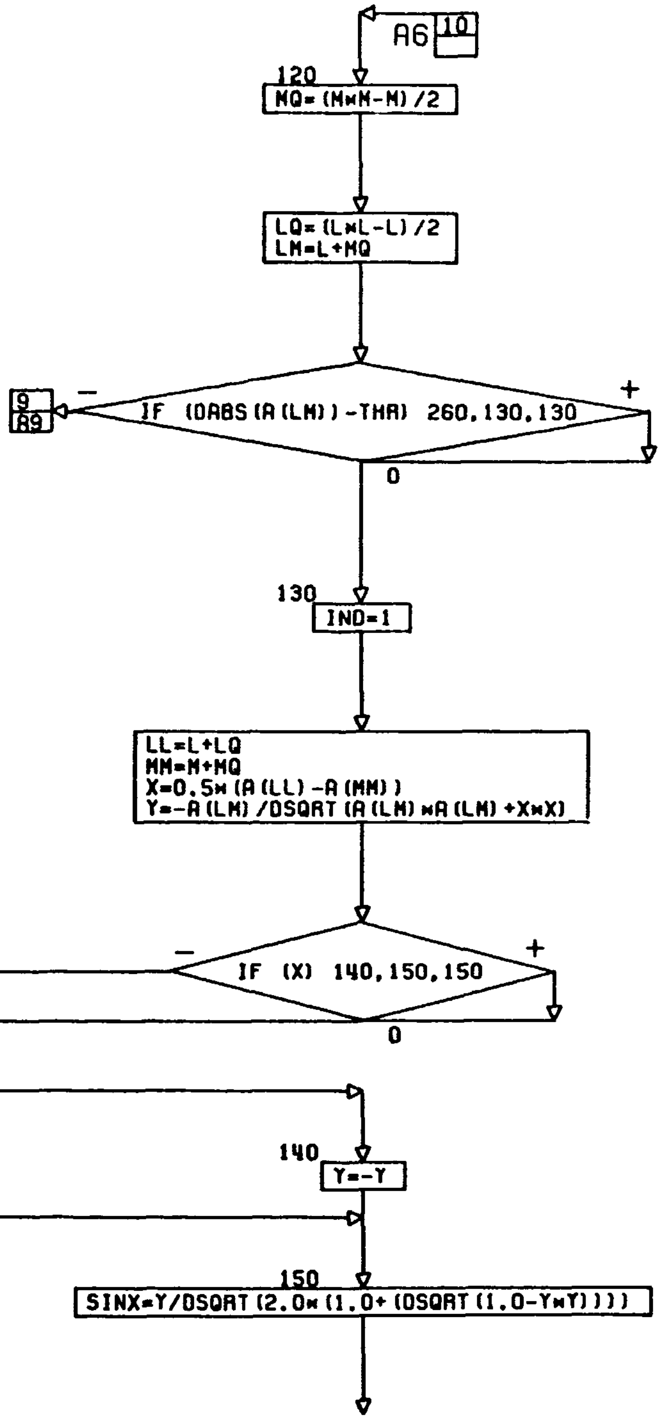


Figure 1-4 (Cont.)

8059-103g



CONT. ON PG 7

Figure 1-4 (Cont.)

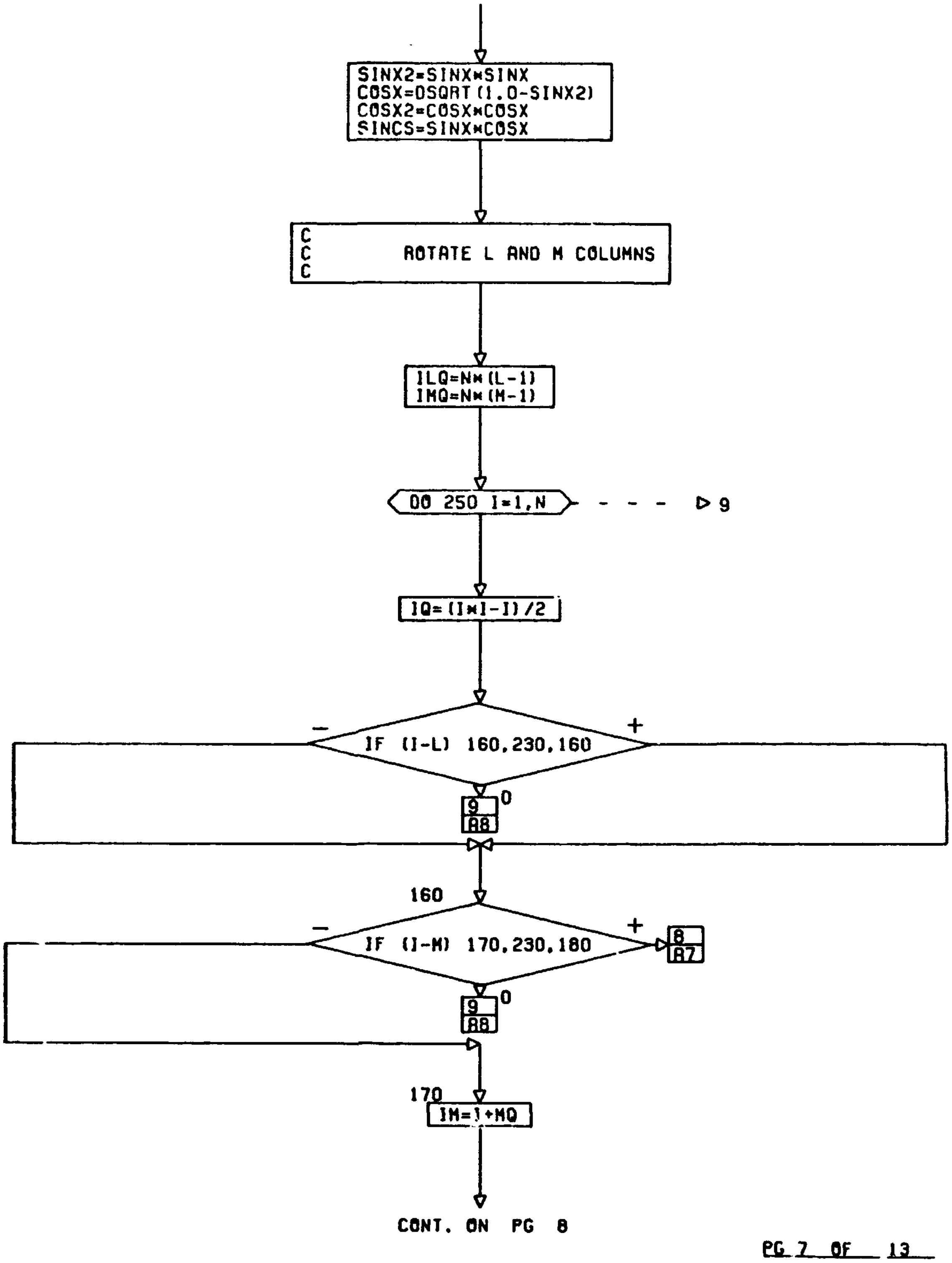
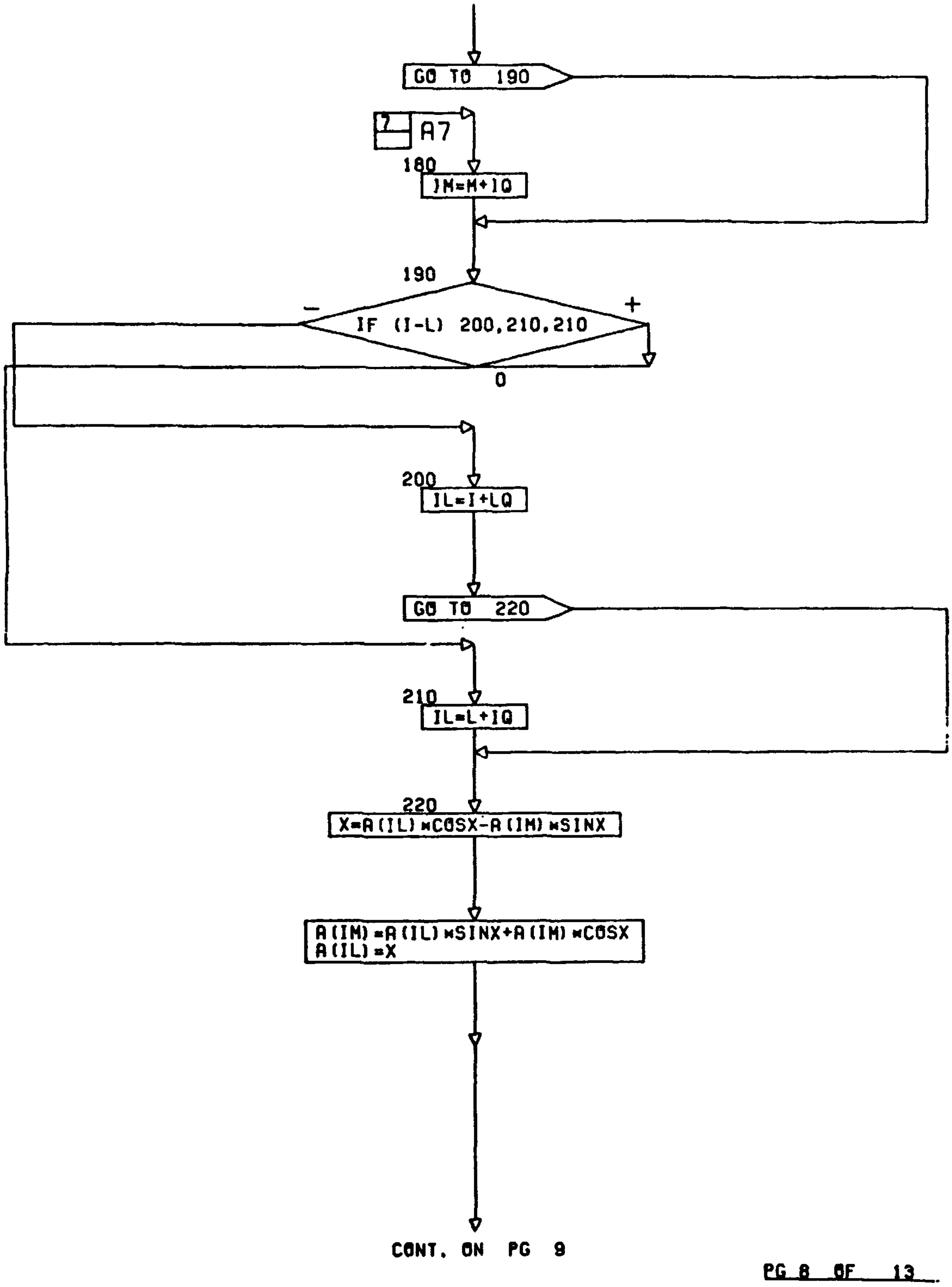


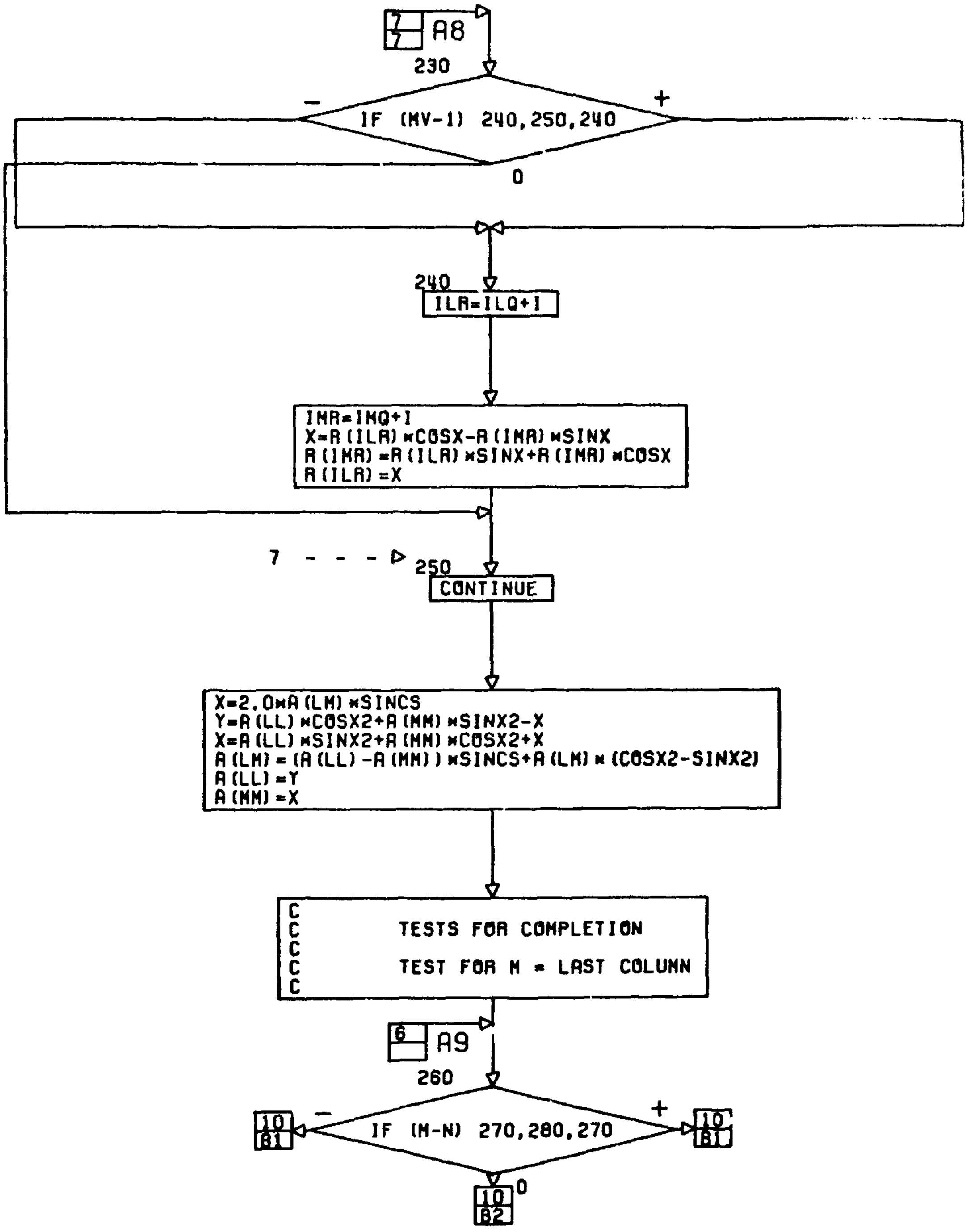
Figure 1-4 (Cont.)

8059-1031



8059-103]

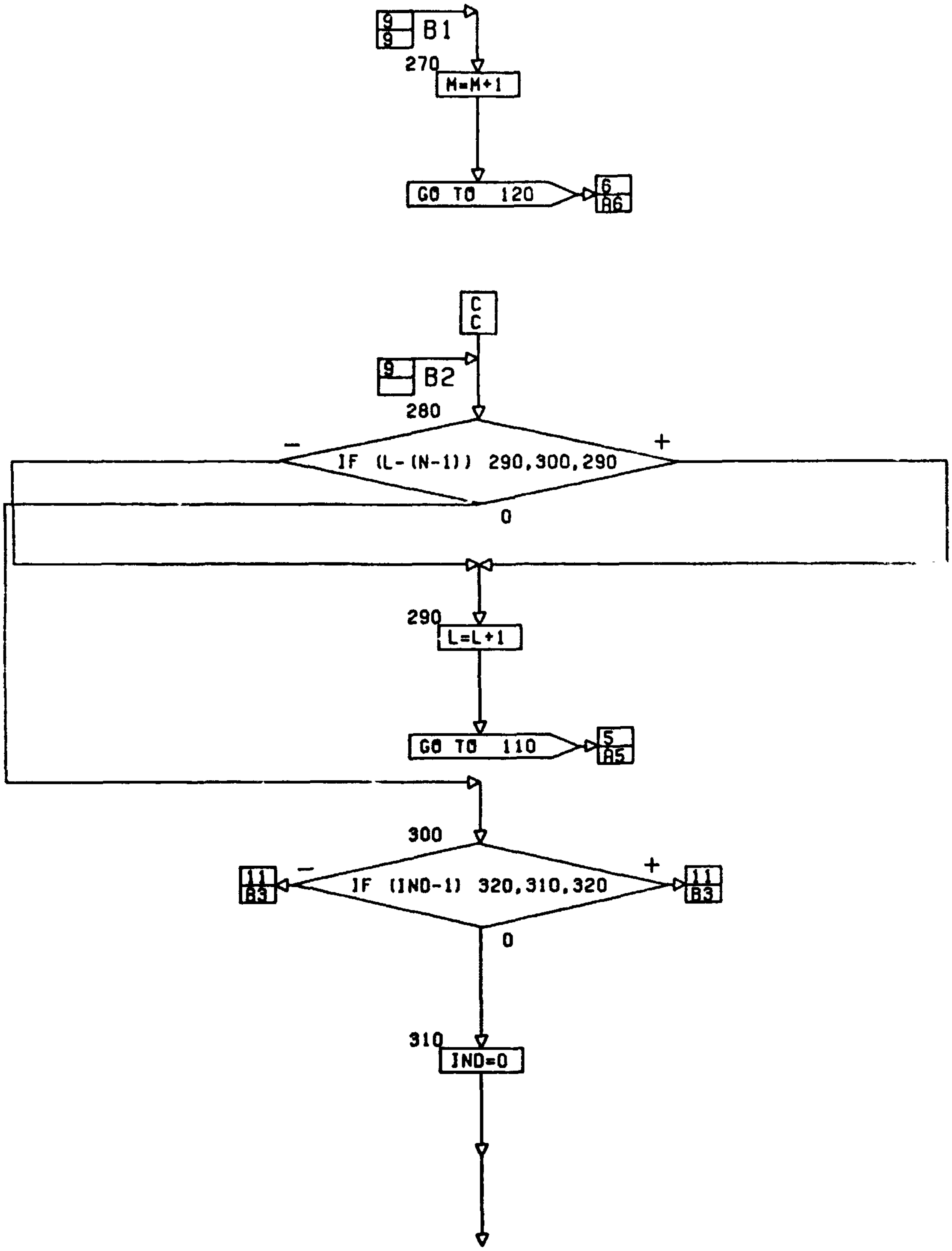
Figure 1-4 (Cont.)



```
IF (MV-1) 240,250,240

ILR=ILQ+I

IMR=IMQ+I
X=A(ILA)*COSX-A(IMA)*SINX
A(IMA)=A(ILA)*SINX+A(IMA)*COSX
A(ILA)=X

CONTINUE

X=2.0*A(LM)*SINCS
Y=A(LL)*COSX2+A(MM)*SINX2-X
X=A(LL)*SINX2+A(MM)*COSX2+X
A(LM)=(A(LL)-A(MM))*SINCS+A(LM)*(COSX2-SINX2)
A(LL)=Y
A(MM)=X

C
C         TESTS FOR COMPLETION
C
C         TEST FOR M = LAST COLUMN
C

IF (M-N) 270,280,270
```

CONT. ON PG. 10

Figure 1-4 (Cont.)



CONT. ON PG 11

PG 10 OF 13

8059-1031

Figure 1-4 (Cont.)

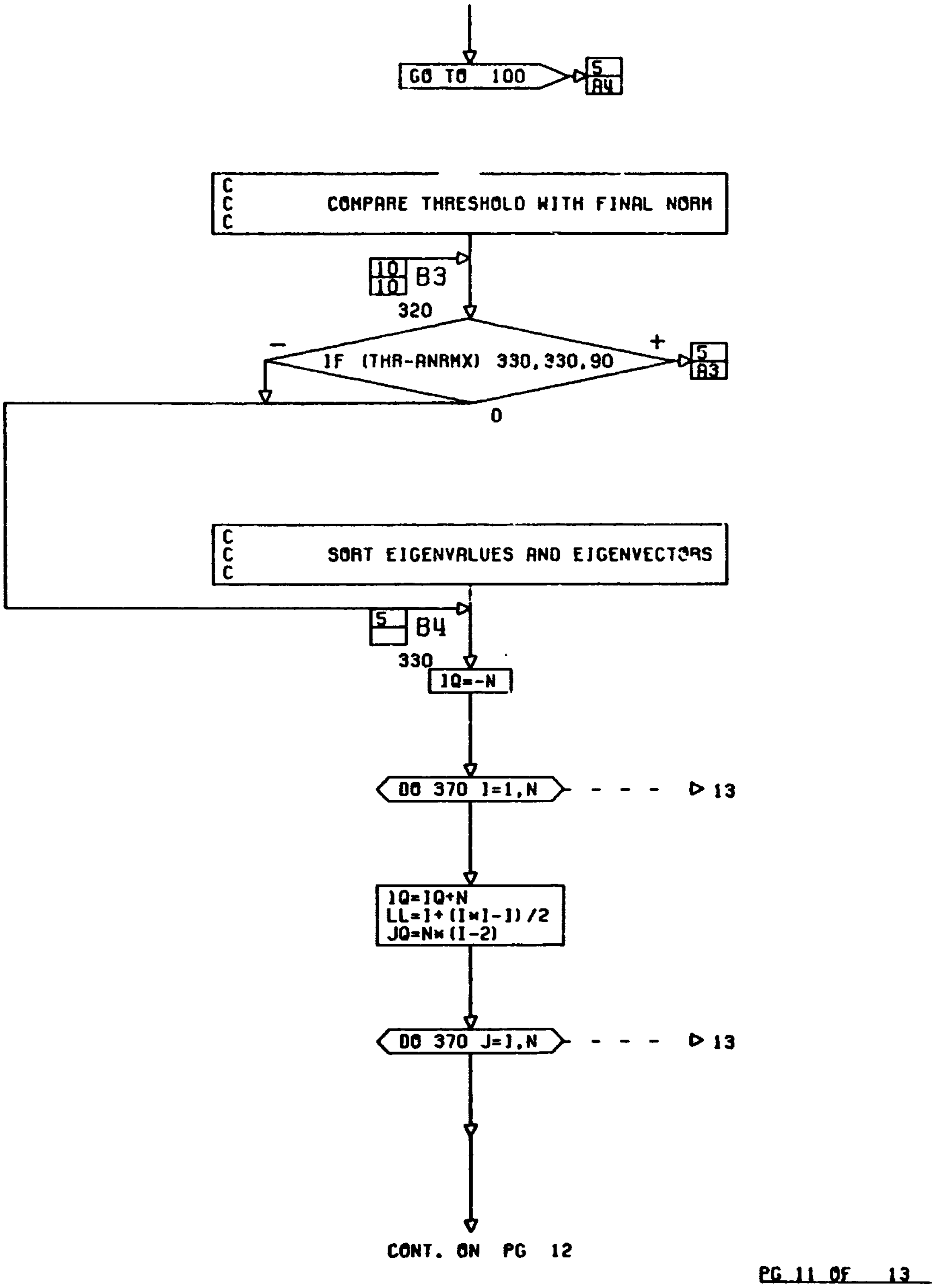
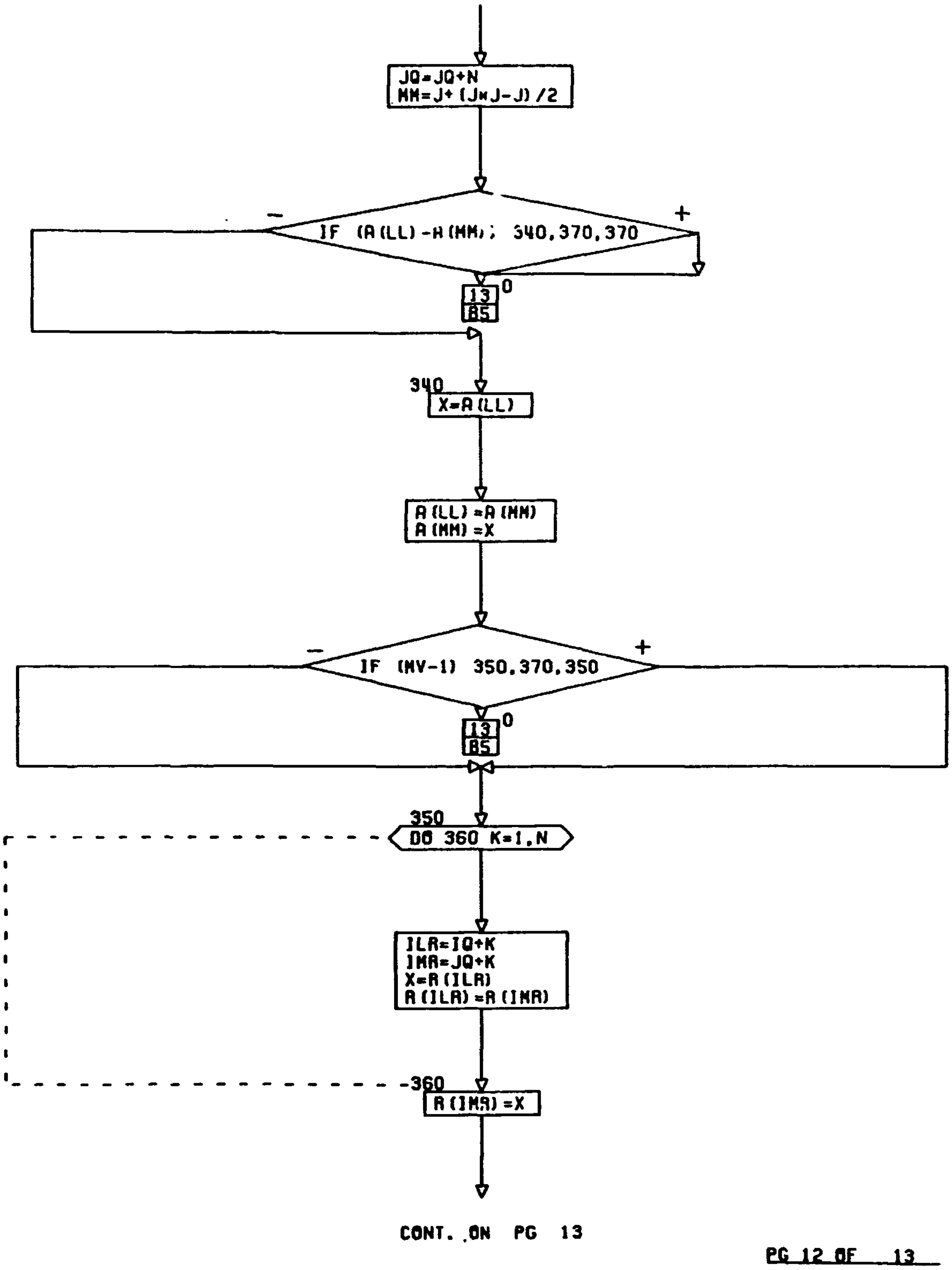


Figure 1-4 (Cont.)



CONT. ON PG 13

Figure 1-4 (Cont.)

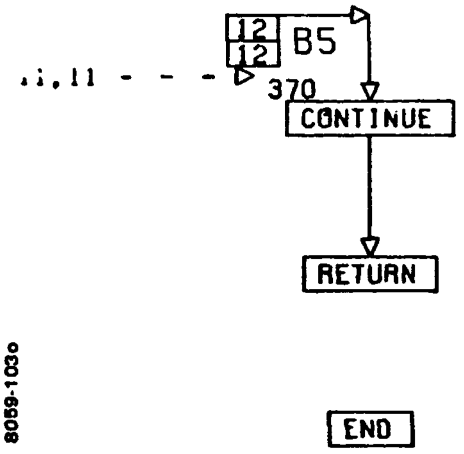


Figure 1-4 (Cont.)

PG 13 FINAL

KEY TO NOTATION

Subroutine TRACE

m = number of eigenvalues = (number of variables)

k number of factors kept

λ_i, $i=1, m$ = eigenvalues

t = total variance

c = number of factors to keep, or percentage of total variance to be explained, or minimum eigenvalue to be kept

d_j, $j=1, k$ = cumulative proportion of total variance.

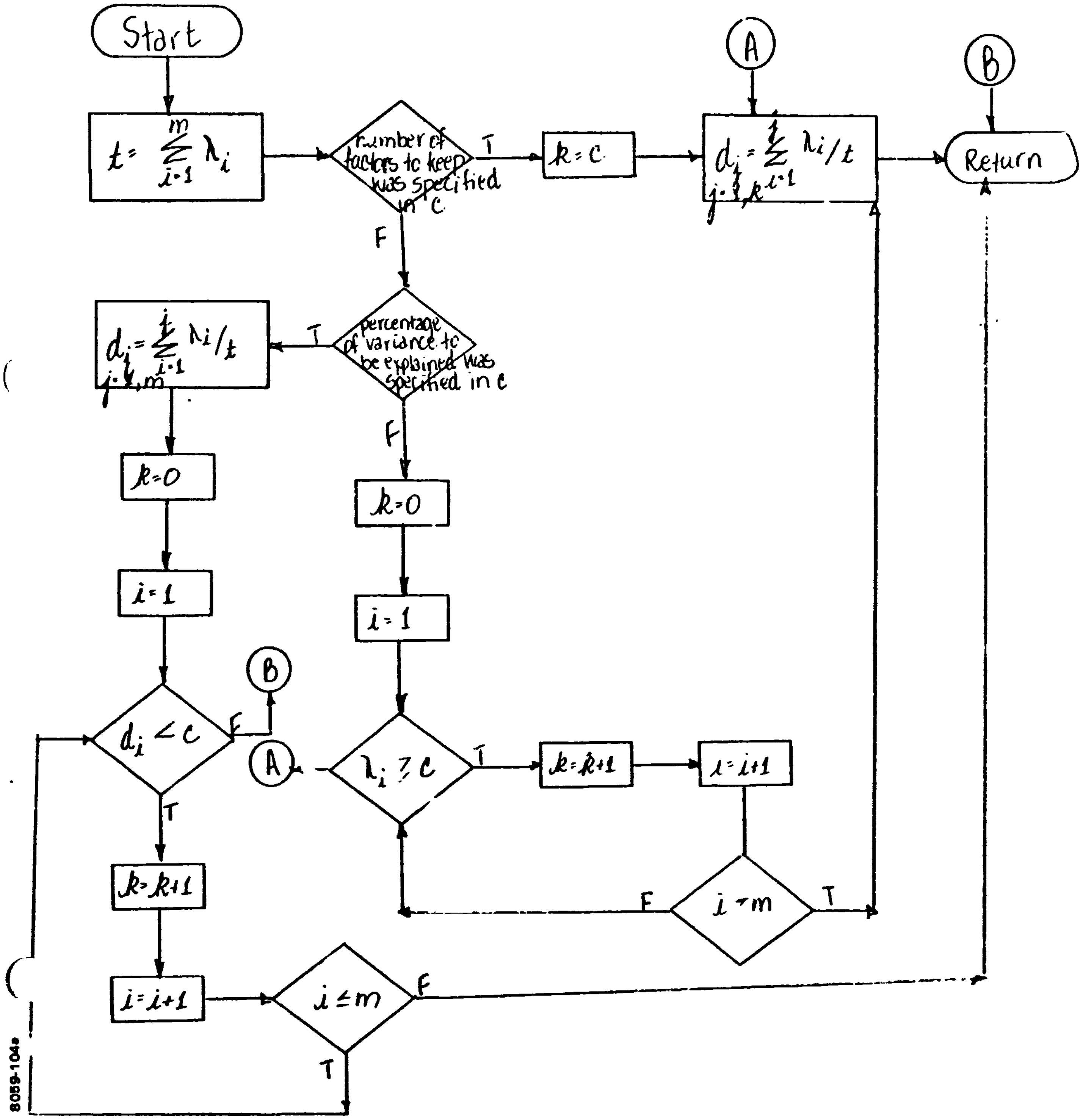
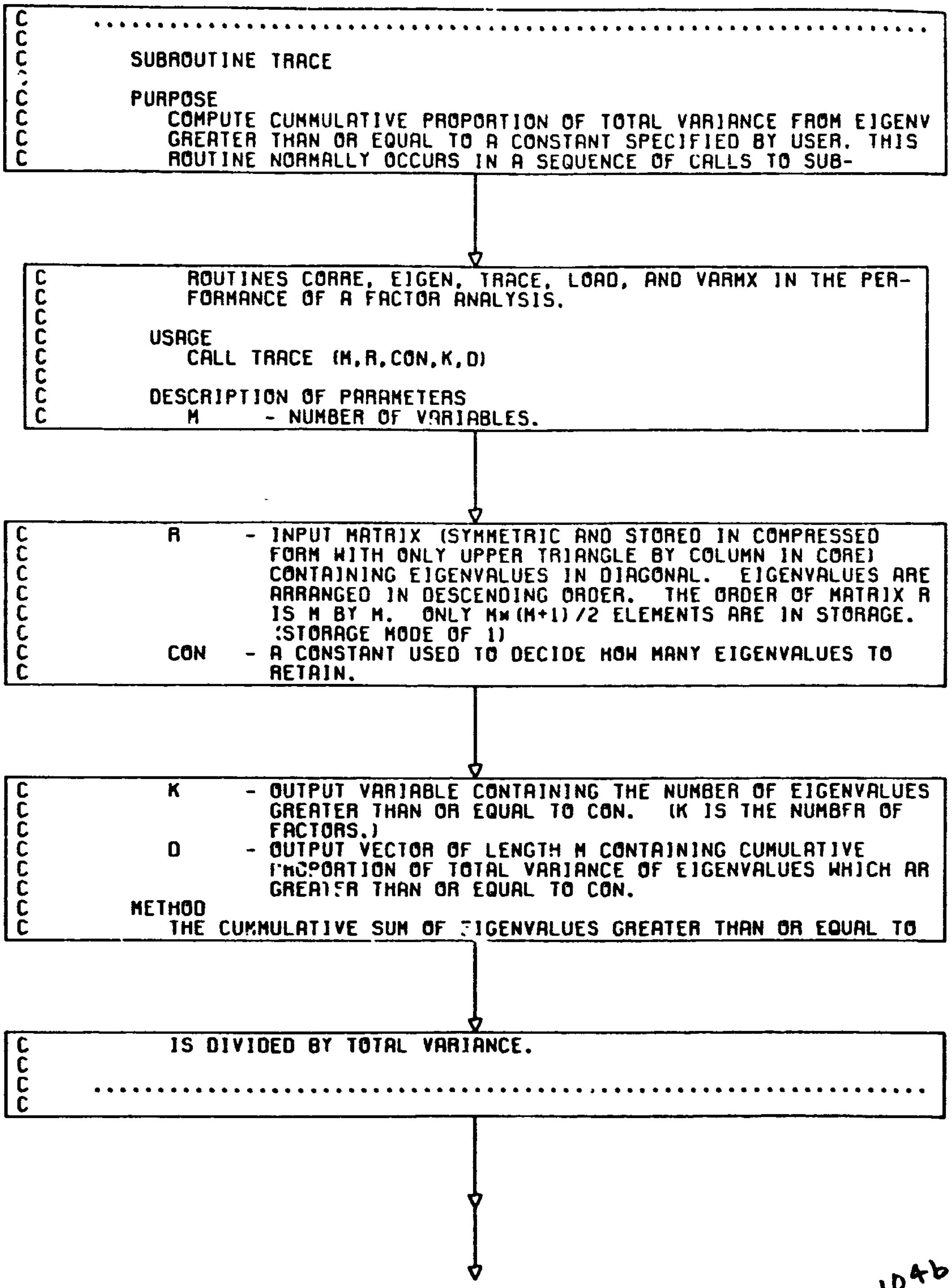

Start
$t = \sum_{i=1}^{m} \lambda_i$
number of factors to keep was specified in c
T
F
$k = c$
A
$d_j = \sum_{i=1}^{j} \lambda_i / t$ $j=1,k$
B
Return
percentage of variance to be explained was specified in c
T
F
$d_j = \sum_{i=1}^{j} \lambda_i / t$ $j=1,m$
$k=0$
$k=0$
$i=1$
$i=1$
B
$d_i < c$
F
T
A
$\lambda_i \geq c$
T
$k=k+1$
$i=i+1$
$i = m$
F
T
$k=k+1$
$i=i+1$
$i \leq m$
F
T
8059-104e

Figure 1-5 Subroutine TRACE



```
C     ..............................................................
C
C        SUBROUTINE TRACE
C
C        PURPOSE
C           COMPUTE CUMMULATIVE PROPORTION OF TOTAL VARIANCE FROM EIGENV
C           GREATER THAN OR EQUAL TO A CONSTANT SPECIFIED BY USER. THIS
C           ROUTINE NORMALLY OCCURS IN A SEQUENCE OF CALLS TO SUB-
```

```
C           ROUTINES CORRE, EIGEN, TRACE, LOAD, AND VARMX IN THE PER-
C           FORMANCE OF A FACTOR ANALYSIS.
C
C        USAGE
C           CALL TRACE (M,R,CON,K,D)
C
C        DESCRIPTION OF PARAMETERS
C           M     - NUMBER OF VARIABLES.
```

```
C           R     - INPUT MATRIX (SYMMETRIC AND STORED IN COMPRESSED
C                   FORM WITH ONLY UPPER TRIANGLE BY COLUMN IN CORE)
C                   CONTAINING EIGENVALUES IN DIAGONAL.  EIGENVALUES ARE
C                   ARRANGED IN DESCENDING ORDER.  THE ORDER OF MATRIX R
C                   IS M BY M.  ONLY M*(M+1)/2 ELEMENTS ARE IN STORAGE.
C                   (STORAGE MODE OF 1)
C           CON   - A CONSTANT USED TO DECIDE HOW MANY EIGENVALUES TO
C                   RETAIN.
```

```
C           K     - OUTPUT VARIABLE CONTAINING THE NUMBER OF EIGENVALUES
C                   GREATER THAN OR EQUAL TO CON.  (K IS THE NUMBER OF
C                   FACTORS.)
C           D     - OUTPUT VECTOR OF LENGTH M CONTAINING CUMULATIVE
C                   PROPORTION OF TOTAL VARIANCE OF EIGENVALUES WHICH AR
C                   GREATER THAN OR EQUAL TO CON.
C        METHOD
C           THE CUMMULATIVE SUM OF EIGENVALUES GREATER THAN OR EQUAL TO
```

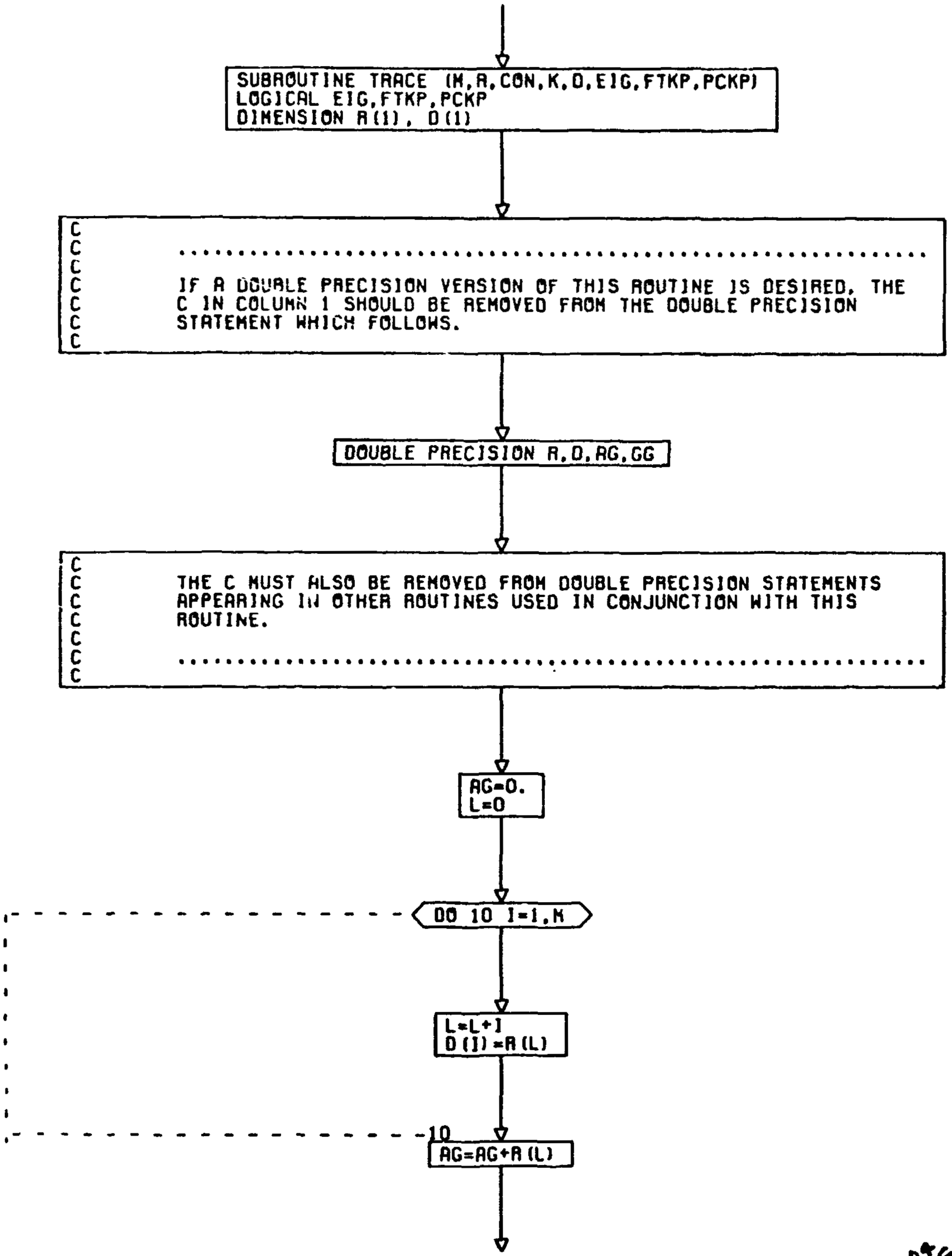
```
C           IS DIVIDED BY TOTAL VARIANCE.
C
C     ..............................................................
C
```

CONT. ON PG 2

104b

PG 1 OF 5

Figure 1-5 (Cont.)



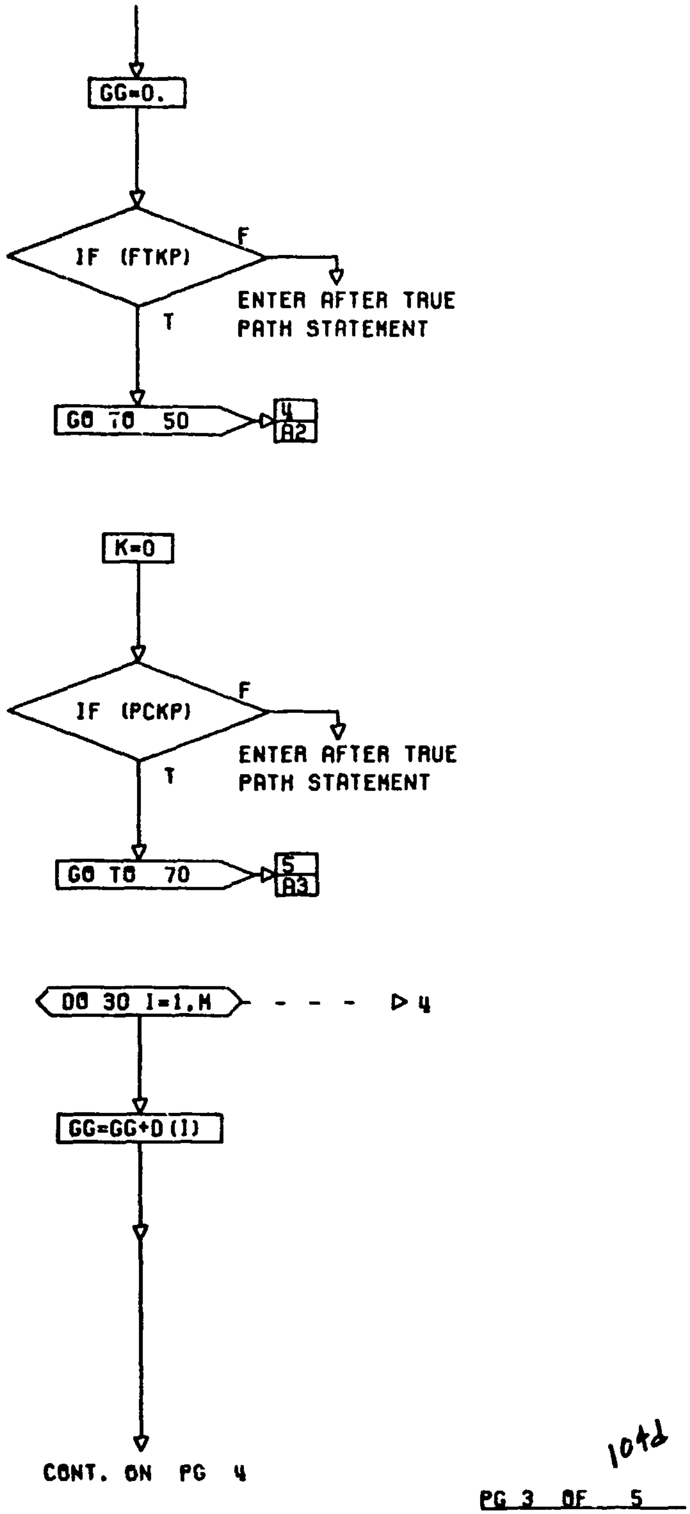
```
      SUBROUTINE TRACE (M,A,CON,K,D,EIG,FTKP,PCKP)
      LOGICAL EIG,FTKP,PCKP
      DIMENSION A(1), D(1)
C
C         ..............................................................
C
C         IF A DOUBLE PRECISION VERSION OF THIS ROUTINE IS DESIRED, THE
C         C IN COLUMN 1 SHOULD BE REMOVED FROM THE DOUBLE PRECISION
C         STATEMENT WHICH FOLLOWS.
C
      DOUBLE PRECISION A,D,AG,GG
C
C         THE C MUST ALSO BE REMOVED FROM DOUBLE PRECISION STATEMENTS
C         APPEARING IN OTHER ROUTINES USED IN CONJUNCTION WITH THIS
C         ROUTINE.
C
C         ..............................................................
C
      AG=0.
      L=0
      DO 10 I=1,M
      L=L+I
      D(I)=A(L)
   10 AG=AG+A(L)
```

CONT. ON PG 3

104c

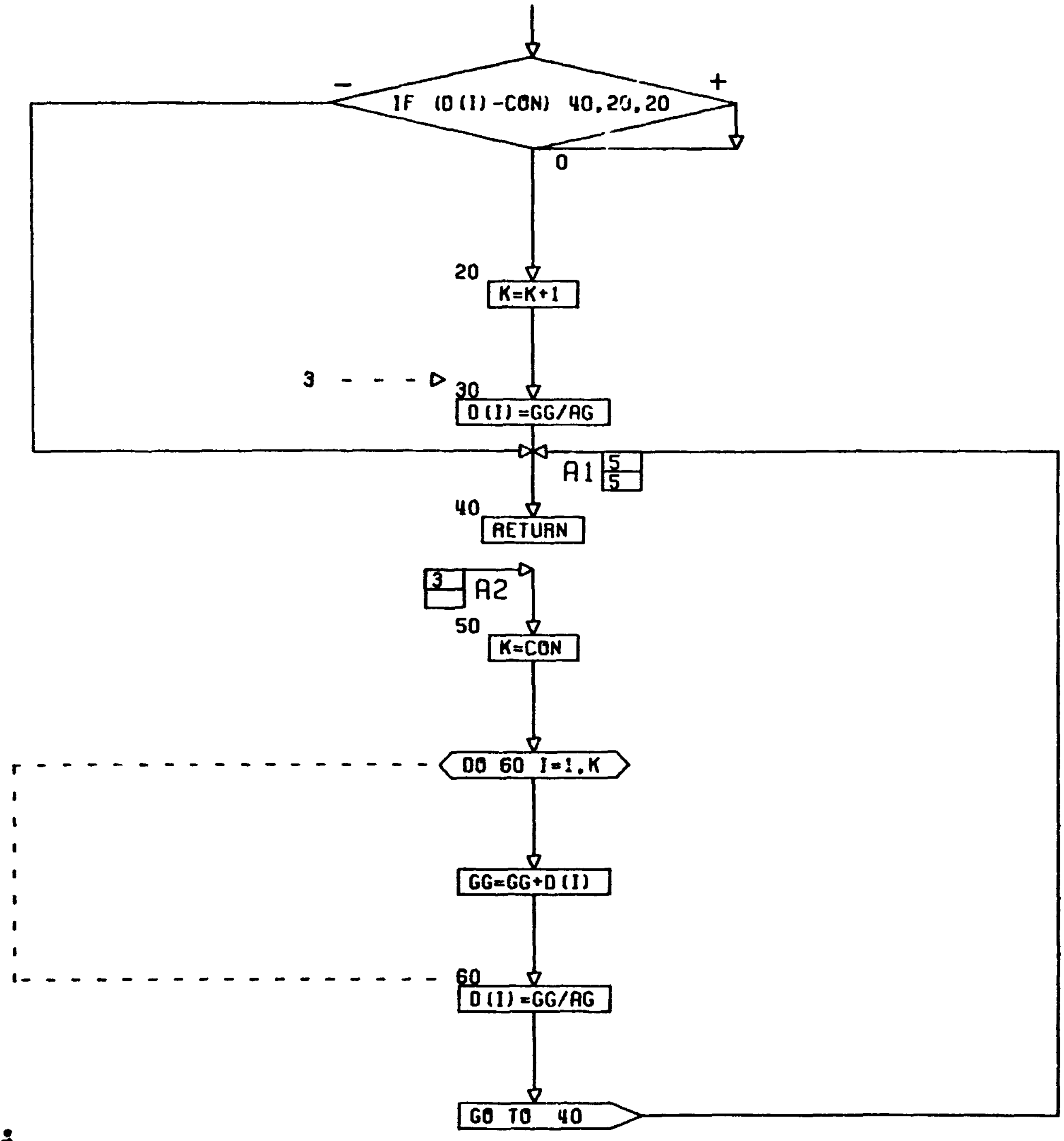
PG 2 OF 5

Figure 1-5 (Cont.)



104d

Figure 1-5 (Cont.)



CONT. ON PG 5

8069-104e

Figure 1-5 (Cont.)

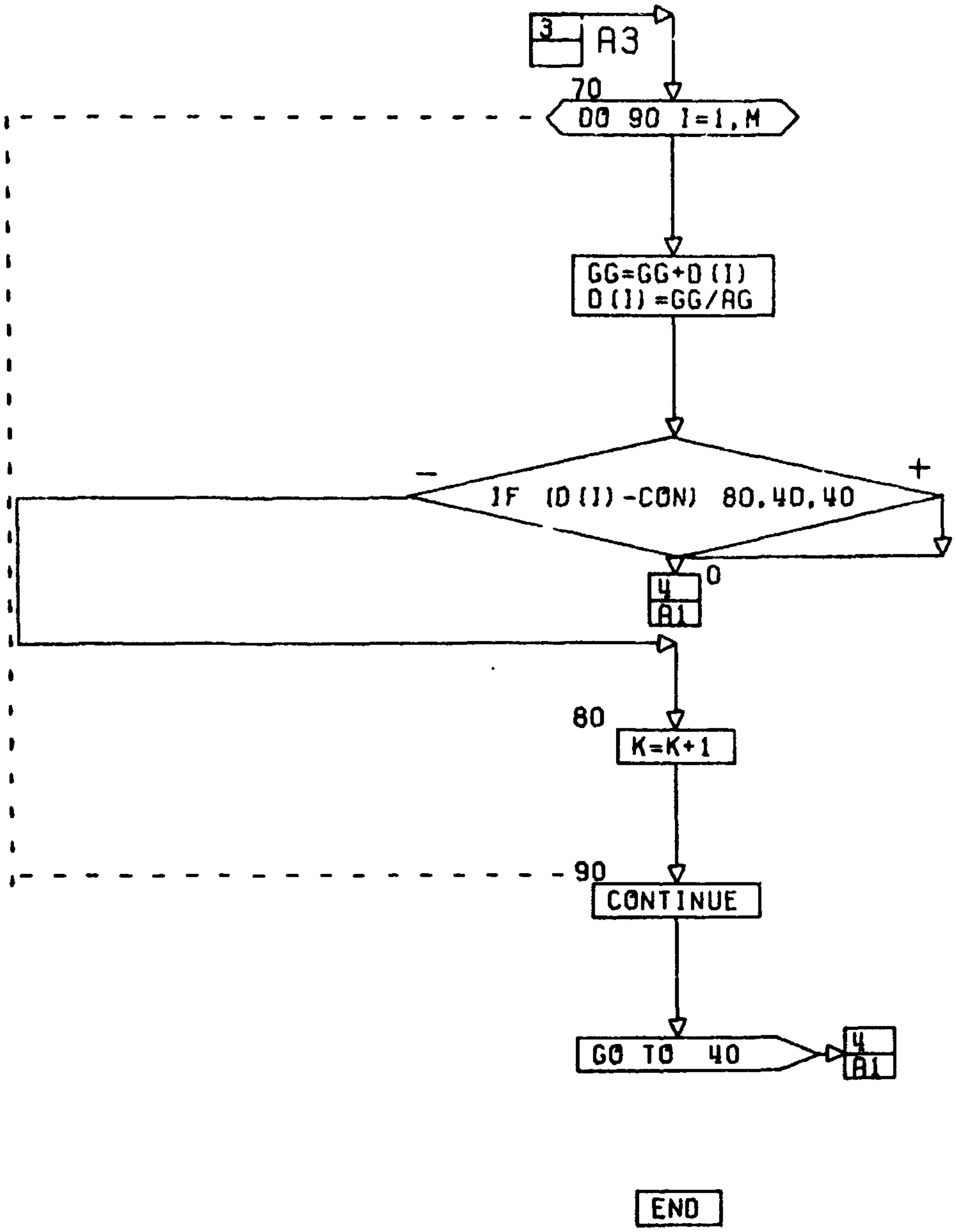


Figure 1-5 (Cont.)

8059-1041

KEY TO NOTATION

Subroutine LOAD

${}_{m}V_{k}$ = input eigenvectors, columnwise

${}_{m}B_{k}$ = factor matrix

λ_{i}, i=1, k = eigenvalues.

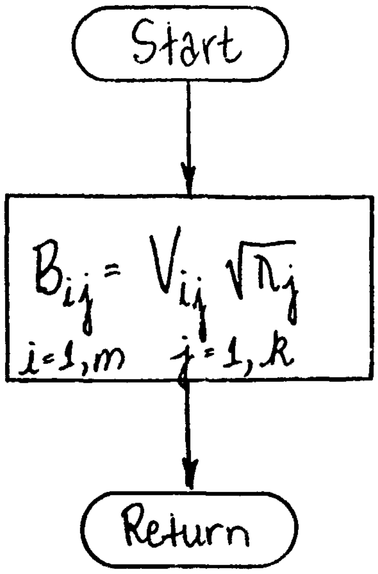
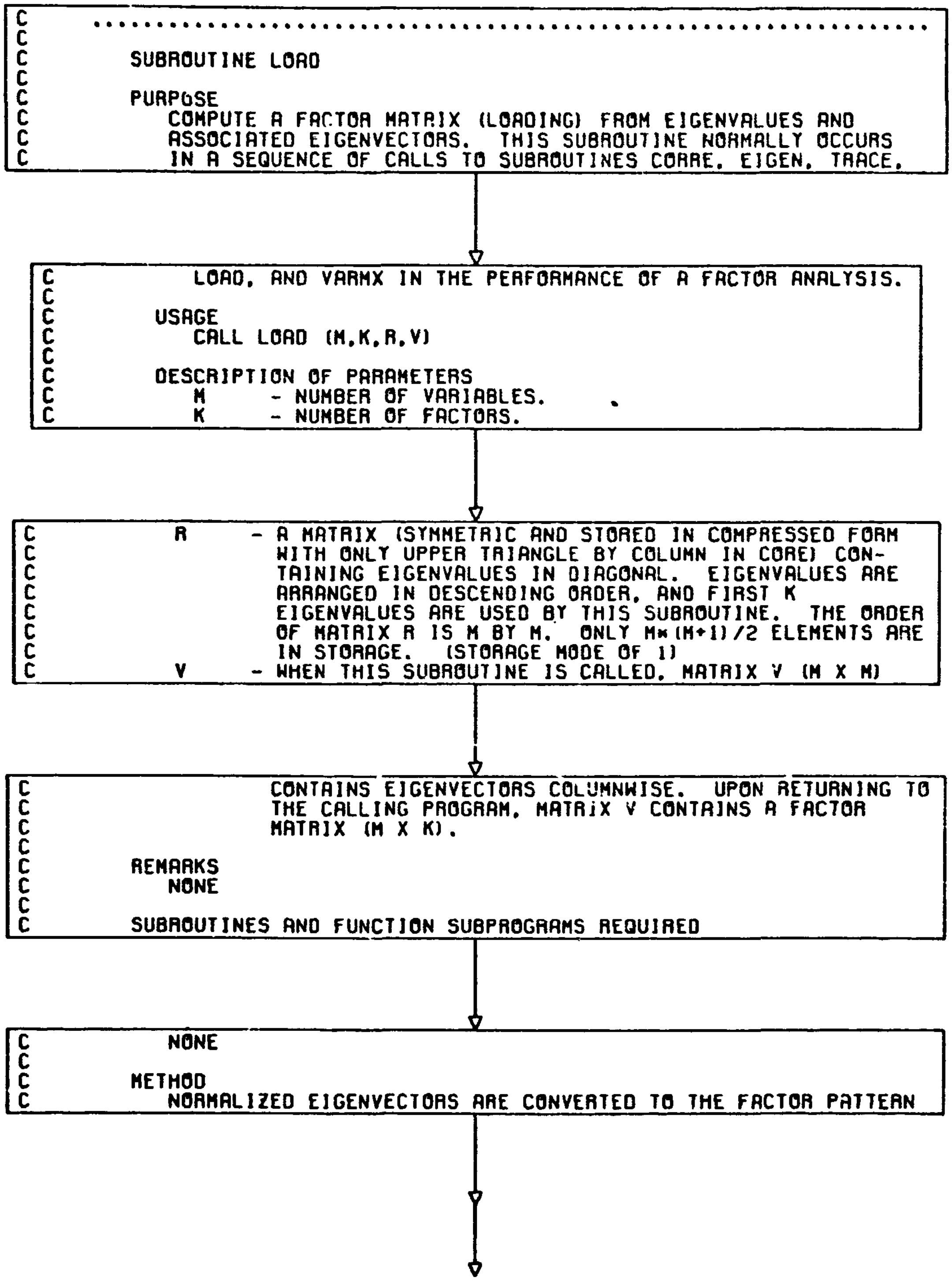


Figure 1-6 Subroutine LOAD



```
C     ...............................................................
C
C        SUBROUTINE LOAD
C
C        PURPOSE
C           COMPUTE A FACTOR MATRIX (LOADING) FROM EIGENVALUES AND
C           ASSOCIATED EIGENVECTORS.  THIS SUBROUTINE NORMALLY OCCURS
C           IN A SEQUENCE OF CALLS TO SUBROUTINES CORRE, EIGEN, TRACE,
```

```
C           LOAD, AND VARMX IN THE PERFORMANCE OF A FACTOR ANALYSIS.
C
C        USAGE
C           CALL LOAD (M,K,R,V)
C
C        DESCRIPTION OF PARAMETERS
C           M      - NUMBER OF VARIABLES.
C           K      - NUMBER OF FACTORS.
```

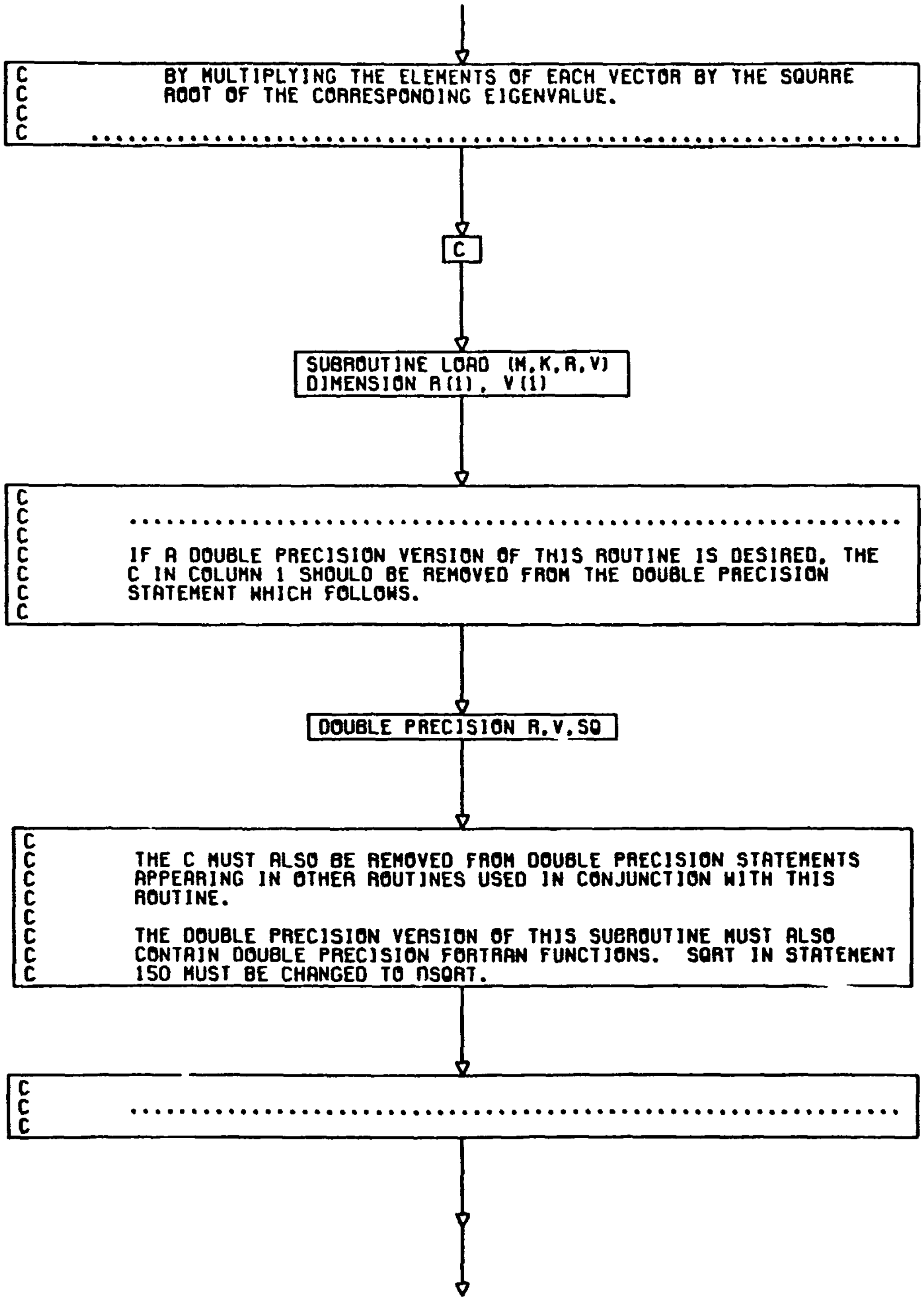
```
C           R      - A MATRIX (SYMMETRIC AND STORED IN COMPRESSED FORM
C                    WITH ONLY UPPER TRIANGLE BY COLUMN IN CORE) CON-
C                    TAINING EIGENVALUES IN DIAGONAL.  EIGENVALUES ARE
C                    ARRANGED IN DESCENDING ORDER, AND FIRST K
C                    EIGENVALUES ARE USED BY THIS SUBROUTINE.  THE ORDER
C                    OF MATRIX R IS M BY M.  ONLY M*(M+1)/2 ELEMENTS ARE
C                    IN STORAGE.  (STORAGE MODE OF 1)
C           V      - WHEN THIS SUBROUTINE IS CALLED, MATRIX V (M X M)
```

```
C                    CONTAINS EIGENVECTORS COLUMNWISE.  UPON RETURNING TO
C                    THE CALLING PROGRAM, MATRIX V CONTAINS A FACTOR
C                    MATRIX (M X K).
C
C        REMARKS
C           NONE
C
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
```

```
C           NONE
C
C        METHOD
C           NORMALIZED EIGENVECTORS ARE CONVERTED TO THE FACTOR PATTERN
```

CONT. ON PG 2

Figure 1-6 (Cont.)



```
C          BY MULTIPLYING THE ELEMENTS OF EACH VECTOR BY THE SQUARE
C          ROOT OF THE CORRESPONDING EIGENVALUE.
C
C     ..................................................................

C

SUBROUTINE LOAD (M,K,R,V)
DIMENSION R(1), V(1)

C
C        ...............................................................
C
C        IF A DOUBLE PRECISION VERSION OF THIS ROUTINE IS DESIRED, THE
C        C IN COLUMN 1 SHOULD BE REMOVED FROM THE DOUBLE PRECISION
C        STATEMENT WHICH FOLLOWS.
C

DOUBLE PRECISION R,V,SQ

C
C        THE C MUST ALSO BE REMOVED FROM DOUBLE PRECISION STATEMENTS
C        APPEARING IN OTHER ROUTINES USED IN CONJUNCTION WITH THIS
C        ROUTINE.
C
C        THE DOUBLE PRECISION VERSION OF THIS SUBROUTINE MUST ALSO
C        CONTAIN DOUBLE PRECISION FORTRAN FUNCTIONS.  SQRT IN STATEMENT
C        150 MUST BE CHANGED TO DSQRT.

C
C        ...............................................................
C
```

CONT. ON PG 3

Figure 1-6 (Cont.)

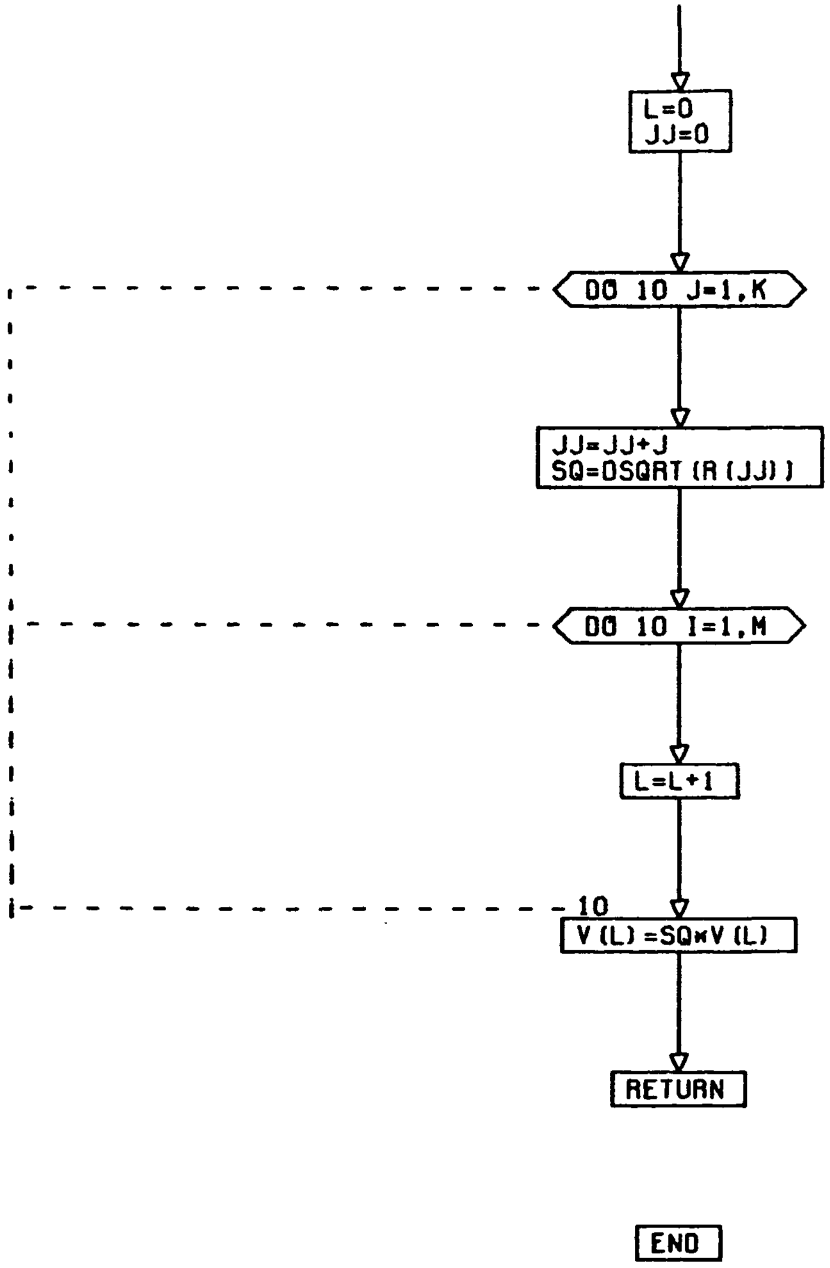


Figure 1-6 (Cont.)

1.3 CLASSIFICATION AND INFORMATION DIVERGENCE PROGRAMS

At the base of all of the classification and information divergence programs is the computation of the probability density function computed from the training sets evaluated at each sample signature. The Bayesian Decision Function Program and the Information Divergence Program provide several options for probability density computation. In contrast, the Density Lister, Decision Ruler, and Density Divergence Program make restricted use of the hyperpyramid smoothing function for the density computation.

The probability density function of signatures for objects of class k, $f_k(\vec{x})$, can be evaluated in one of several ways.

The univariate normal probability density computation is based upon population parameters:

$$f_k(x_j) = \frac{1}{\sqrt{2\pi}\,\sigma_{jk}} \exp\left(-1/2\,(x_j - \bar{x}_{jk})^2/\sigma_{jk}^2\right) \qquad (1\text{-}5)$$

where

j is the channel index

k is the population (target) index.

The multivariate normal probability density computation is:

$$f_k(\vec{x}) = \frac{|A_k|^{1/2}}{(2\pi)^{m/2}} \exp\left(-1/2\,\vec{Y}^T A_k \vec{Y}\right), \qquad (1\text{-}6)$$

where

m is the number of channels

$\vec{Y}$ is the signature $\vec{x}$ expressed about the kth training set mear

A_k is the inverse covariance matrix of the kth training set.

The Specht function is a smoothing method for weighting the contribution of each point in a data sample to the total density function evaluated at a point not in the sample. A smoothing function is used to compute the weight given to each sample point. Donald Specht* concluded that an exponential smoothing function should be used and subsequently expanded in polynomial form. The polynomial expansion, however, is unsuitable for this study. The Specht smoothing function used throughout the study is:

$$f_k(\vec{x}) = \frac{1}{N_k(2\pi)^{m/2}\left(\prod_{i=1}^{m} S_{ik}\right)^{1/2}} \sum_{j=1}^{N_k} \exp\left(-\frac{1}{2}\sum_{i=1}^{m} \frac{(x_i - y_{ijk})^2}{S_{ik}}\right) \qquad (1\text{-}7)$$

where

N_k is the number of observations in the k^{th} training set

S_{ik} is the Specht smoothing parameter for the i^{th} training set

y_{ijk} is the i^{th} component of the j^{th} signature of the k^{th} training set

x_i is the i^{th} component of $\vec{x}$.

The exponential smoothing is extremely time consuming. In an effort to reduce the computation time, the hyperpyramid smoothing function** is implemented. The hyperpyramid smoothing function is:

*Specht, Donald F., "Generation of Polynomial Discriminant Functions for Pattern Recognition," Stanford Electronics Laboratories, Stanford, Calif., Tech. Report No. 6764-5, May 1966.

**Smoothing functions are discussed in Section 4.3, Vol. I.

$$f_k(\vec{x}) = \frac{1}{2^{m-1}\, N_k\, (S_{1k}\, S_{2k}\, \ldots\, S_{mk})} \sum_{j=1}^{N_k} g(\vec{x}, \vec{Y}_{jk})$$

where

$$g(\vec{x}, \vec{Y}_{jk}) = 1 - \max \left\{ \left| \frac{x_i - y_{ijk}}{S_{ik}} \right| , \; i = 1, m \right\} \quad \text{for}$$

$$\max \left\{ \left| \frac{x_i - y_{ijk}}{S_{ik}} \right| , \; i = 1, m \right\} \leq 1, \tag{1-8}$$

$$g(\vec{x}, \vec{Y}_j) = 0 \text{ for } \max \left\{ \left| \frac{x_i - y_{ijk}}{S_{ik}} \right| , \; i = 1, m \right\} > 1.$$

1.3.1 Bayesian Decision Function Program

The Bayesian Decision Function Program assigns arbitrary signatures from sample sets to one of the targets characterized by the training sets. The assignment is based upon the probability density functions computed from the training sets evaluated at each sample signature, the a priori probability occurrence of each target, and the penalties of misclassification. The program will compute density functions in one of six ways and will accept training set data in one of four forms.

The options for probability density function computation are as follows:

1. As a product of univariate normal probability densities, using means and standard deviations of each channel or factor from the training sets.

2. As a product of probability densities from univariate histograms.

3. As multivariate normal probability density, using the covariance matrix of the training sets.

4. and 5. As a multivariate Specht function or product of bivariate Specht functions, computed by smoothing the training set data. The density function may be computed over all channels simultaneously, or over pairs of components in which case, the probability density for the observation signature is the product of densities for pairs. The bivariate option can actually be used to subselect any number of channels.

6. As a multivariate hyperpyramid smoothing function.

Training set data may be in the form of digitized scanner data, continuous or factor score data, univariate histograms, or parametric (means and standard deviations) form. Sample data for classification can be digitized scanner data or continuous data.

Training set forms can be used in conjunction with density computation options according to the following table.

TABLE 1-1

USE OF TRAINING SET FORMS WITH DENSITY COMPUTATIONS

Training Set Form	Possible Density Computation
Digitized Scanner Data or Continuous Data	Product of Univariate Normal Bivariate from Specht Multivariate from Specht Hyperpyramid Smoothing Multivariate Normal
Univariate Histograms	Product of Univariate Normal Product of Univariate Histograms
Parametric	Product of Univariate Normal

The READ subroutine reads training set data and computes from these data quantities needed for the chosen density computation option. COMPUT calculates the probability density function of each sample signature when attributed to each training set, while the main program calculates the Bayesian Decision Function and handles output of signature target assignment.

1.3.1.1 Bayesian Decision Function Mathematical Description

In Bayesian classification of arbitrary samples, the decision criterion is minimum cost. The expected cost of an experiment is:

$$E(w) = \sum_{k=1}^{K} \sum_{j=1}^{K} w(j,k)P(j,k) \qquad (1\text{-}9)$$

where

w is the cost of the experiment

w(j, k) is the cost of assigning an object of class j to class k

P(j, k) is the probability of assigning an object of class j to class k.

The probability P(j, k) can be expressed as the product of an a priori and a conditional probability:

$$P(j,k) = P(j)P(k|j), \qquad (1\text{-}10)$$

where

P(j) is the a priori probability of occurrence for objects of class j,

P(k|j) is the probability of assigning an object to class k provided it belongs to class j.

Let Γ_k be the set of signatures (a region in the space of signatures) which is attributed to class k. Then, by definition,

$$P(k|j) = \int_{\Gamma_k} f_j(\vec{x})\, d\vec{x}, \qquad (1\text{-}11)$$

where

$\vec{x}$ is an observed signature

$f_j(\vec{x})$ is the probability density function of signatures for objects of class j, or

$$P(k\ j) = \int_{\Omega} \Gamma_k\ (\vec{x})\ f_j\ (\vec{x})\ d\vec{x} \qquad (1\text{-}12)$$

where

Ω is the entire signature space

$\Gamma_k (\vec{x}) = 1$ for $\vec{x} \in \Gamma_k$

$\Gamma_k (\vec{x}) = 0$ for $\vec{x} \in (\Omega - \Gamma_k)$.

By substitution, the expected cost is then

$$\begin{aligned} E(w) &= \sum_{k=1}^{K} \sum_{j=1}^{K} w(j,k)P(j) \int_{\Omega} \Gamma_k (\vec{x})\ f_j\ (\vec{x})\ d\vec{x} \qquad (1\text{-}13) \\ &= \int_{\Omega} \sum_{k=1}^{K} \Gamma_k (\vec{x}) \left[\sum_{j=1}^{K} w(j,k)P(j)\ f_j\ (\vec{x}) \right] d\vec{x}. \end{aligned}$$

Minimization of cost occurs when the cost function in square brackets is evaluated for each k and $\Gamma_k (\vec{x})$ is set to 1 for the k associated with the minimum cost.

1.3.1.2 Flow Chart of the Bayesian Decision Function Program

The following flow chart (Figure 1-7) makes reference to Subroutine MINV, inversion of a matrix, whose Fortran IV statement diagram only is provided. Refer to Figures 1-8 and 1-9 for subroutines READ and COMPUT.

KEY TO NOTATION

Main Program, Bayesian Decision Function

m	= number of channels
mr	= number of training sets (targets)
ms	= number of samples to be classified

$nr_k, k=1, mr$ = number of signatures in k^{th} training set

$ns_k, k=1, ms$ = number of signatures in k^{th} sample set

mk = number of subsets of channels

mn = number of channels in each subset of channels

$N_{ij}, i=1, mr, j=1, mk*mn$ = channel numbers in the subsets of channels for i^{th} training set

$P_i, i=1, mr$ = a priori probability of occurrence of target i

$C_{ij}, i=1, mr, j=1, mr$ = cost of attributing a signature of target i to target j

$S_{ij}, i=1, m, j=1, mr$ = smoothing parameters for the i^{th} channel of the j^{th} training set (for Specht function or hyperpyramid function)

$dmax_i, i=1, mr$ = upper bound on truncation error for computing the probability density function of a signature attributed to the i^{th} training set (for Specht function)

$du_i, i=1, mr$ = denominator of the univariate normal probability density function for the i^{th} training set

$sd_{ij}, i=1, m, j=1, mr$ = standard deviation in the i^{th} channel of the j^{th} training set

$dnm_{ij}, i=1, mk, j=1, mr$ = denominator of the probability density function for the i^{th} subset of channels of the j^{th} training set (for Specht function or hyperpyramid function)

$smax_{ij}, i=1, mk, j=1, mr$ = upper limit on sum of squares for truncation of computation of density function (for Specht function)

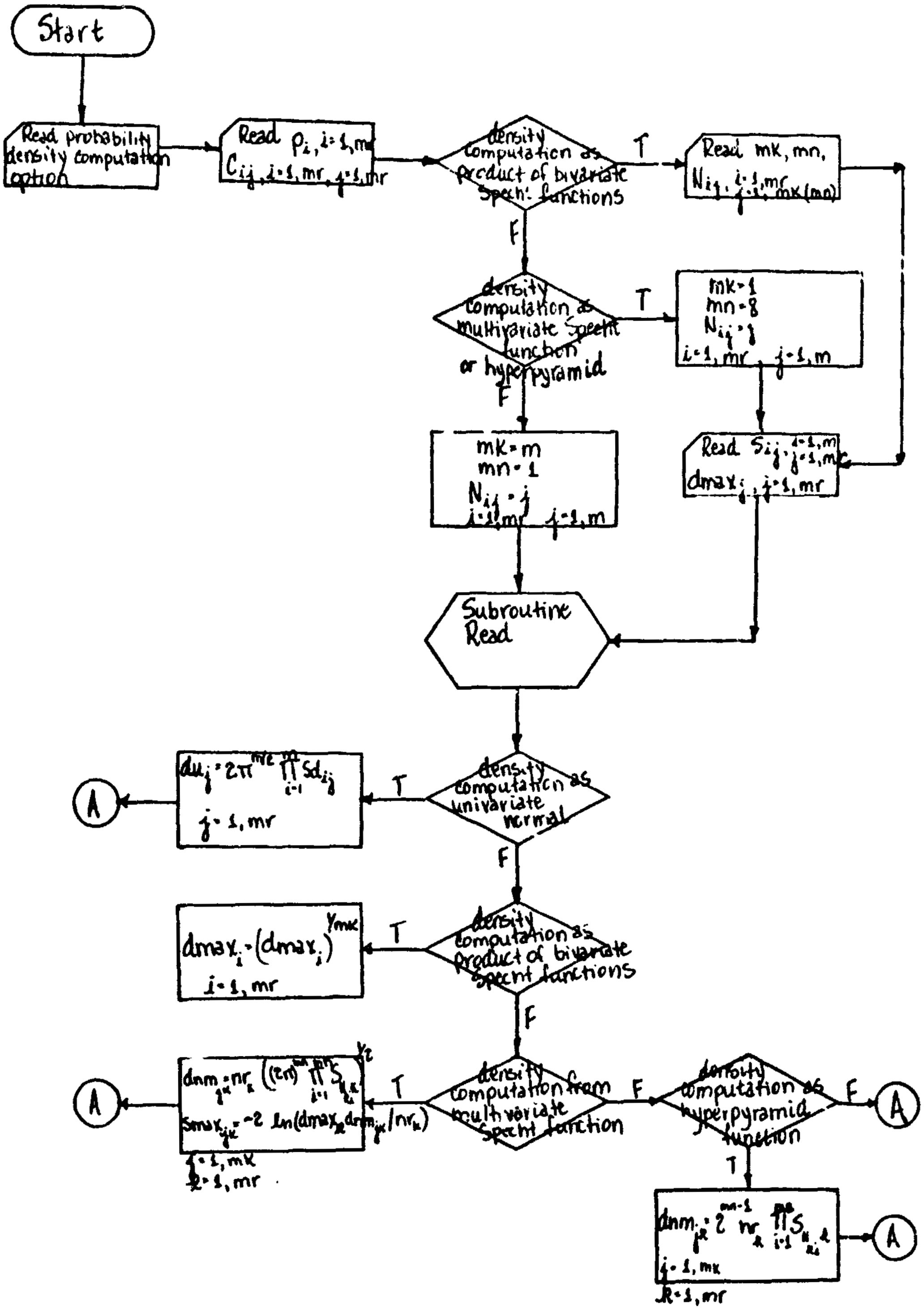
$X_{ijk}, i=1, m, j=1, nr_k, k=1, mr$ = j^{th} signature of k^{th} training set

$Y_i, i=1, m$ = signature of sample set

$f_i, i=1, mr$ = probability density function of a sample signature when attributed to the i^{th} training set

$bdf_i, i=1, mr$ = Bayesian decision function of i^{th} training set

$pr_i, i=1, mr$ = Bayesian probability that signature is occurrence of target i.


Start
Read probability density computation option
Read p_i, $i=1,mr$ C_{ij}, $i=1,mr$, $j=1,mr$
density computation as product of bivariate Specht functions
T
F
Read mk, mn, N_{ij}, $i=1,mr$ $j=1,mk(mn)$
density computation as multivariate Specht function or hyperpyramid
T
F
mk=1 mn=8 $N_{ij}=j$ $i=1,mr$, $j=1,m$
mk=m mn=1 $N_{ij}=j$ $i=1,mr$ $j=1,m$
Read S_{ij}, $i=1,m$ $j=1,mr$ $dmax_j$, $j=1,mr$
Subroutine Read
density computation as univariate normal
T
F
$du_j = 2\pi^{m/2} \prod_{i=1}^{m} Sd_{ij}$ $j=1,mr$
A
density computation as product of bivariate Specht functions
T
F
$dmax_i = (dmax_i)^{1/mk}$ $i=1,mr$
density computation from multivariate Specht function
T
F
$dnm_{jk} = nr_k \left((2\pi)^{mn} \prod_{i=1}^{mn} S_{N_{ji}k}\right)^{1/2}$
$smax_{jk} = -2 \ln(dmax_k dnm_{jk}/nr_k)$
$j=1,mk$ $k=1,mr$
A
density computation as hyperpyramid function
F
T
A
$dnm_{jk} = 2^{mn-1} nr_k \prod_{i=1}^{mn} S_{N_{ji},k}$ $j=1,mk$ $k=1,mr$
A

8059-106a

Figure 1-7 Main Program, Bayesian Decision Function

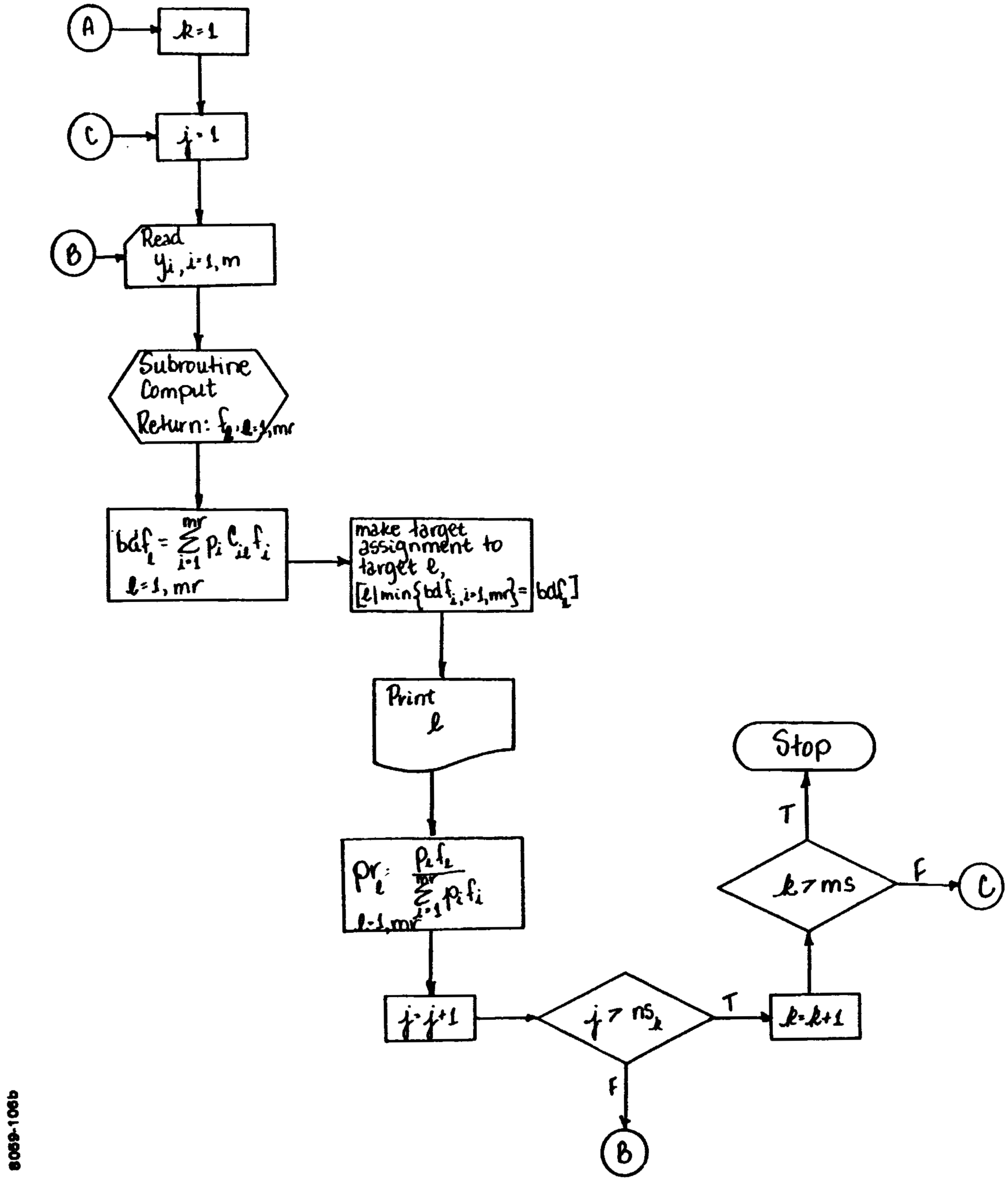

A
$k=1$
C
$j=1$
B
Read $y_i, i=1,m$
Subroutine Comput Return: $f_\ell, \ell=1,mr$
$bdf_\ell = \sum_{i=1}^{mr} p_i c_{i\ell} f_i$ $\ell=1, mr$
make target assignment to target ℓ, $[\ell | \min\{bdf_i, i=1,mr\} = bdf_\ell]$
Print ℓ
$pr_\ell = \frac{p_\ell f_\ell}{\sum_{i=1}^{mr} p_i f_i}$ $\ell=1,mr$
$j=j+1$
$j > ns_k$
T
F
B
$k=k+1$
$k > ms$
T
F
C
Stop
8059-106b

Figure 1-7 (Cont.)

KEY TO NOTATION

Subroutine READ

m = number of channels

$nr_k, k=1, mr$ = number of signatures in kth training set

$ni_k, k=1, mr$ = number of intervals in histogram of kth training set

$\overline{X}_{ik}, i=1, m, k=1, mr$ = mean of ith channel of the kth training set

$sd_{ik}, i=1, m, k=1, mr$ = standard deviation of the ith channel of the kth training set

$xmid_{ik}, i=1, m, k=1, mr$ = midpoint of the first interval of the histogram of the ith channel of the kth training set

$delt_{ik}, i=1, m, k=1, mr$ = length of interval of the histogram of the ith channel of the kth training set

$H_{ijk}, i=1, m, j=1, ni_k, k=1, mr$ = frequency in ith channel, jth interval, of histogram of kth training set

$det_k, k=1, mr$ = determinant of the covariance matrix of the kth training set

$D_k, k=1, mr$ = determinant of the inverse of the covariance matrix of the kth training set

$B_{ijk}, i=1, m, j=1, m, k=1, mr$ = m x m covariance matrix for kth training set

$A_{ijk}, i=1, m, j=1, m, k=1, mr$ = m x m inverse of covariance matrix for kth training set

$X_{ijk}, i=1, m, j=1, nr_k, k=1, mr$ = jth signature of kth training set

mr = number of training sets (targets)

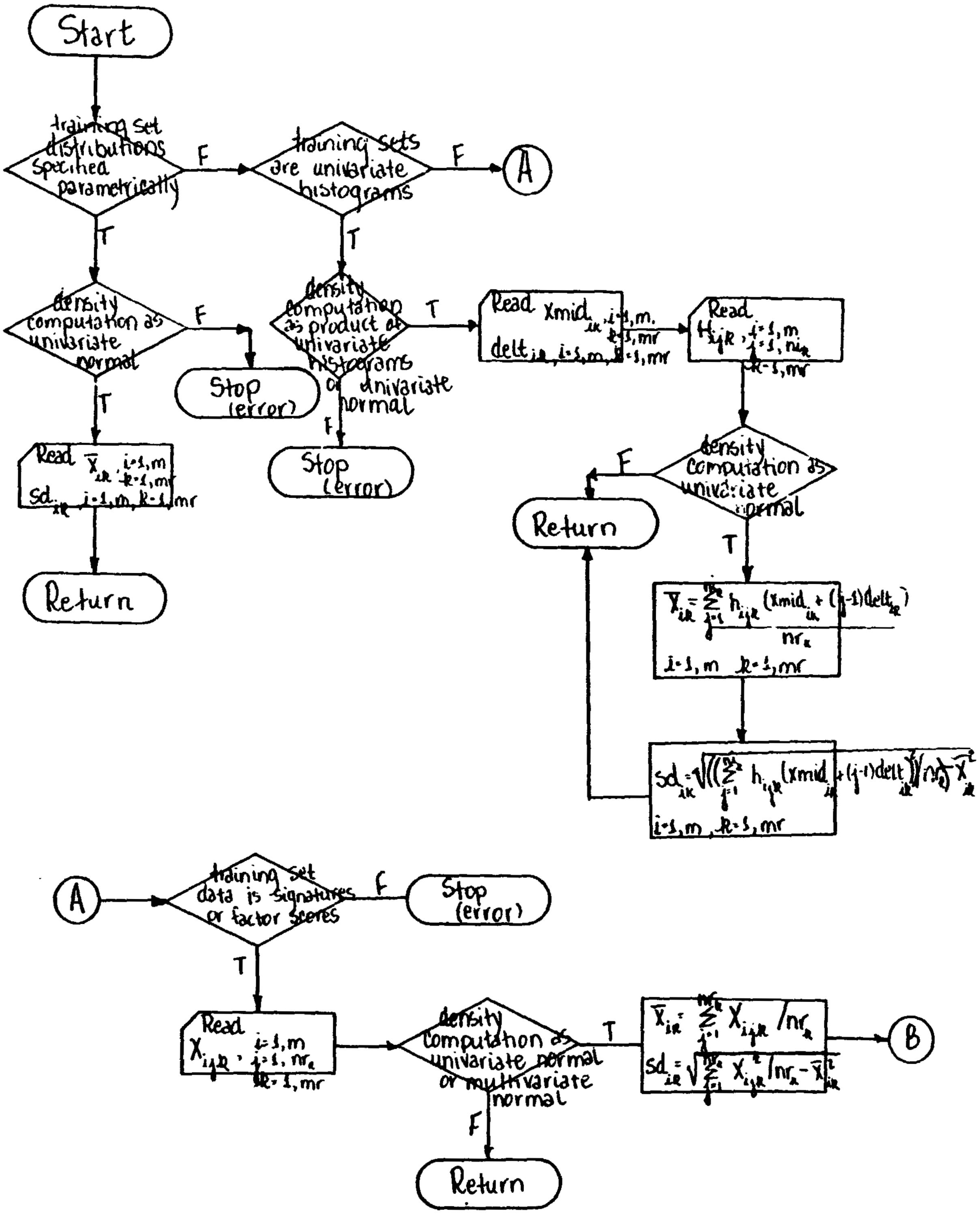

Start
training set distributions specified parametrically
F
training sets are univariate histograms
F
A
T
T
density computation as univariate normal
F
Stop (error)
T
Read $\bar{X}_{ik}$, i=1,m k=1,mr
sd_{ik} i=1,m, k=1,mr
Return
density computation as product of univariate histograms or univariate normal
T
F
Stop (error)
Read xmid$_{ik}$, i=1,m, k=1,mr
delt$_{ik}$, i=1,m, k=1,mr
Read H_{ijk}, i=1,m j=1,n_{ik} k=1,mr
density computation as univariate normal
F
Return
T
$\bar{X}_{ik} = \sum_{j=1}^{n_{ik}} h_{ijk}(\text{xmid}_{ik} + (j-1)\text{delt}_{ik}) / nr_k$
i=1,m k=1,mr
$sd_{ik} = \sqrt{\left(\left(\sum_{j=1}^{n_{ik}} h_{ijk}(\text{xmid}_{ik} + (j-1)\text{delt}_{ik})^2\right) / nr_k\right) - \bar{X}_{ik}^2}$
i=1,m, k=1,mr
A
training set data is signatures or factor scores
F
Stop (error)
T
Read X_{ijk}, i=1,m j=1,nr_k k=1,mr
density computation as univariate normal or multivariate normal
T
$\bar{X}_{ik} = \sum_{j=1}^{nr_k} X_{ijk} / nr_k$
$sd_{ik} = \sqrt{\sum_{j=1}^{nr_k} X_{ijk}^2 / nr_k - \bar{X}_{ik}^2}$
B
F
Return

Figure 1-8 Subroutine READ

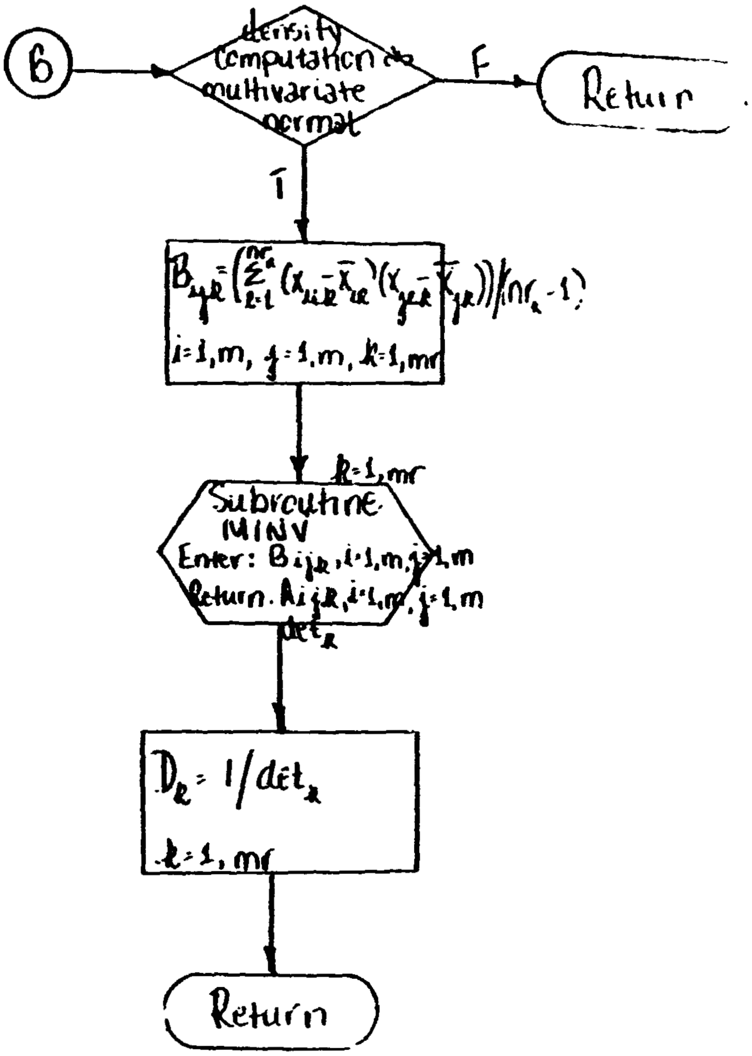
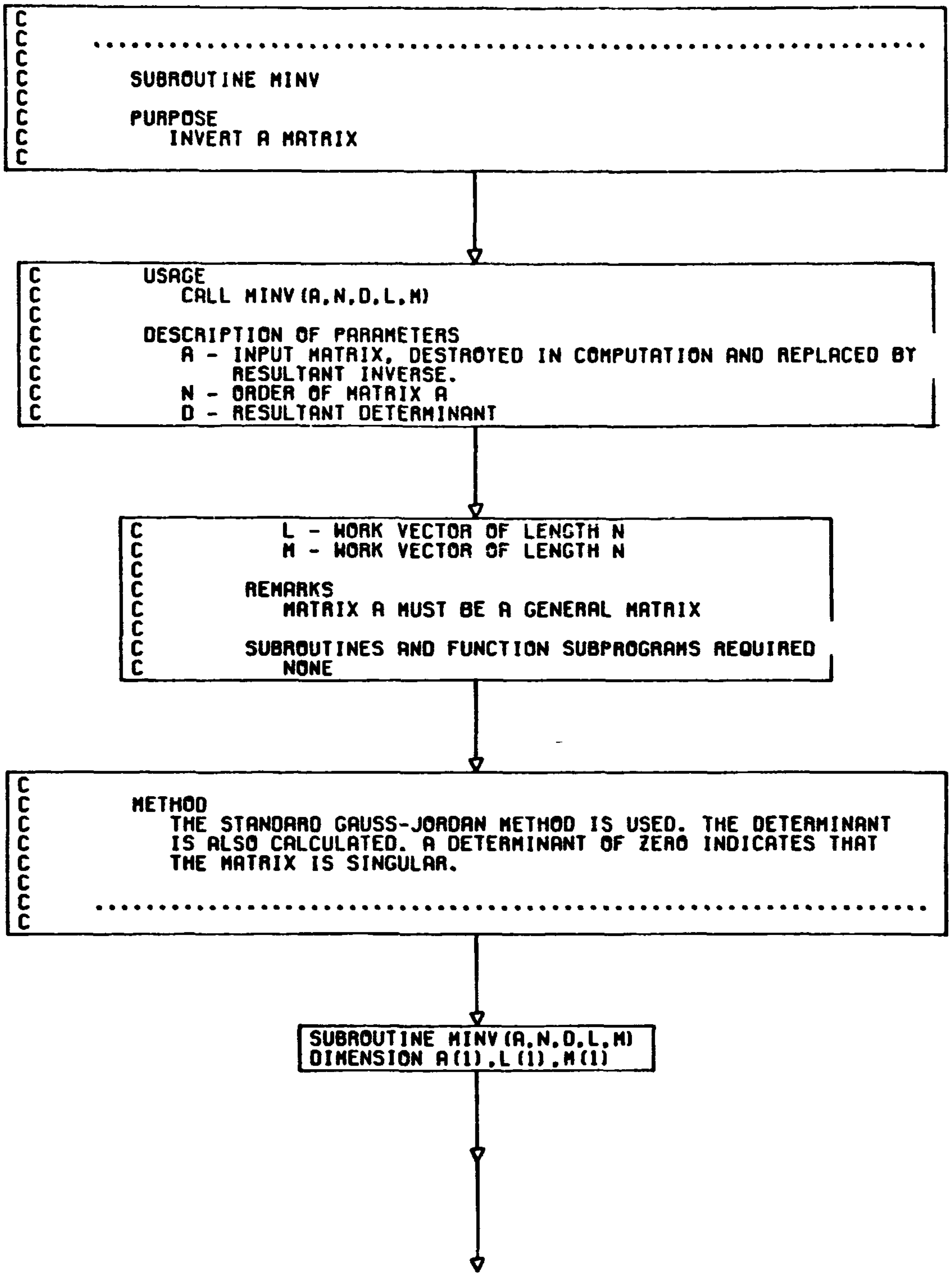

6
density computation do multivariate normal
F
Return
T
$B_{ijk} = \left(\sum_{l=1}^{nr_k}(x_{ilk} - \bar{x}_{ik})(x_{jlk} - \bar{x}_{jk})\right) / (nr_k - 1)$
i=1,m, j=1,m, k=1,mr
k=1,mr
Subroutine MINV
Enter: B_{ijk}, i=1,m, j=1,m
Return. A_{ijk}, i=1,m, j=1,m
det_k
$D_k = 1/det_k$
k=1,mr
Return

Figure 1-8 (Cont.)



```
C
C     ...............................................................
C
C        SUBROUTINE MINV
C
C        PURPOSE
C           INVERT A MATRIX
C
```

```
C        USAGE
C           CALL MINV(A,N,D,L,M)
C
C        DESCRIPTION OF PARAMETERS
C           A - INPUT MATRIX, DESTROYED IN COMPUTATION AND REPLACED BY
C               RESULTANT INVERSE.
C           N - ORDER OF MATRIX A
C           D - RESULTANT DETERMINANT
```

```
C           L - WORK VECTOR OF LENGTH N
C           M - WORK VECTOR OF LENGTH N
C
C        REMARKS
C           MATRIX A MUST BE A GENERAL MATRIX
C
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C           NONE
```

```
C
C        METHOD
C           THE STANDARD GAUSS-JORDAN METHOD IS USED. THE DETERMINANT
C           IS ALSO CALCULATED. A DETERMINANT OF ZERO INDICATES THAT
C           THE MATRIX IS SINGULAR.
C
C     ...............................................................
C
```

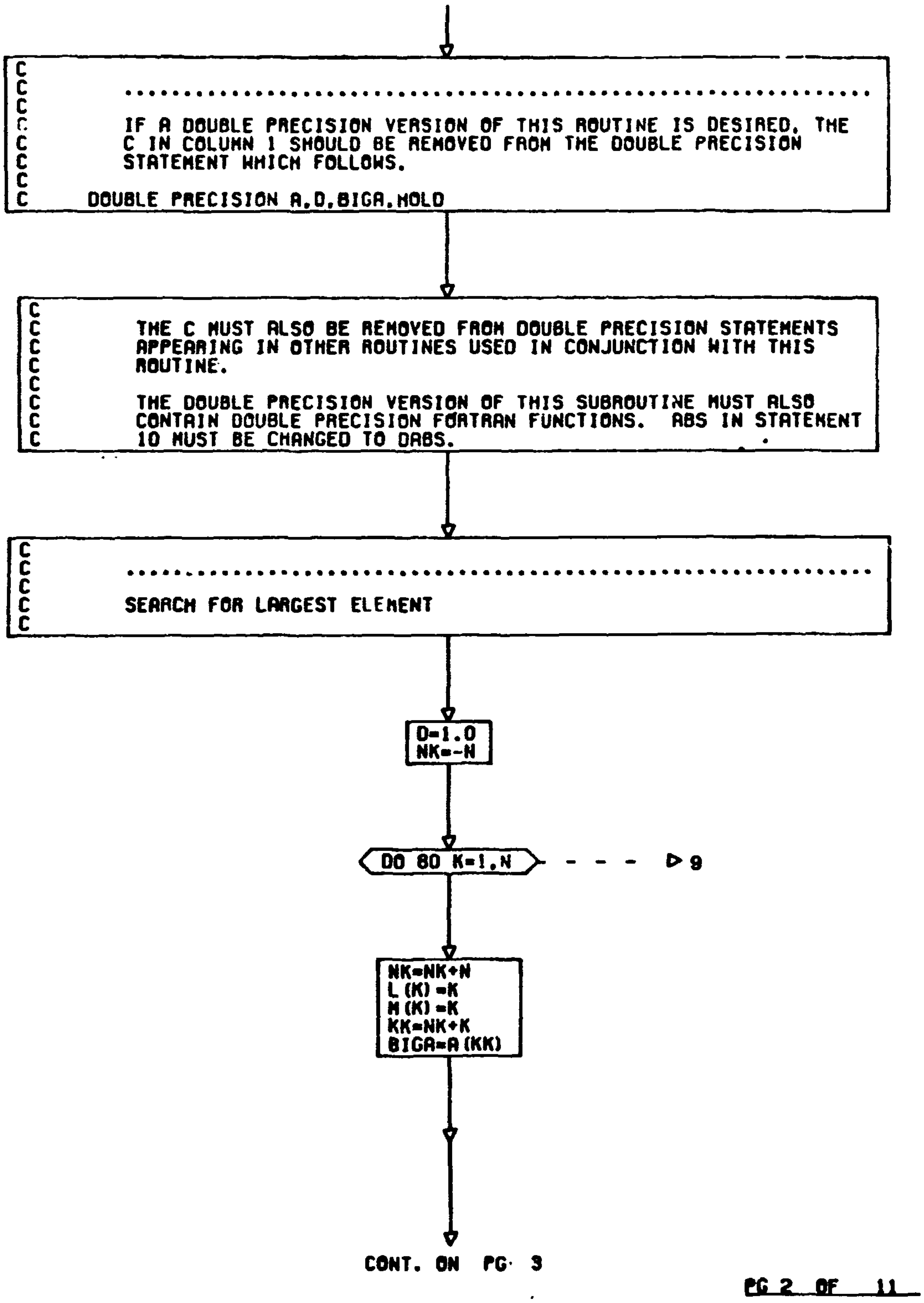
```
SUBROUTINE MINV(A,N,D,L,M)
DIMENSION A(1),L(1),M(1)
```

CONT. ON PG 2

PG 1 OF 11

8059-107c

Figure 1-8 (Cont.)



```
C
C        ............................................................
C
C        IF A DOUBLE PRECISION VERSION OF THIS ROUTINE IS DESIRED, THE
C        C IN COLUMN 1 SHOULD BE REMOVED FROM THE DOUBLE PRECISION
C        STATEMENT WHICH FOLLOWS.
C
C     DOUBLE PRECISION A,D,BIGA,HOLD
```

```
C
C        THE C MUST ALSO BE REMOVED FROM DOUBLE PRECISION STATEMENTS
C        APPEARING IN OTHER ROUTINES USED IN CONJUNCTION WITH THIS
C        ROUTINE.
C
C        THE DOUBLE PRECISION VERSION OF THIS SUBROUTINE MUST ALSO
C        CONTAIN DOUBLE PRECISION FORTRAN FUNCTIONS.  ABS IN STATEMENT
C        10 MUST BE CHANGED TO DABS.
```

```
C
C        ............................................................
C
C        SEARCH FOR LARGEST ELEMENT
C
```

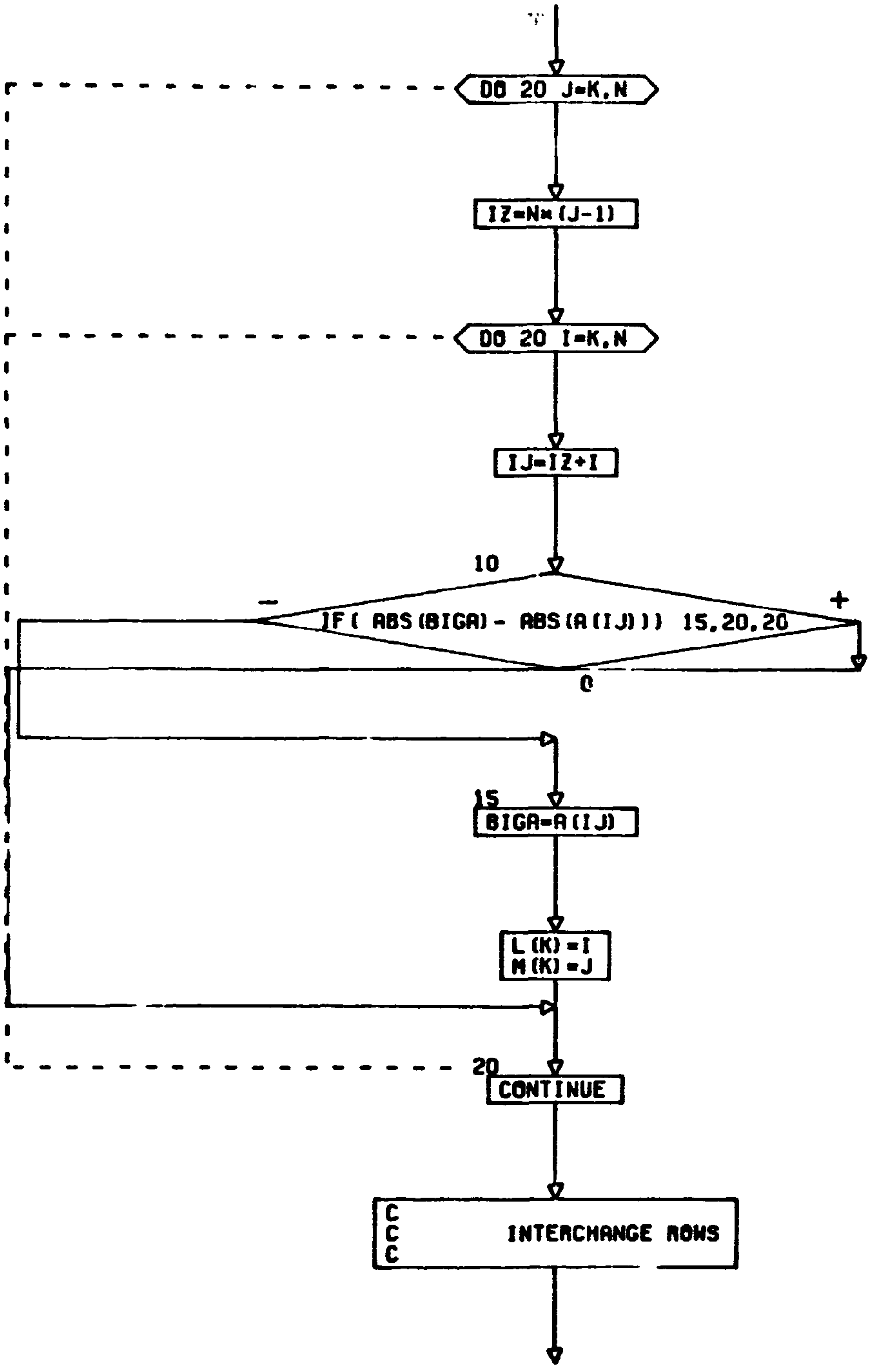
```
D=1.0
NK=-N
```

```
DO 80 K=1,N
```

```
NK=NK+N
L(K)=K
M(K)=K
KK=NK+K
BIGA=A(KK)
```

CONT. ON PG 3

Figure 1-8 (Cont.)



CONT. ON PG 4

PG 3 OF 11

Figure 1-8 (Cont.)

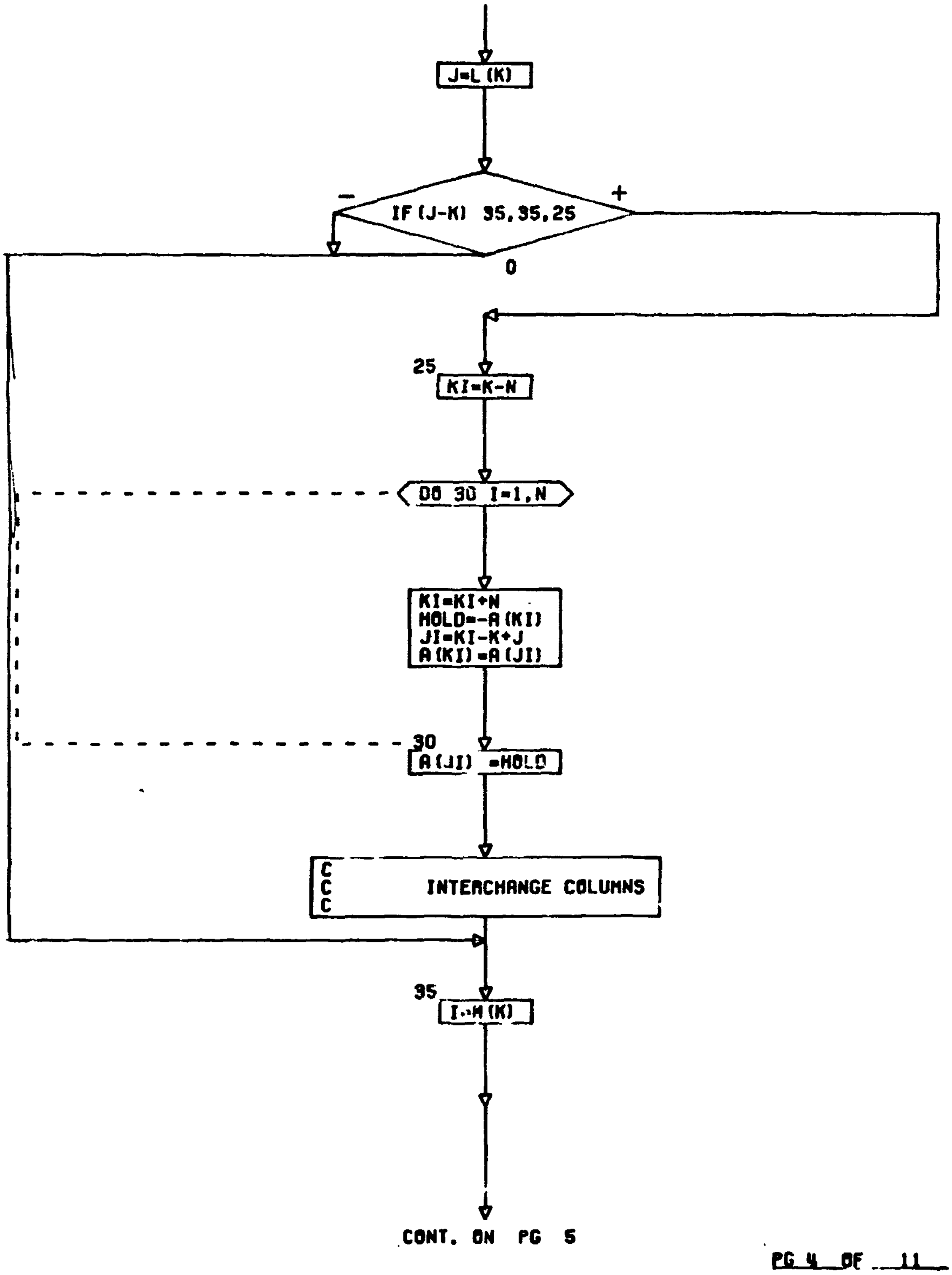


Figure 1-8 (Cont.)

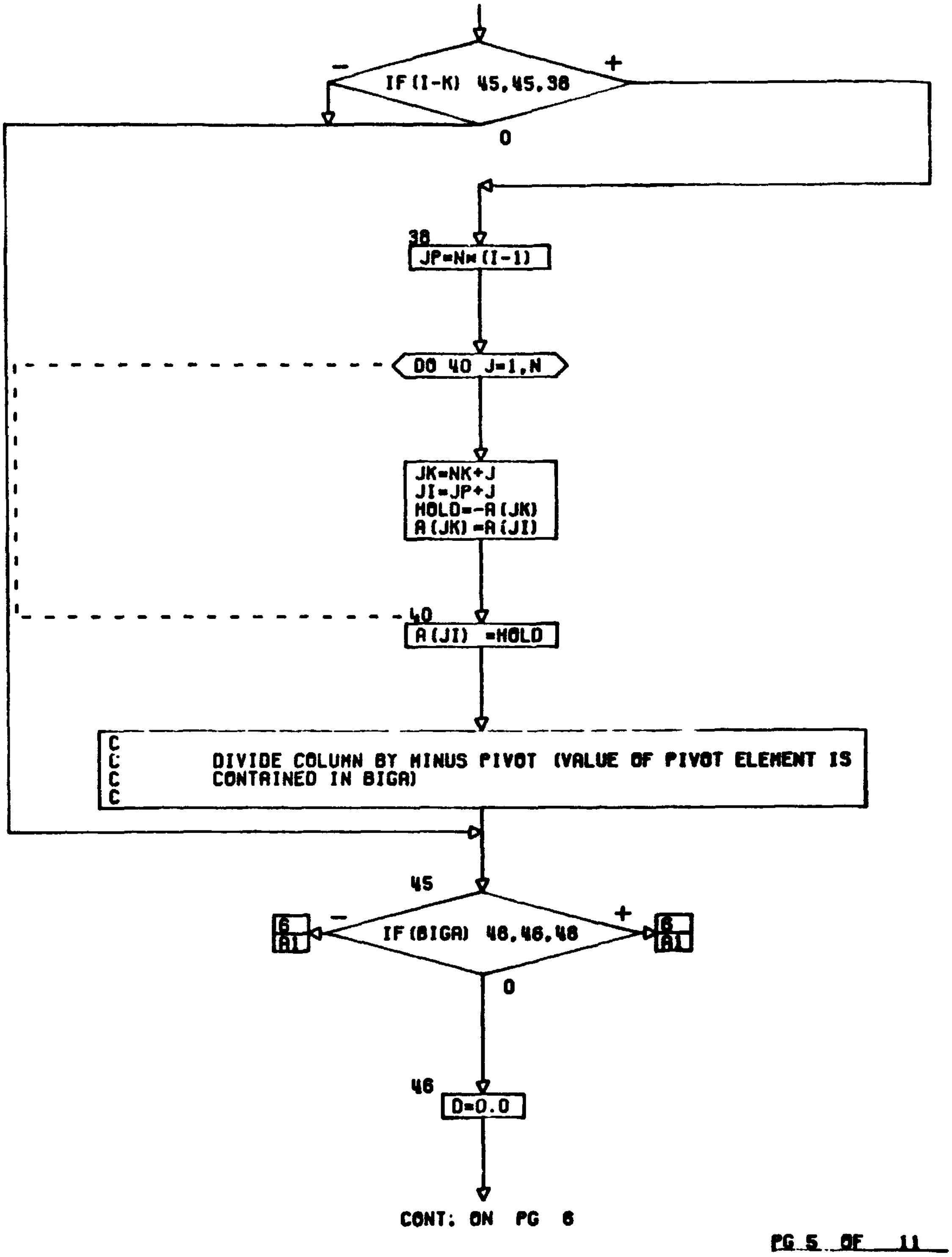


Figure 1-8 (Cont.)

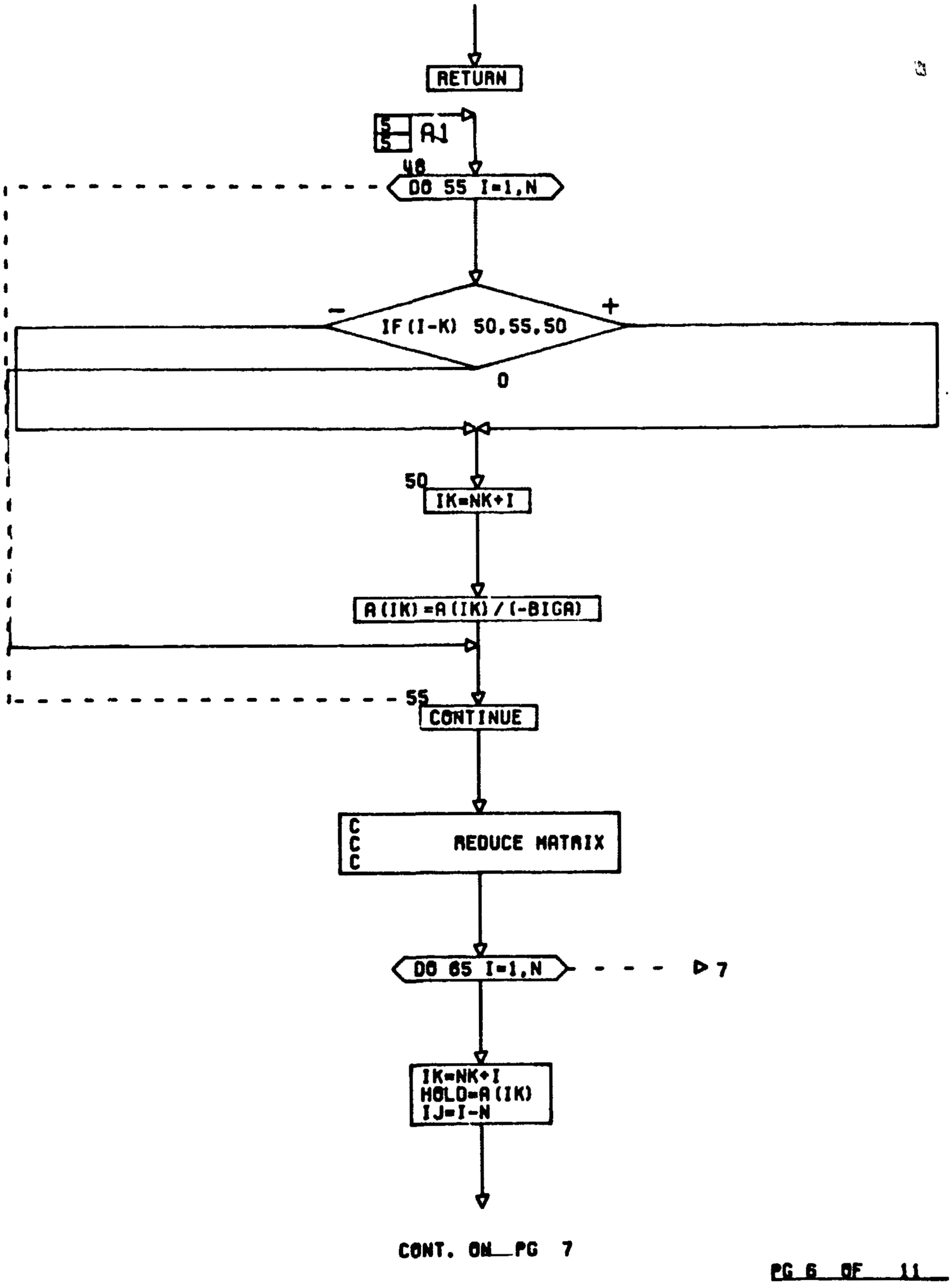
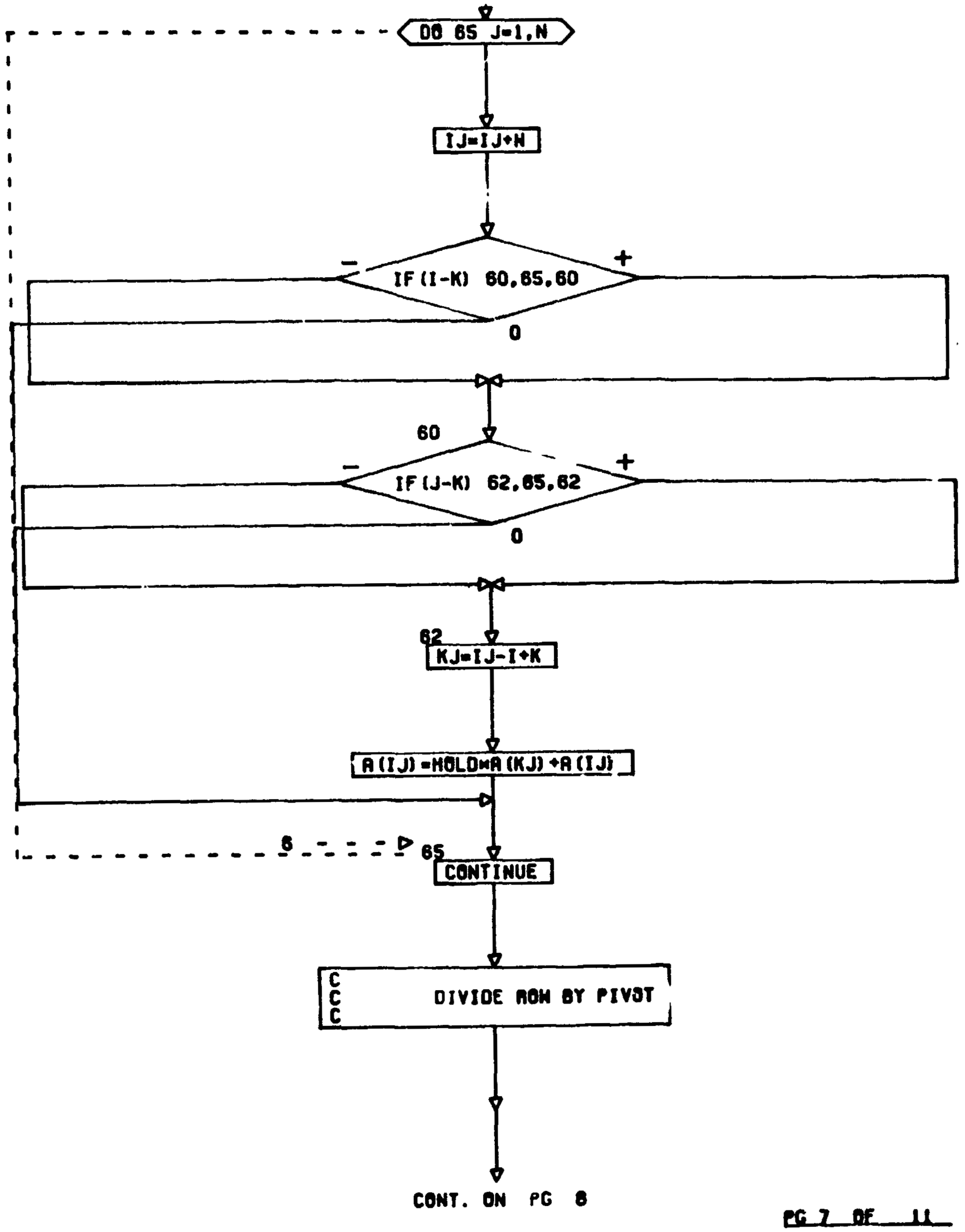


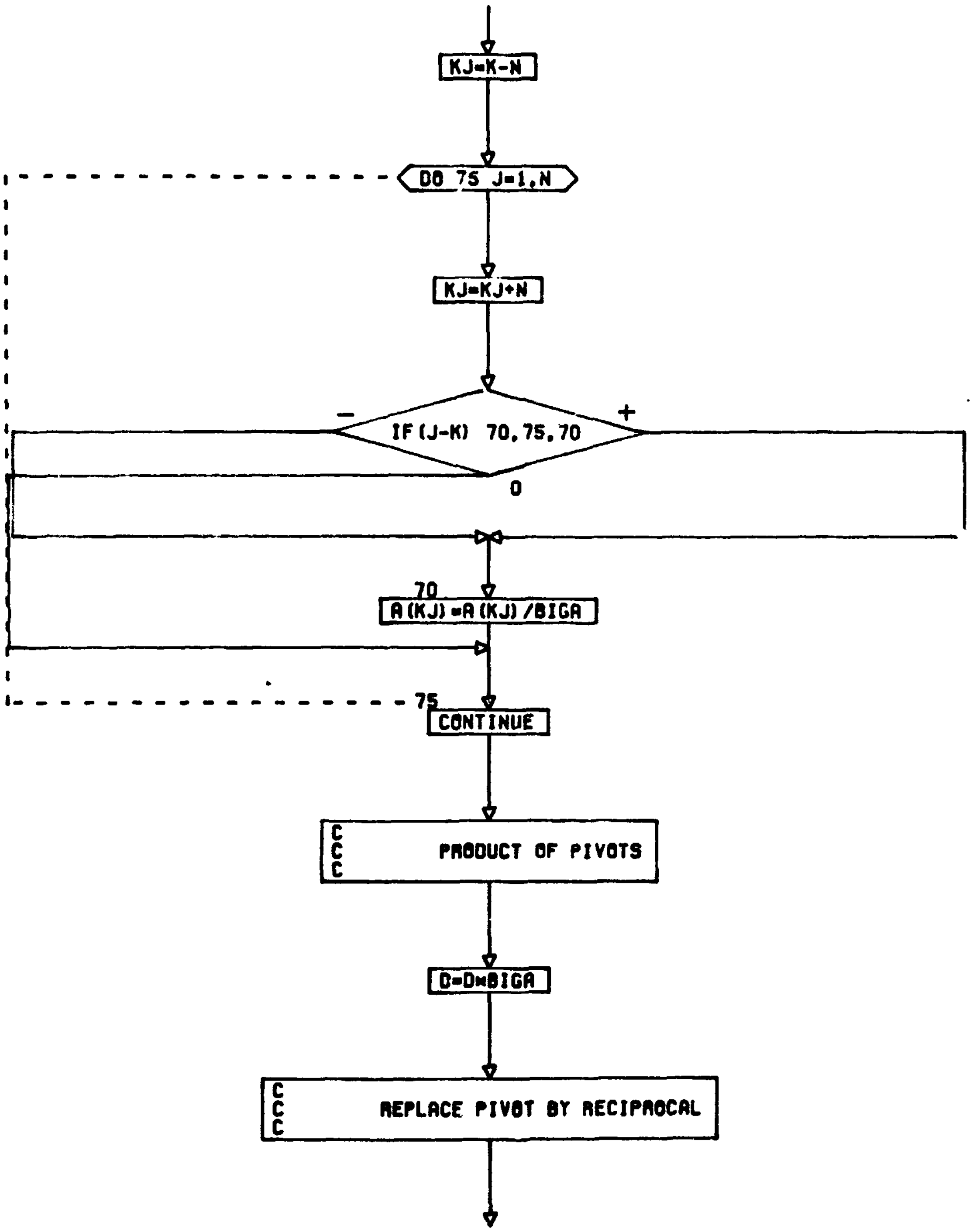
Figure 1-8 (Cont.)

8059-107h



8059-1071

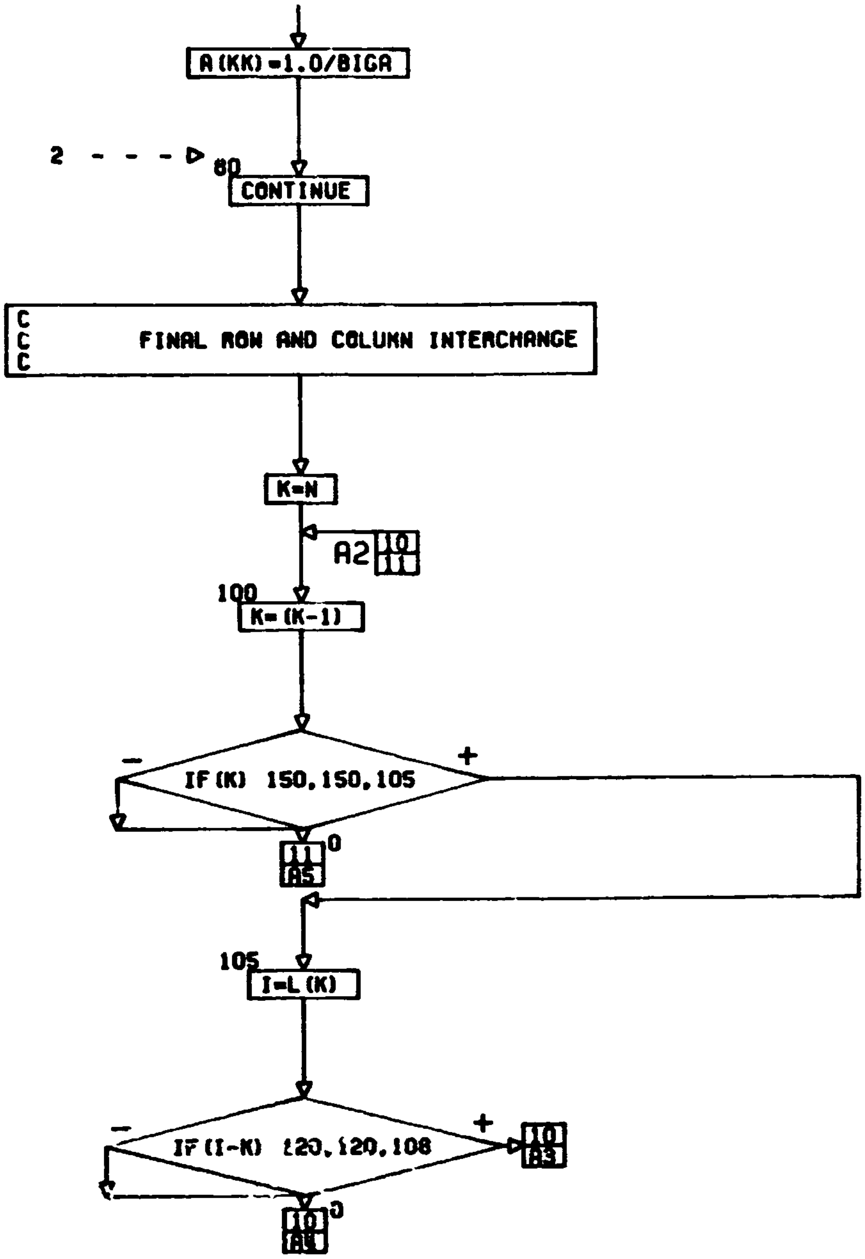
Figure 1-8 (Cont.)



CONT. ON PG 9

PG 8 OF 11

Figure 1-8 (Cont.)

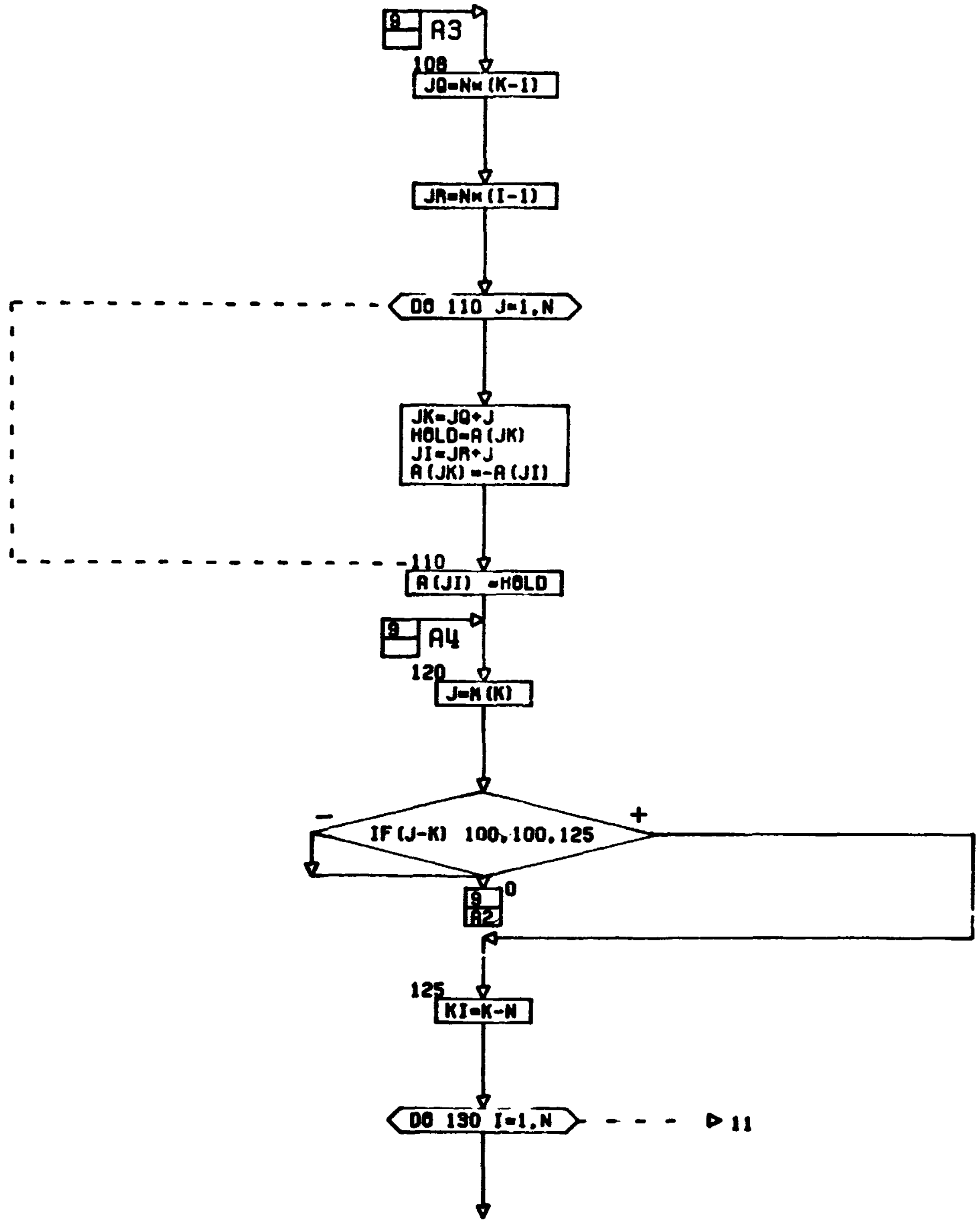


CONT. ON PG 10

PG 9 OF 11

8089-107k

Figure 1-8 (Cont.)



CONT. ON PG 11

PG 10 OF 11

Figure 1-8 (Cont.)

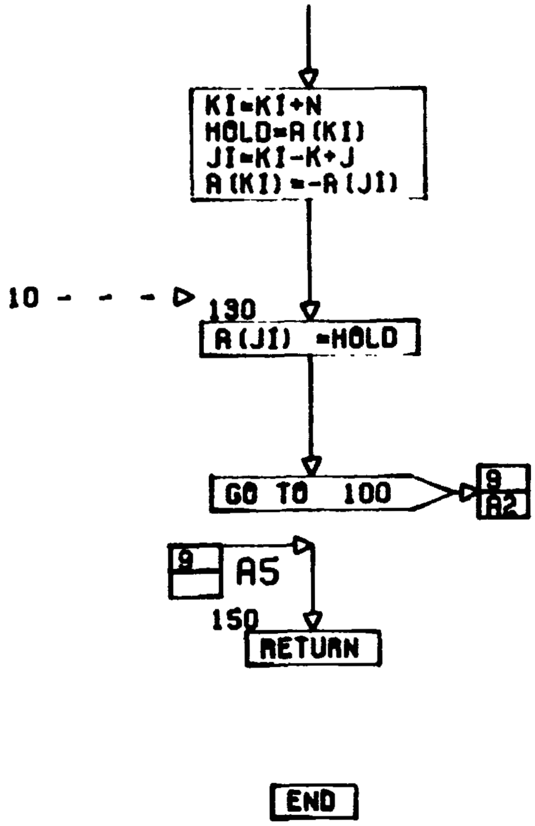


Figure 1-8 (Cont.)

KEY TO NOTATION

Subroutine COMPUT

m = number of channels

mr = number of training sets (targets)

mk = number of subsets of channels

mn = number of channels in each subset of channels

$ni_k, k=1, mr$ = number of intervals in histogram of k^{th} training set

$X_{ik}, i=1, m, k=1, mr$ = mean of i^{th} channel of k^{th} training set

$sd_{ik}, i=1, m, k=1, mr$ = standard deviation of i^{th} channel of k^{th} training set

$N_{ij}, i=1, mr, j=1, mk*mn$ = channel numbers in the subsets of channels of the i^{th} training set

$xmid_{ik}, i=1, m, k=1, mr$ = midpoint of the first interval of the histogram of the i^{th} channel of the k^{th} training set

$delt_{ik}, i=1, m, k=1, mr$ = length of interval in histogram of the i^{th} channel of the k^{th} training set

$H_{ijk}, i=1, m, j=1, ni_k, k=1, mr$ = frequency in j^{th} interval of histogram of i^{th} channel of k^{th} training set

$smax_{ij}, i=1, mk, j=1, mr$ = upper limit on sum of squares for truncation of computation of density function (Specht function)

$dnm_{ij}, i=1, mk, j=1, mr$ = denominator of the Specht function or hyper-pyramid function for the i^{th} subset of channels of the j^{th} training set

$du_i, i=1, mr$ = denominator of the univariate normal probability density function for the i^{th} training set

S_{ij}, i=1, m, j=1, mr = smoothing parameters for Specht function, or hyperpyramid function, i^{th} channel of j^{th} training set

A_{ijk}, i=1, m, j=1, m, k=1, mr = inverse of covariance matrix (m x m) for k^{th} training set

D_k, k=1, mr = determinant of A for k^{th} training set

Y_i, i=1, m = signature from sample set

X_{ijk}, i=1, m, j=1, nr_k, k=1, mr = j^{th} signature of k^{th} training set

f_i, i=1, mr = probability density function of a sample signature when attributed to the i^{th} training set

fi_k, k=1, mk = probability density function for k^{th} subset of channels

nr_k, k=1, mr = number of signatures in k^{th} training set.

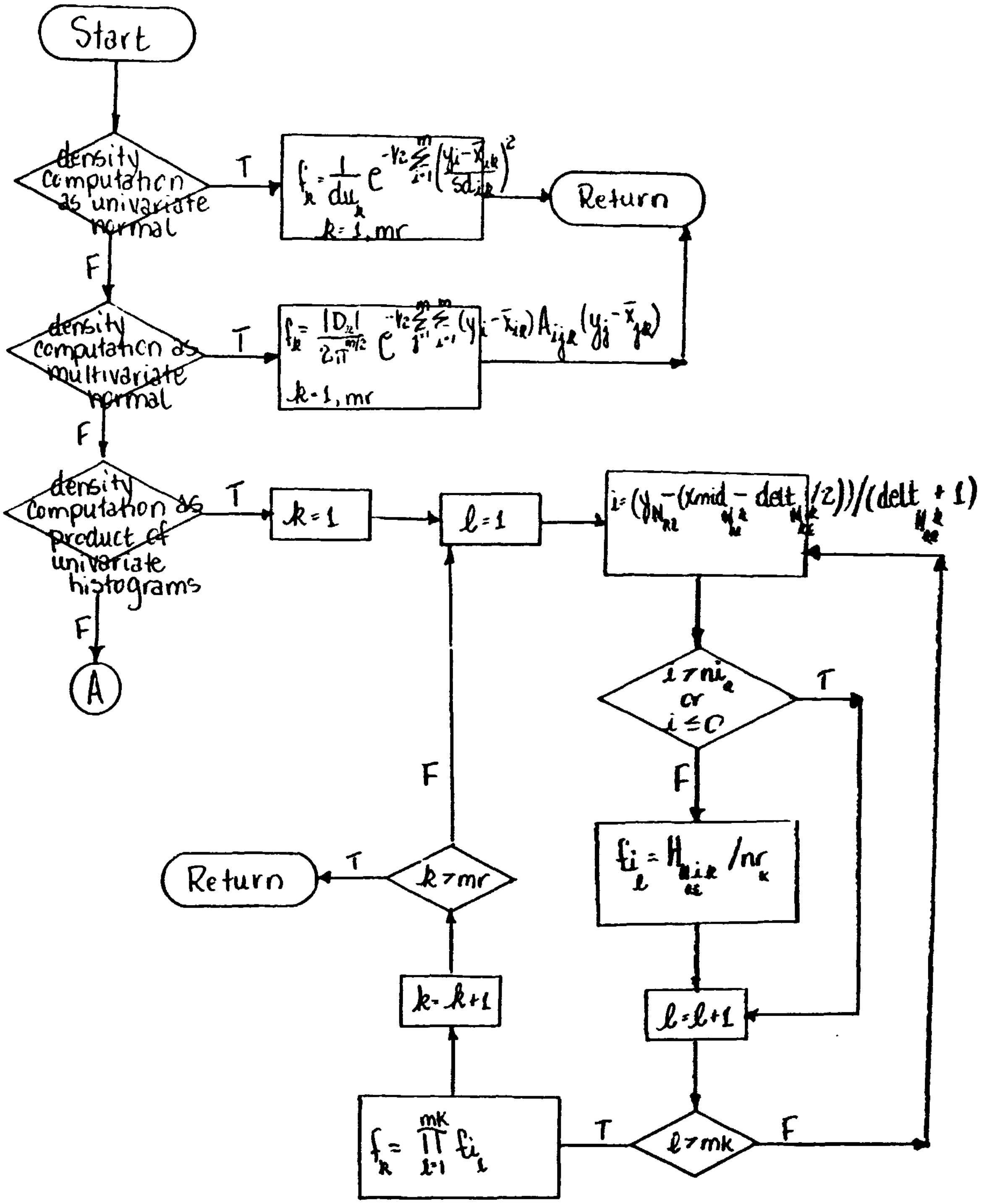
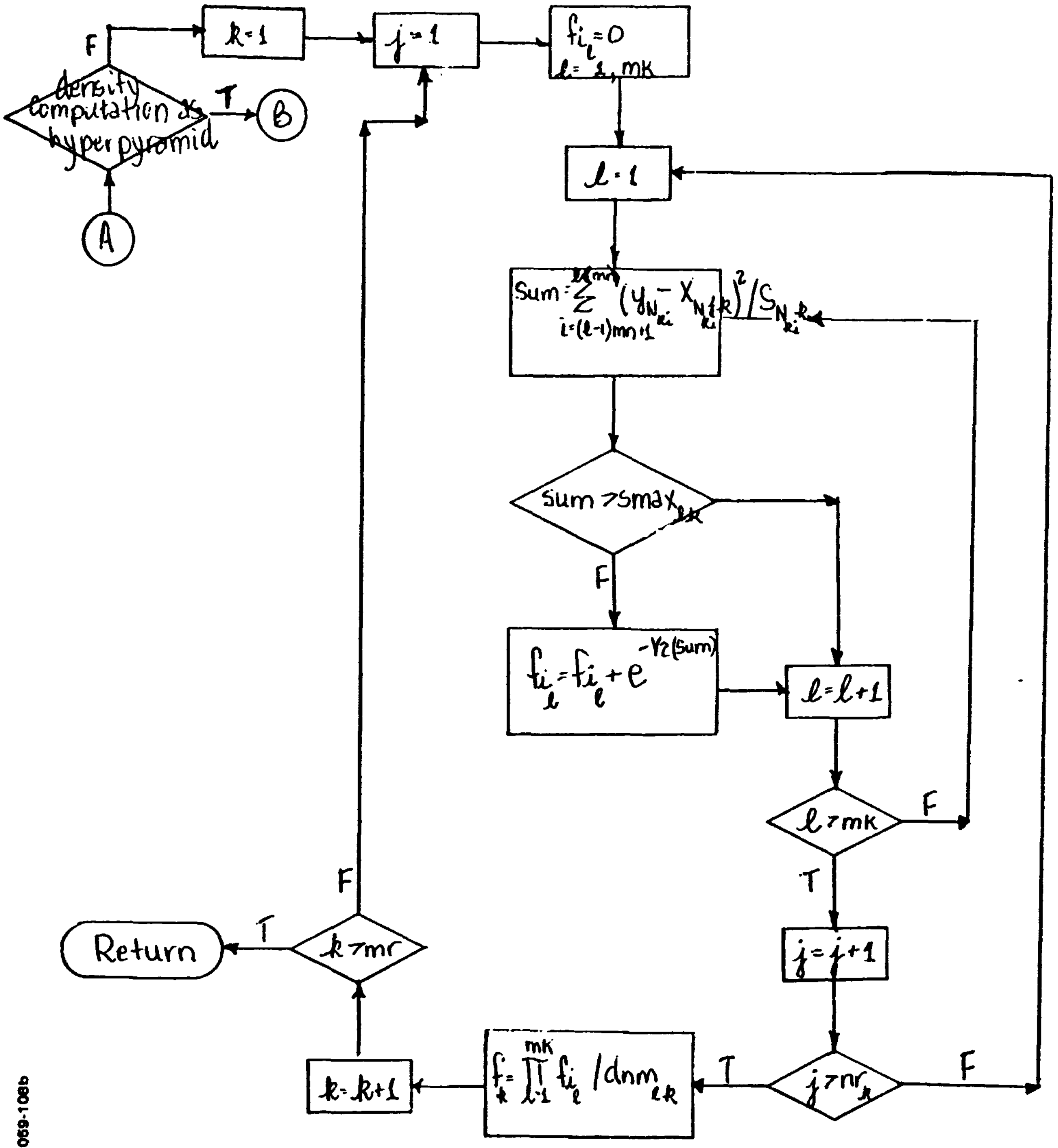

Start
density computation as univariate normal
T
$f_k = \frac{1}{du_k} e^{-1/2 \sum_{i=1}^{m} \left(\frac{y_i - \bar{x}_{ik}}{sd_{ik}}\right)^2}$
$k = 1, mr$
Return
F
density computation as multivariate normal
T
$f_k = \frac{|D_k|}{2\pi^{m/2}} e^{-1/2 \sum_{j=1}^{m} \sum_{i=1}^{m} (y_i - \bar{x}_{ik}) A_{ijk} (y_j - \bar{x}_{jk})}$
$k = 1, mr$
F
density computation as product of univariate histograms
T
$k = 1$
$\ell = 1$
$i = (y_{N_{\ell k}} - (xmid_{N_{\ell k}} - delt_{N_{\ell k}}/2))/(delt_{N_{\ell k}} + 1)$
F
A
$i > ni_\ell$ or $i \le 0$
T
F
$fi_\ell = H_{N_{\ell k} i k} / nr_k$
$\ell = \ell + 1$
$\ell > mk$
T
F
$f_k = \prod_{\ell=1}^{mk} fi_\ell$
$k = k + 1$
$k > mr$
T
F
Return

Figure 1-9 Subroutine COMPUT

8059-108a


F
k=1
j=1
$f_{i_\ell}=0$
$\ell=1, mK$
density computation is hyperpyramid
T
B
A
$\ell=1$
$Sum=\sum_{i=(\ell-1)mn+1}^{\ell(mn)}(y_{N_{ki}}-X_{N_{ki}\ell k})^2/S_{N_{ki}k}$
$sum > smax_{\ell k}$
F
$f_{i_\ell}=f_{i_\ell}+e^{-1/2(sum)}$
$\ell=\ell+1$
$\ell > mK$
F
T
$j=j+1$
$j > nr_k$
T
F
$f_k=\prod_{\ell=1}^{mK} f_{i_\ell}/dnm_{\ell k}$
$k=k+1$
$k > mr$
F
T
Return

8059-108b

Figure 1-9 (Cont.)

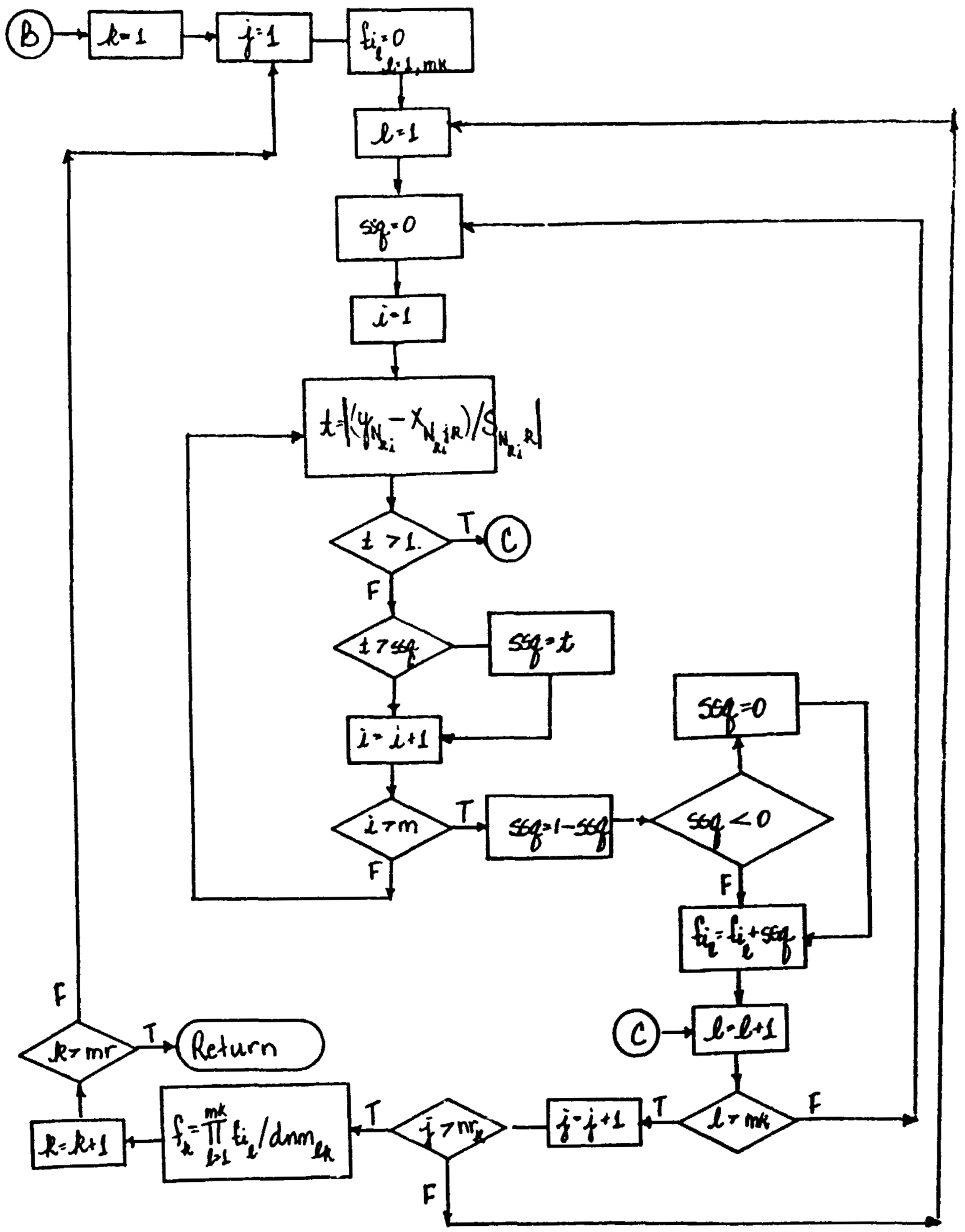


Figure 1-9 (Cont.)

5059-108c

1.3.2 Information Divergence Program

The Information Divergence Program computes the measure of the amount of information which the differences between two distributions contribute to the classification of a spectral signature. The training set data may be in one of the forms discussed for the Bayesian Decision Function Program, while the data referred to as sample data are the representation of the training set data in digitized scanner or continuous data form.

The computation is again based upon the probability density functions computed from the training sets evaluated at each sample signature, and the options for this computation are the same as those in the Bayesian Decision Function Program.

1.3.2.1 Information Divergence Mathematical Description

Divergence is defined as the sum of two information measures.

Suppose a signature $\vec{x}$ arises from one of two object classes, j or k. The likelihood ratio method of classification is to evaluate the likelihood function

$$L(\vec{x}) = \frac{f_j(\vec{x})}{f_k(\vec{x})} \qquad (1\text{-}14)$$

and assign the signature to object class j, if L exceeds some criterion value, and to class k otherwise. The amount of information in the likelihood ratio for making the decision is log L. (Log_2 L gives the information in bits.)

To obtain the average information per observation for evaluating the hypothesis that the signature belongs to object class j against the alternative class k, it is necessary to take the expected value over distribution j:

$$I(j;k) = \int \left(\log \frac{f_j(\tilde{x})}{f_k(\tilde{x})} \right) f_j(\tilde{x})\, dx \, . \qquad (1\text{-}15)$$

Similarly, the average information for evaluating the hypothesis that the signature belongs to class k against the alternative class j is

$$I(k;j) = \int \left(\log \frac{f_k(\tilde{x})}{f_j(\tilde{x})} \right) f_k(\tilde{x})\, dx \, . \qquad (1\text{-}16)$$

The divergence between the two classes is

$$J(j,k) = I(j;k) + I(k;j)\ . \tag{1-17}$$

Since the distributions are unknown, they are assumed to be characterized by the training data sets and the alogrithms for evaluating the density functions, $f_k(\vec{x})$ and $f_j(\vec{x})$, as described in Section 1.3.

The required computations then become

$$\begin{aligned} I(j;k) &= \frac{1}{N_j} \sum_{i=1}^{N_j} \log \frac{f_j(\tilde{x}_i)}{f_k(\tilde{x}_i)}\ , \\ I(k;j) &= \frac{1}{N_k} \sum_{i=1}^{N_k} \log \frac{f_k(\tilde{x}_i)}{f_j(\tilde{x}_i)}\ , \end{aligned} \tag{1-18}$$

and

$$J(j,k) = I(j;k) + I(k;j).$$

1.3.2.2 Flow Chart of Information Divergence Program

All flow charts of subroutines referenced in Figure 1-10 may be found in Section 1.3.1.2.

KEY TO NOTATION

Main Program, Information Divergence

m	= number of channels
mr	= number of training sets to be diverged (= number of sample sets)
mk	= number of subsets of channels
mn	= number of channels in each subset of channels

N_{ij}, i=1, mr, j=1, mk*mn = channel numbers in the subsets of channels for i^{th} training set

ns_k, k=1, mr = number of signatures in k^{th} sample set

S_{ij}, i=1, m, j=1, mr = smoothing parameters for the i^{th} channel of the j^{th} training set (for Specht function or hyperpyramid function)

$dmax_i$, i=1, mr = upper bound on truncation error for computing the probability density function of a signature attributed to the i^{th} training set (for Specht function)

du_i, i=1, mr = denominator of the univariate normal probability density function for the i^{th} training set

sd_{ij}, i=1, mj=1, mr = standard deviation in the i^{th} channel of the j^{th} training set

dnm_{ij}, i=1, mk, j=1, mr = denominator of the probability density function for the i^{th} subset of channels of the j^{th} training set

$smax_{ij}$, i=1, m, j=1, mr = upper limit on sum of squares for truncation of computation of density function (for Specht function)

Y_i, i=1, m = signature from sample set

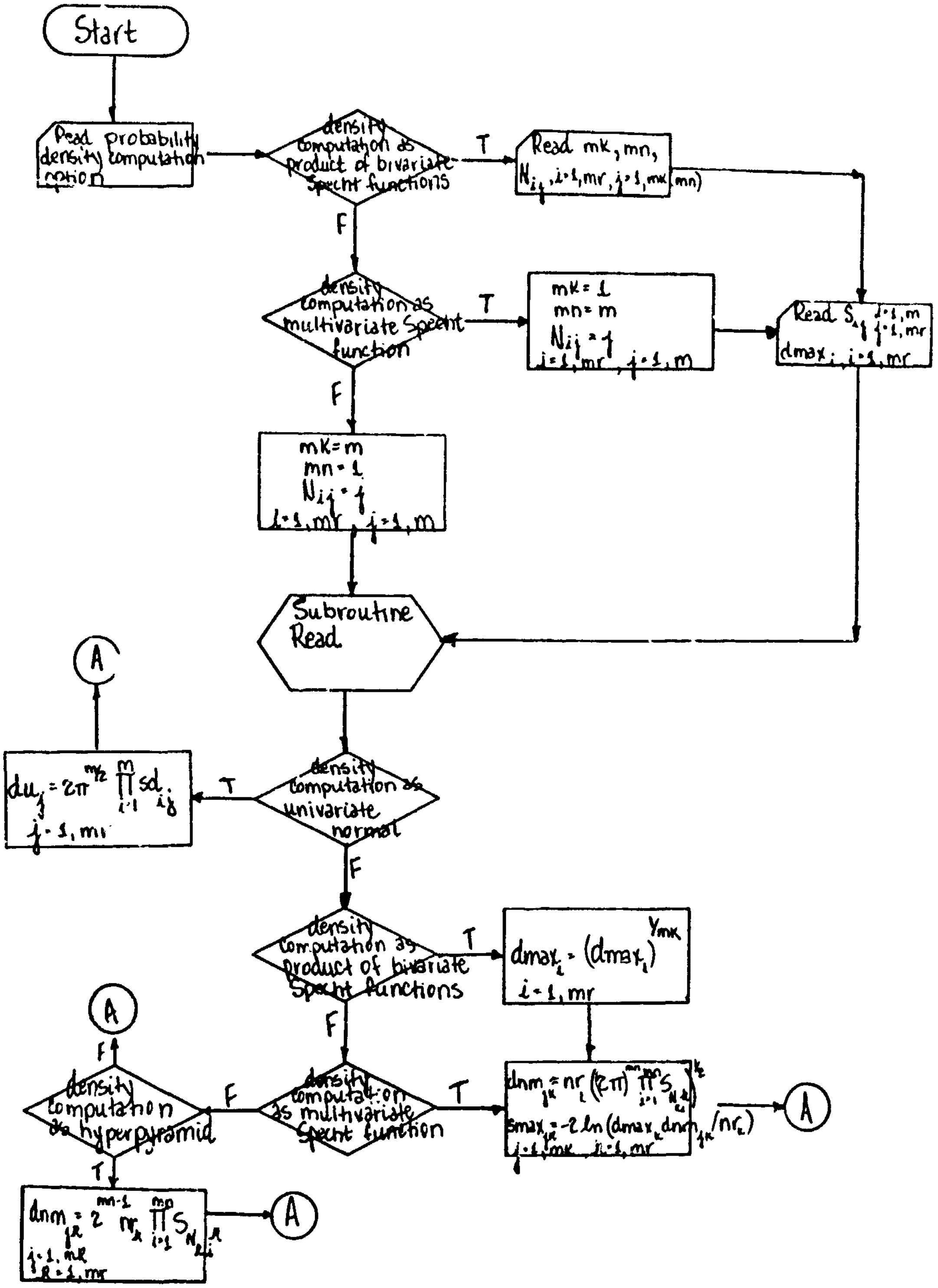
f_i, i=1, mr = probability density function of a sample signature when attributed to the i^{th} training set

djk_{ij}, i=1, mr, j=1, mr = total information in training set i for discrimination in favor of set i against the alternative set j

$f\ell_i$, i=1, mr = logarithm of the probability density function of a sample signature when attributed to the i^{th} training set

dv_{ij}, i=1, mr, j=1, mr = divergence between training sets i and j

nr_k, k=1, mr = number of signatures in k^{th} training set.


Start
Read probability density computation option
density computation as product of bivariate Specht functions
T
F
Read mk, mn, N_{ij}, i=1,mr, j=1,mk(mn)
density computation as multivariate Specht function
T
F
mk = 1
mn = m
N_{ij} = j
i=1,mr, j=1,m
Read S_{ij} i=1,m j=1,mr
$dmax_i$, i=1,mr
mk=m
mn=1
N_{ij} = j
i=1,mr, j=1,m
Subroutine Read
A
$du_j = 2\pi^{m/2} \prod_{i=1}^{m} sd_{ij}$
j=1,mr
density computation as univariate normal
T
F
density computation as product of bivariate Specht functions
T
F
$dmax_i = (dmax_i)^{1/mk}$
i=1,mr
A
F
density computation as hyperpyramid
F
T
density computation as multivariate Specht function
T
$dnm_{jk} = nr_k \left((2\pi)^{mn} \prod_{i=1}^{mn} S_{N_{ki}}\right)^{1/2}$
$smax_{jk} = -2\ln(dmax_k \, dnm_{jk} / nr_k)$
j=1,mk, k=1,mr
A
$dnm_{jk} = 2^{mn-1} nr_k \prod_{i=1}^{mn} S_{N_{ki}}$
j=1,mk
k=1,mr
A

8059-109a

Figure 1-10 Main Program, Information Divergence

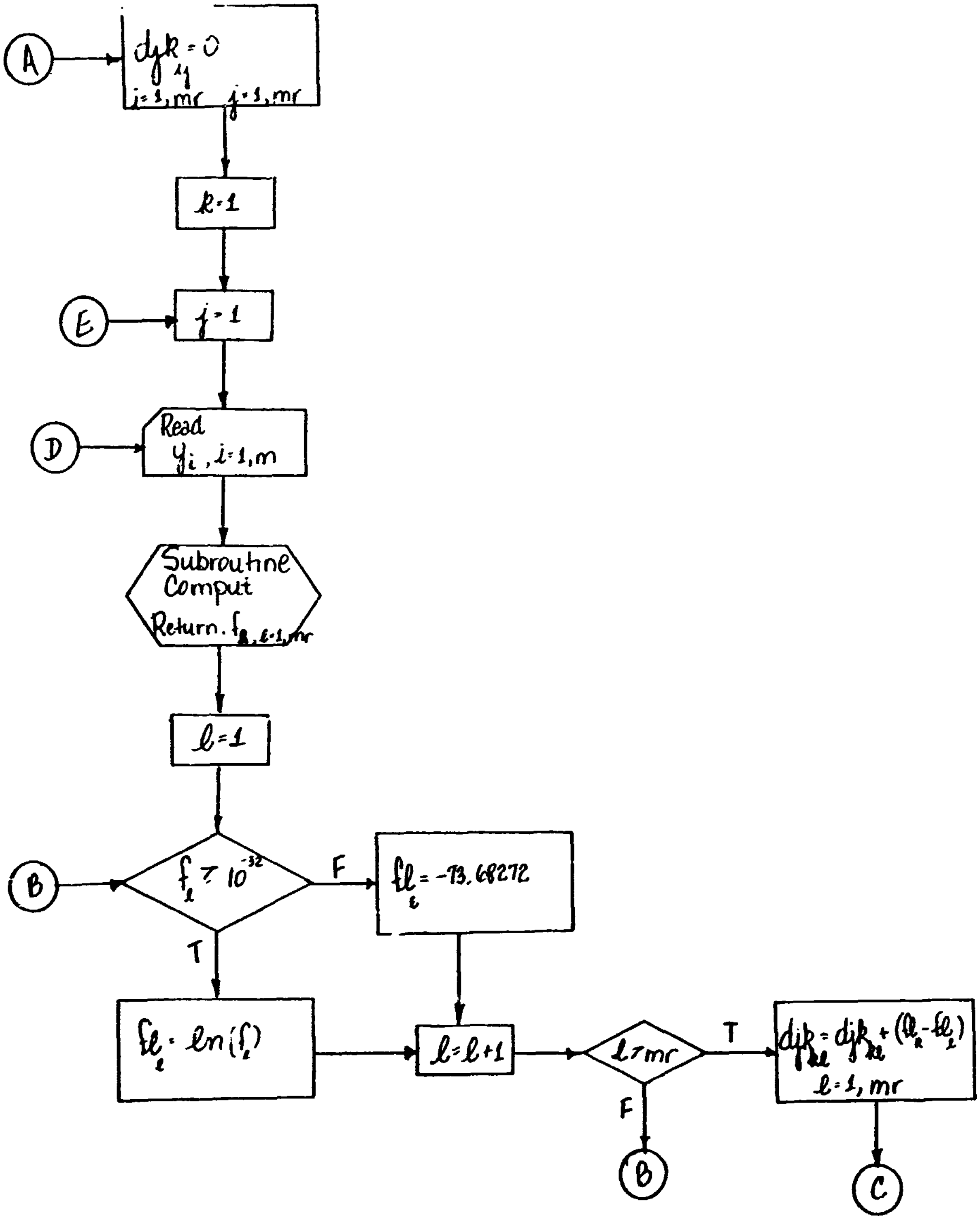


Figure 1-10 (Cont.)

8069-109b

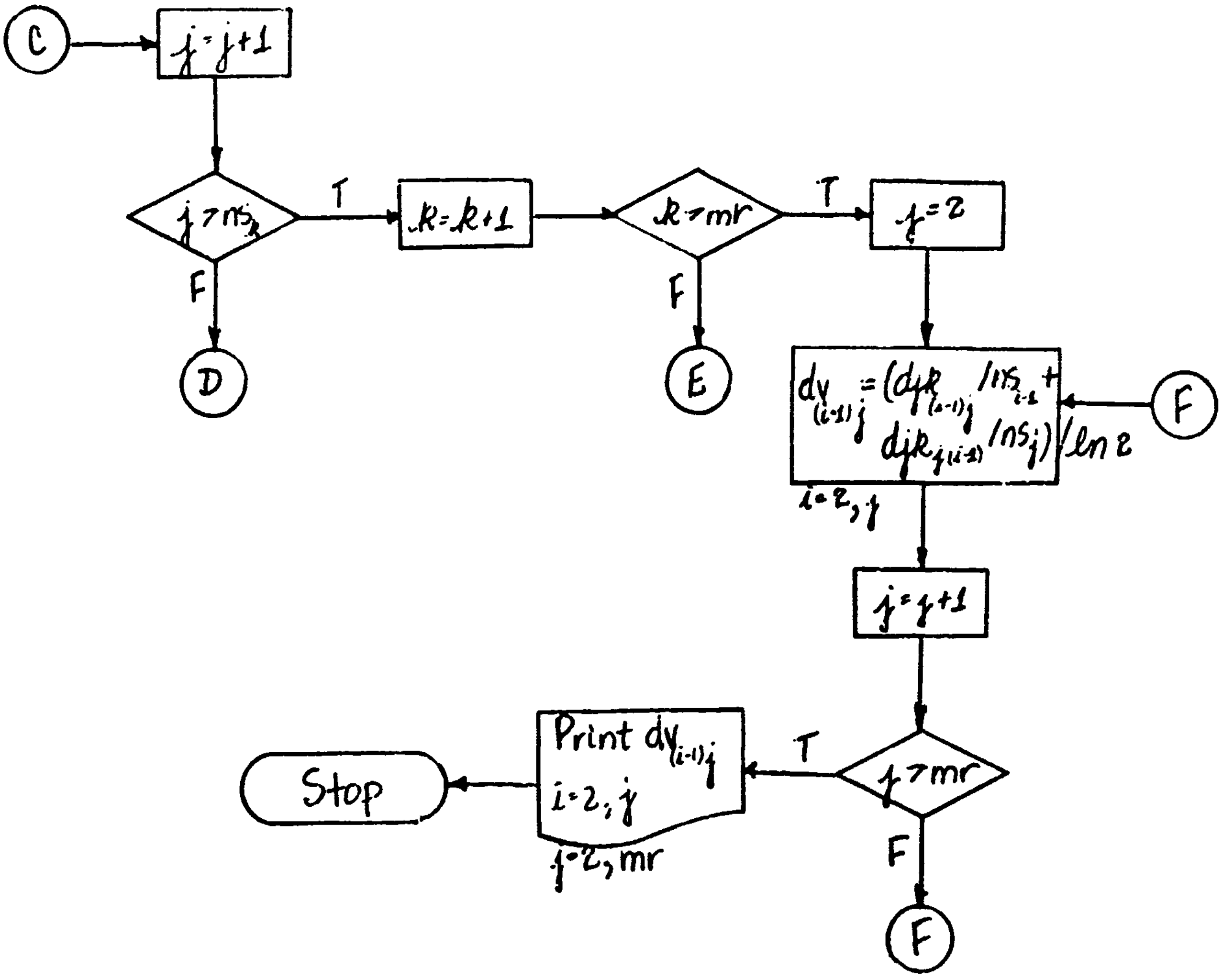


Figure 1-10 (Cont.)

1.3.3 Density Divergence Program

The modified divergence computation, referred to as density divergence, is used to determine the divergence between two representations of the same population. The Density Divergence Program specifically computes the divergence between a population known to be multivariate normal and representation of that population with a training set and the hyperpyramid smoothing function. It is useful in determining the utility of a given set of smoothing parameters. A small density divergence indicates a valid smoothing function representation of the true distribution of the population.

The divergence computation (non-modified) is:

$$J = \int f_1(\vec{x}) \log \frac{f_1(\vec{x})}{f_2(\vec{x})} \, d\vec{x} + \int f_2(\vec{x}) \log \frac{f_2(\vec{x})}{f_1(\vec{x})} \, d\vec{x} \, . \tag{1-19}$$

Define:

$$f_1 \, d\vec{x} = dP \text{ and} \int f_1 \, d\vec{x} \longrightarrow \frac{1}{N} \sum_{i=1}^{N} \tag{1-20}$$

Then

$$d\vec{x} \equiv \frac{dP}{f_1} \quad \text{and substituting in Equation 1-19}$$

$$\begin{aligned} J &= \frac{1}{N} \sum_{i=1}^{N} \left(\log \frac{f_1(\vec{x}_i)}{f_2(\vec{x}_i)} - \frac{f_2(\vec{x}_i)}{f_1(\vec{x}_i)} \right) \\ &= \frac{1}{N} \sum_{i=1}^{N} \left(1 - \frac{f_2(\vec{x}_i)}{f_1(\vec{x}_i)} \right) \log \frac{f_1(\vec{x}_i)}{f_2(\vec{x}_i)} \, . \end{aligned} \tag{1-21}$$

When f_1 is the expression for the multivariate normal probability density* and f_2 is the hyperpyramid function* using a sample from the distribution, the choice of smoothing parameters used in f_2 may be evaluated.

*Refer to Section 1.3.

Input to the program consists of a sample from the population, hyperpyramid smoothing parameters for the sample, and the population inverse covariance matrix and channel means.

1.3.3.1 Flow Chart of Density Divergence Program

Refer to Figure 1-11.

KEY TO NOTATION

Main Program, Density Divergence and Subroutine COMPUT2.

m = number of channels

N = number of signatures in sample

X_{ij}, $i=1,m, j=1,N$ = sample from population

S_i, $i=1,m$ = smoothing parameter for ith channel, hyperpyramid function

${}_{m}A_{m}$ = inverse covariance matrix of population

det = determinant of inverse covariance matrix

$\bar{x}_i$, $i=1,m$ = ith channel mean for population

dnm = denominator of hyperpyramid smoothing function

f_1 = multivariate normal probability density evaluated at one signature

f_2 = hyperpyramid smoothing function evaluated at one signature

y_i, $i=1,m$ = signature at which probability density function is to be evaluated

d = density divergence in bits.

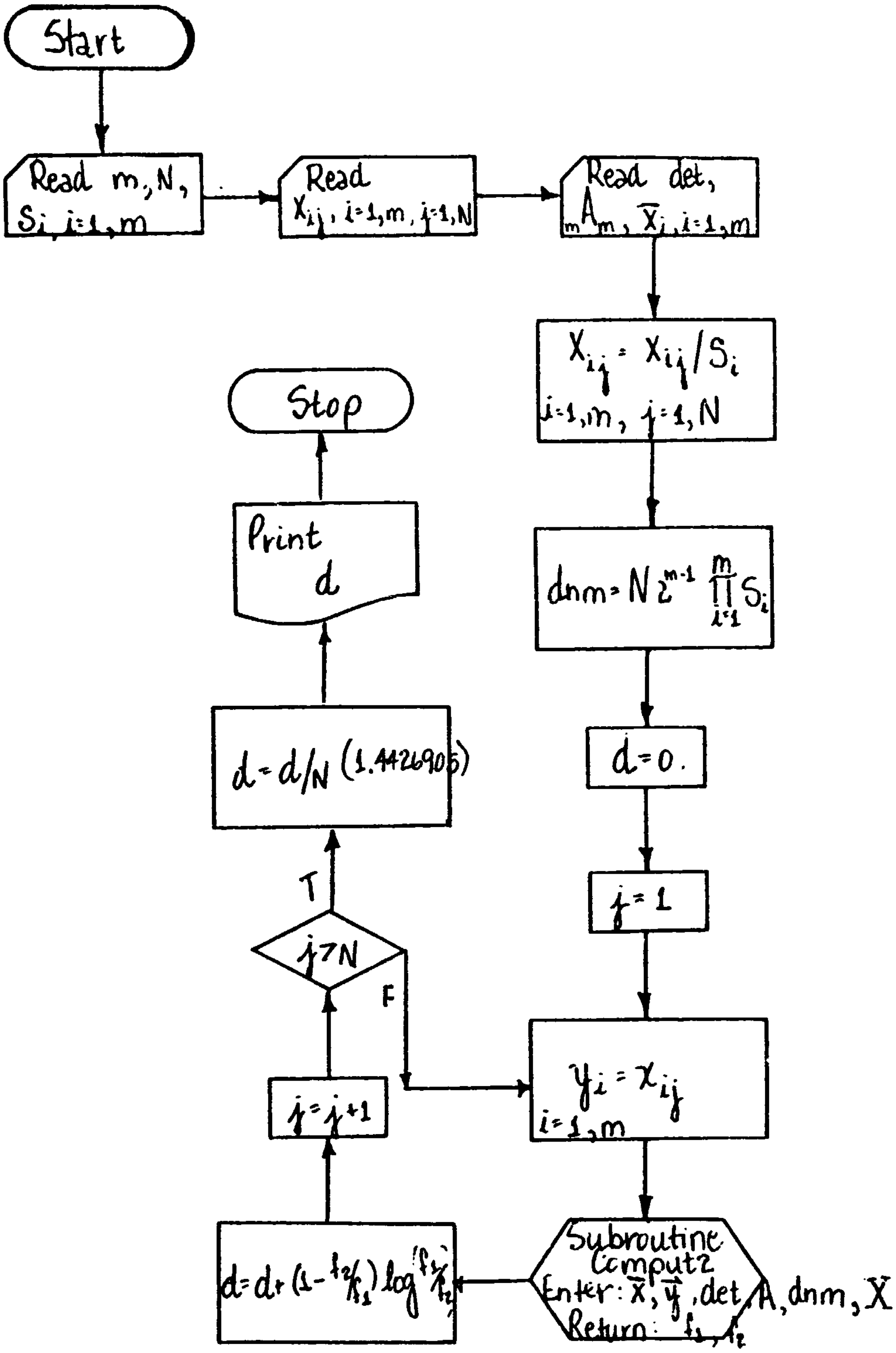

Start
Read m, N, S_i, i=1,m
Read X_{ij}, i=1,m, j=1,N
Read det, ${}_mA_m$, $\bar{X}_i$, i=1,m
$X_{ij} = X_{ij}/S_i$
i=1,m, j=1,N
$dnm = N\,2^{m-1}\prod_{i=1}^{m} S_i$
d=0.
j=1
$y_i = x_{ij}$
i=1,m
Subroutine
Comput2
Enter: $\vec{x}$, $\vec{y}$, det, A, dnm, X
Return: f_1, f_2
$d = d + (1 - {}^{f_2}\!/_{f_1}) \log {}^{f_2}\!/_{f_1}$
j=j+1
j>N
F
T
d=d/N (1.4426905)
Print
d
Stop

Figure 1-11 Main Program, Density Divergence and Subroutine COMPUT2

8059-110a

Subroutine Compute

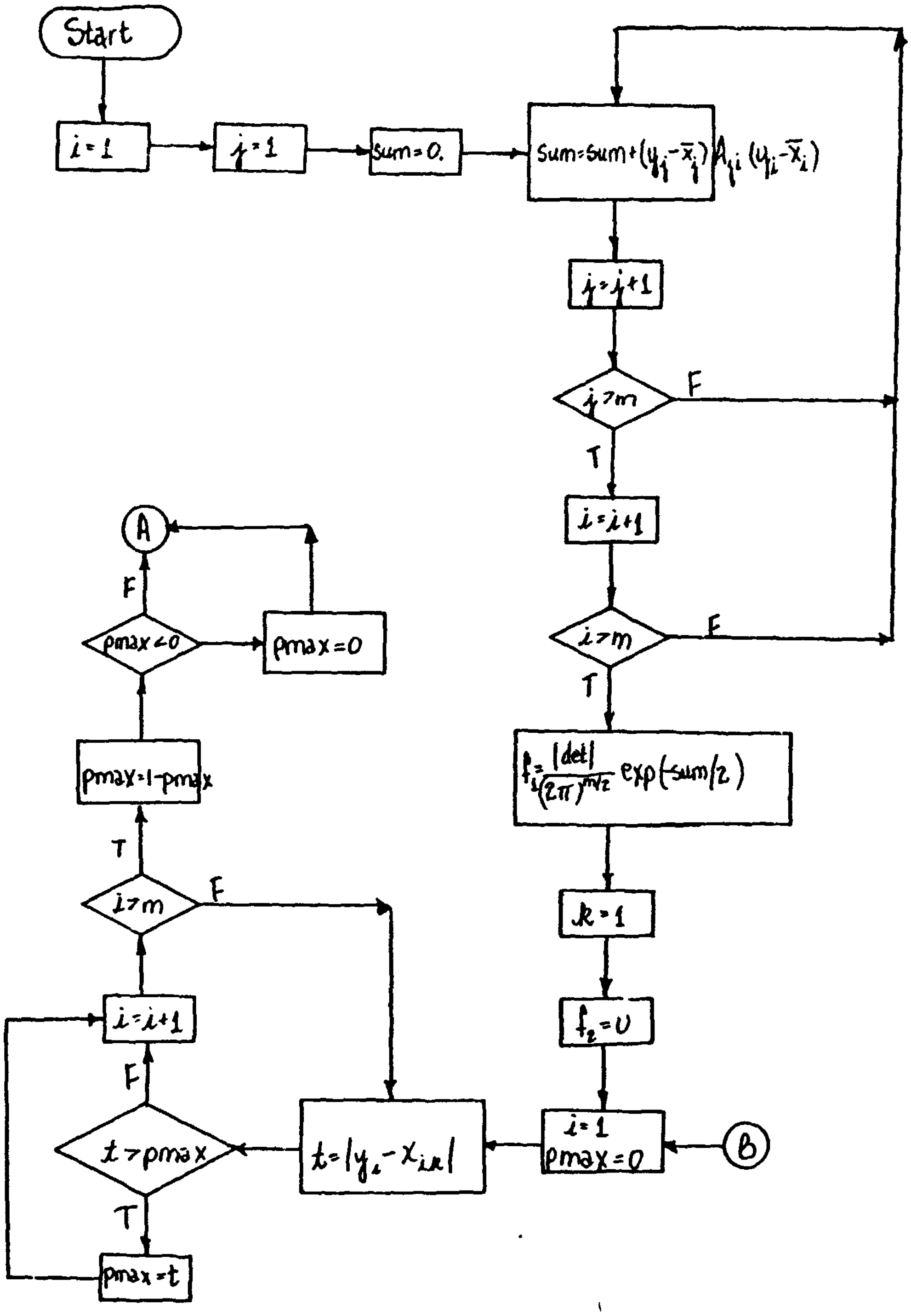


Figure 1-11 (Cont.)

8059-110b

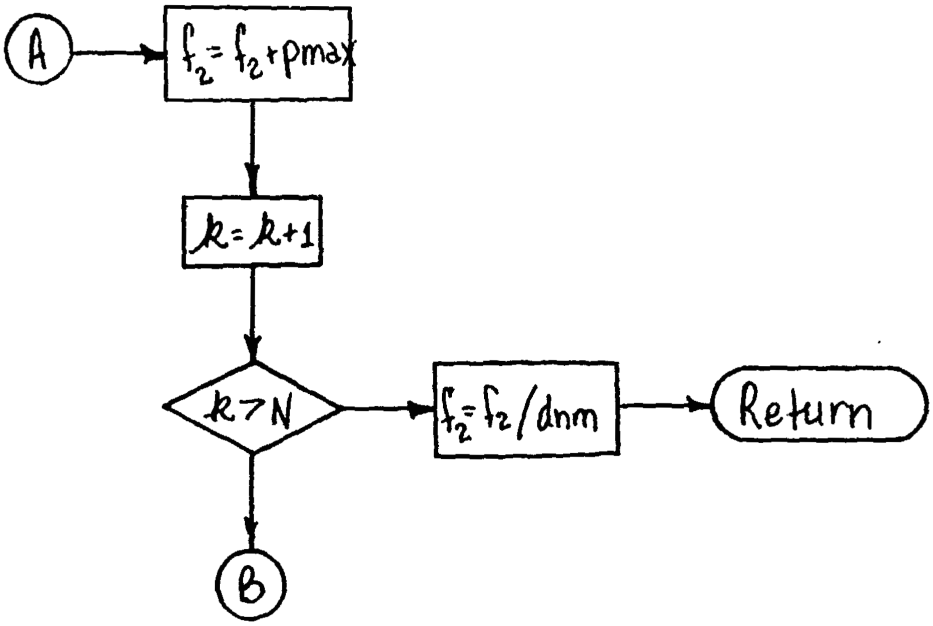


Figure 1-11 (Cont.)

1.3.4 Density Lister Program

Due to the high cost involved in computation of the probability densities necessary for signature analysis, software is needed for calculation and recording of this quantity to be read rather than recomputed by other programs. This program evaluates the hyperpyramid probability density function* at any number of signatures when attributed to one training set, recording the value on an external storage device for later use.

Input to the program is the training set signatures, signatures at which the probability density is to be evaluated, and hyperpyramid smoothing parameters. The output consists of the signatures at which evaluation was done with the probability density appended as the m + 1st component, where m is the number of channels.

1.3.4.1 Flow Chart of Density Lister Program

Refer to Figure 1-12.

KEY TO NOTATION

Main Program, Density Lister

m	=	number of channels
N	=	number of signatures in training set
S_i, i=1,m	=	hyperpyramid smoothing parameters for training set
X_{ij}, i=1,m,j=1,N	=	training set signatures
ns	=	number of groups of signatures at which probability densities will be evaluated
num_i, i=1,ns	=	number of signatures for evaluation in each group
dnm	=	hyperpyramid smoothing function denominator
y_i, i=1,m	=	signature at which probability density is to be evaluated
f	=	hyperpyramid probability density.

*Refer to Section 1.3.

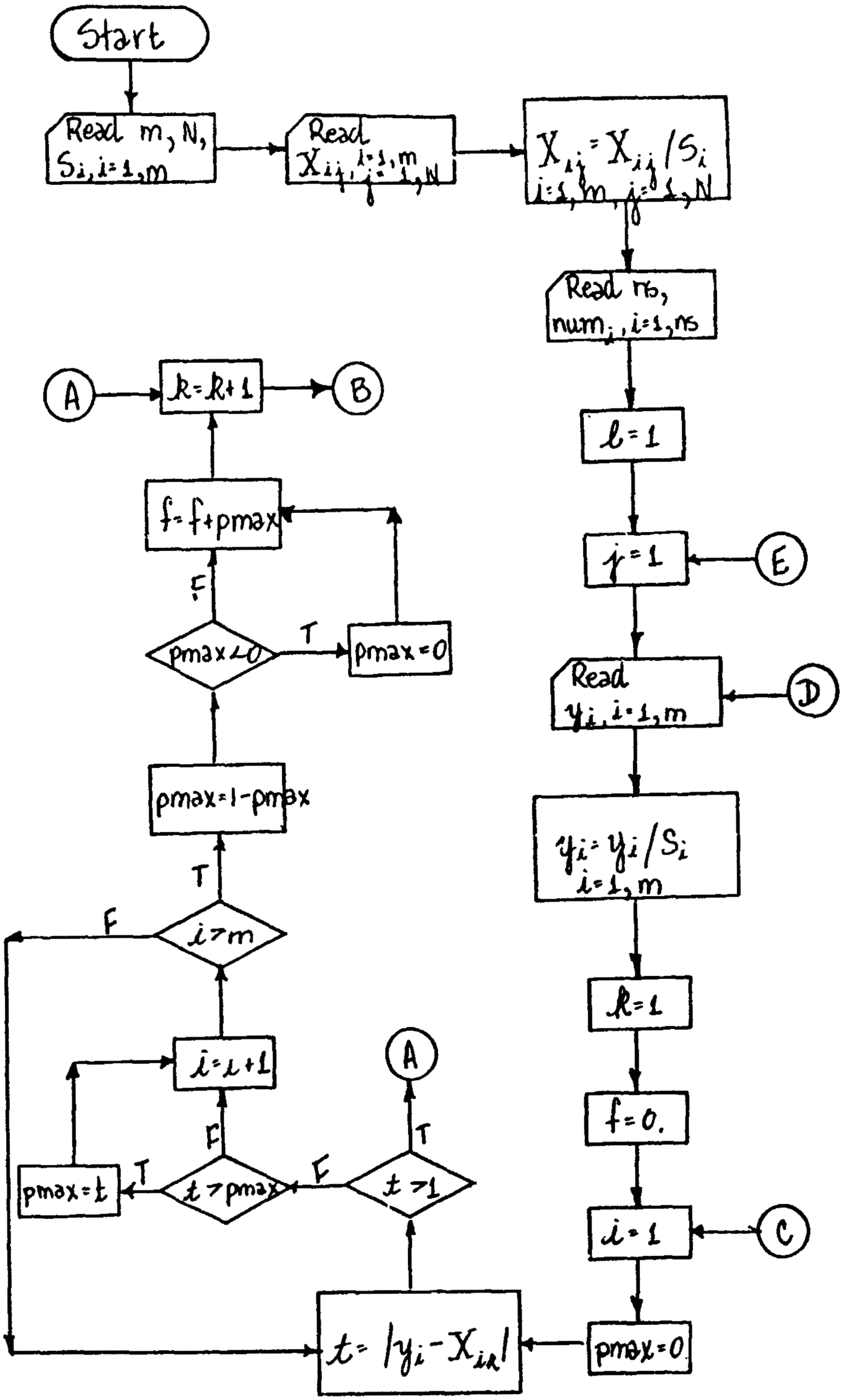


Figure 1-12 Main Program, Density Lister

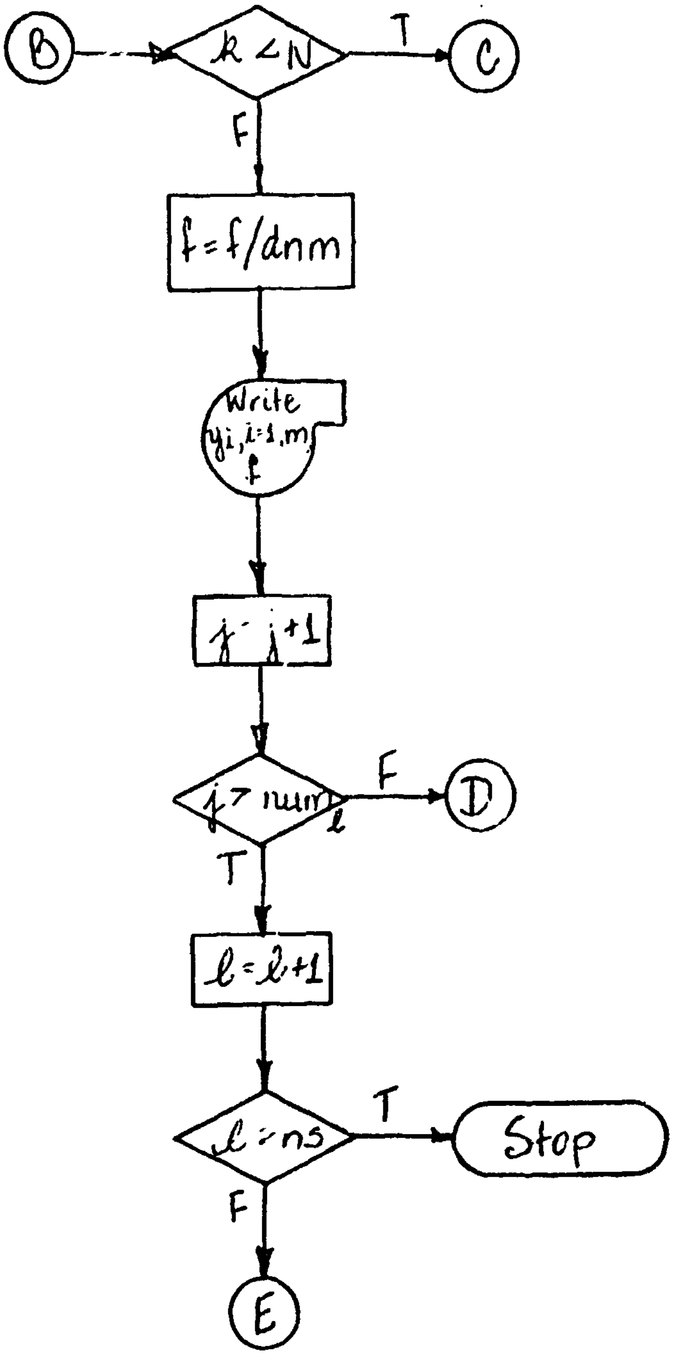


Figure 1-12 (Cont.)

1.3.5 Decision Ruler Program

The Decision Ruler Program has only to read any number of files of Density Lister output in order to assign a signature to a category. As noted in Section 1.3.4, the Density Lister Program records in an external file the hyperpyramid probability density of each signature as attributed to one target. Upon creation of several such files containing the densities computed at identical signatures attributed to different targets, the Decision Ruler Program is used to make target assignment of the sample signature to the category for which that signature's conditional probability density is the largest. The probabilities of correct classification are computed for each sample signature as well.

1.3.5.1 Flow Chart of the Decision Ruler Program

Refer to Figure 1-13.

KEY TO NOTATION

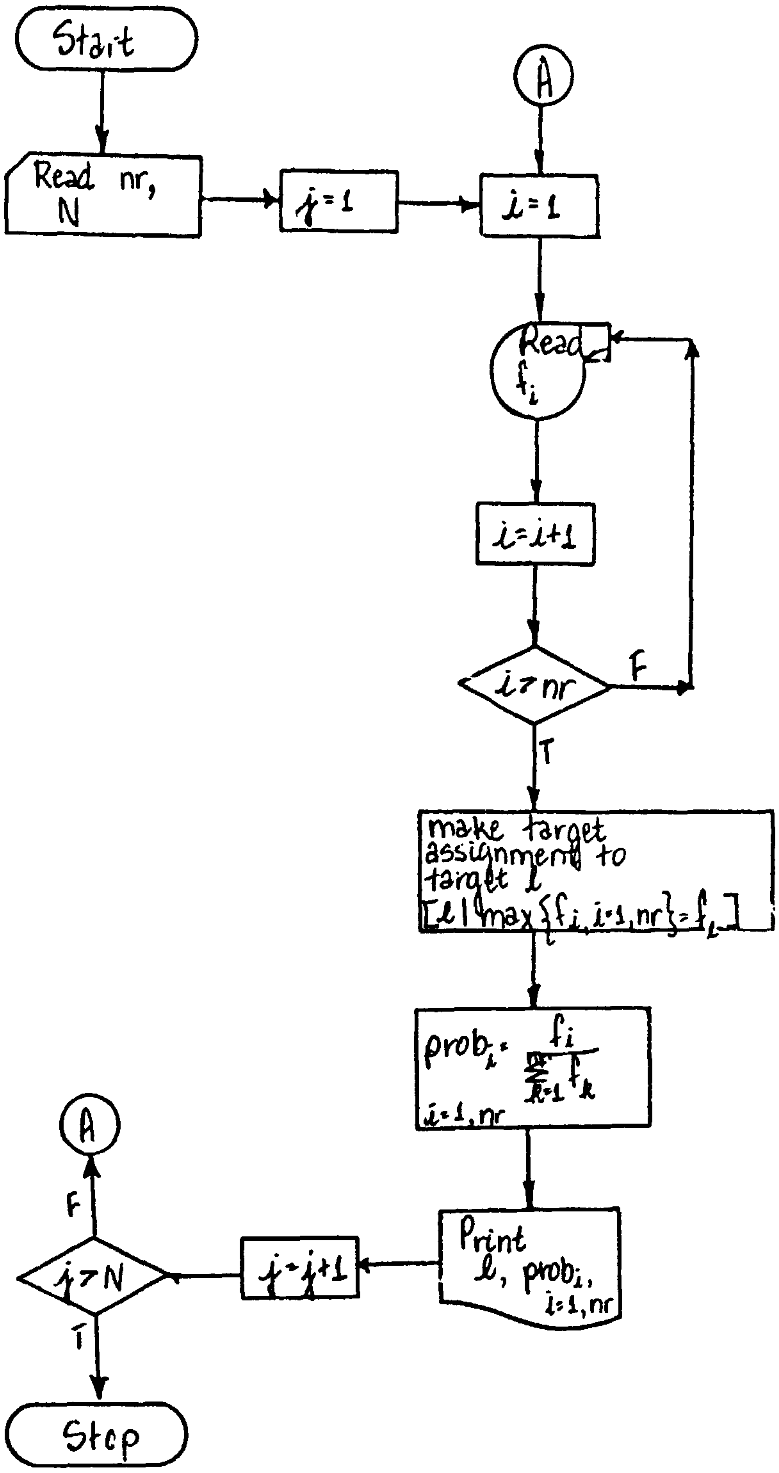
Main Program, Decision Ruler Program

nr = number of targets (number of density Lister files)

N = number of signatures to be classified

f_i, i=1, nr = probability density of sample signature attributed to the i^{th} target

$prob_i$, i=1,nr = probability of the correct classification of sample signature to the i^{th} target.


Start
Read nr, N
j=1
A
i=1
Read f_i
i=i+1
i>nr
F
T
make target assignment to target l
[l | max{f_i, i=1,nr} = f_l]
prob_i = f_i / Σ_{k=1}^{nr} f_k
i=1,nr
Print l, prob_i, i=1,nr
j=j+1
j>N
F
T
A
Stop

8059-112

Figure 1-13 Main Program, Decision Ruler Program

1.4 CLUSTER ANALYSIS

1.4.1 Cluster Analysis Mathematical Description

Frequently, in the course of the analysis of multivariate data, it is useful to represent a population probability density distribution in some parsimonious but reasonably accurate way. Usually, some simple parametric form is assumed and the parameters are estimated from a sample

Even when it would be otherwise difficult to justify, the multivariate normal distribution is used because it is easy to determine the parameters and also easy to evaluate the density expression.

When a sample distribution exhibits excessive skew, kurtosis, multimodality, or other departure from normality, alternative representations are sought. This section describes a technique for dealing with the multimodal or clustered distributions when it may be hypothesized that each cluster is multivariate normal. The procedure requires an estimate of the probability density associated with each observation—this may be accomplished by a variety of methods, among which a particular one described by Specht[*] will serve as an example.

If the observations are resequenced to be in the order of descending probability density, the first observation may clearly be taken to be the mode of the first cluster. Each succeeding observation is then assessed with regard to the probability that it belongs to one of the existing clusters or, alternatively, that a new cluster must be formed.

Of fundamental importance in the assessment is the relation

$$f(\vec{x}) = f(\vec{\underline{x}}) \exp - 1/2 \, (\vec{x} - \vec{\underline{x}})^{T} C^{-1} (x - \vec{\underline{x}}) \qquad (1.4\text{-}1)$$

which gives the probability density f for an observation vector $\vec{x}$ on the hypothesis that the distribution is multivariate normal with mean $\vec{\underline{x}}$ and covariance matrix C. The quadratic form in the right half of Equation 1.4-1 can be replaced with the generalized Mahalanobis squared distance.

[*] D. F. Specht, "Generation of Polynomial Discriminant Functions for Pattern Recognition," IEEE Transactions on Electronic Computers, Vol EC-16, No. 3, June 1967.

$$d^2 = (\vec{x}-\underline{\vec{x}})^T \, C^{-1} \, (\vec{x}-\underline{\vec{x}}) \qquad (1.4\text{-}2)$$

yielding

$$\frac{f\,(\vec{x})}{f\,(\underline{\vec{x}})} = \exp\left(-\frac{d^2}{2}\right). \qquad (1.4\text{-}3)$$

Taking the log of both sides,

$$d^2 = -\,2\ln\left(\frac{f\,(\vec{x})}{f\,(\underline{\vec{x}})}\right). \qquad (1.4\text{-}4)$$

Note that Equations 1.4-2 and 1.4-4 can be regarded as two different ways of computing the same quantity, one using the hypothesized mean and covariance matrix and the other using the previously estimated probability densities.

In the cluster analysis algorithm to be described, the extent of the disagreement between the two computations makes an important contribution to the decision, because appreciable differences will occur only when the wrong cluster (and hence the wrong mean and covariance matrix) has been hypothesized. To allow for the difference, Equation 1.4-4 will be rewritten using the symbol e to represent the value of d that would be expected from the densities $f(\vec{x})$ and $f(\underline{\vec{x}})$.

$$e^2 = -2\ln \frac{f(\vec{x})}{f(\underline{\vec{x}})} \qquad (1.4\text{-}5)$$

At this point, the notation will be permitted to reflect the presence of m clusters, each of which is to be tested with regard to the hypothesis that the current observation x arose from that particular cluster. Evaluating both Equations 1.4-2 and 1.4-5 for the ith cluster,

$$d_i^2 = (\vec{x}-\underline{\vec{x}}_i)^T \, C_i^{-1} \, (\vec{x}-\underline{\vec{x}}_i) \qquad (1.4\text{-}6)$$

$$e_i^2 = -2\ln\left(\frac{f\,(\vec{x})}{f\,(\underline{\vec{x}}_i)}\right). \qquad (1.4\text{-}7)$$

Since d_i is the observed generalized distance (between the observation and the center of the ith cluster) and e_i is the expected generalized distance, it is reasonable to define the variable α_i in Equation 1.4-8 as the expected fraction of the observation relative to the ith cluster,

$$\alpha_i = \frac{e_i}{d_i} \qquad (1.4\text{-}8)$$

and the unexpected fraction β_i then given by Equation 1.4-9:

$$\begin{aligned}\beta_i &= 1 - \alpha_i \\ &= 1 - \frac{e_i}{d_i}\end{aligned} \qquad (1.4\text{-}9)$$

The unexpected part of the observation relative to the ith cluster is the vector:

$$\vec{Z}_i = \beta_i \, (\vec{x}-\underline{\vec{x}}_i) \quad . \qquad (1.4\text{-}10)$$

Clearly, since the covariance matrices C_i, the means $\underline{\vec{x}}_i$, and the probability densities f $(\vec{x})$ and f $(\underline{\vec{x}}_i)$ are estimated from the observations and hence are not free of errors, the unexpected fraction β_i is not identically zero even for the best choice for i.

Although a rule for making a choice has not yet been stated, it may be assumed that one exists. If the rule selects j for the correct value of i, the unexpected part of the observation for the chosen cluster is:

$$\vec{Z}_j = \beta_j \, (\vec{x}-\underline{\vec{x}}_j). \qquad (1.4\text{-}11)$$

The covariance matrix of the Z_j for all previous selections can be called the unexpected covariance matrix. This matrix, denoted by W, plays a central role in the selection rule to be proposed.

The generalized Mahalanobis squared distance for the unexpected part of an observation $\vec{x}$ (relative to the ith cluster) is:

$$g_i^2 = Z_i^T \, W^{-1} \, Z_i \; . \qquad (1.4\text{-}12)$$

The desired selection procedure can now be stated:

1. Evaluate g_i^2 for each cluster.

2. Let j be the value of i which minimizes g_i^2 for i = 1, m.

3. If g_j^2 is gre tor than a prechosen number Q, form a new cluster number m+1 centered at the given observation $\vec{x}$.

4. If g_j^2 is not greater than Q, the observation is classified as part of the jth cluster, with appropriate modifications made to the mean $\vec{\underset{\sim}{x}}_j$ and covariance matrix C_j.

On the hypothesis that unexpected variance is multivariate normal, the g_i^2 will possess the chi-squared distribution and Q can be chosen to yield a specified probability of forming a new cluster by chance when a large deviation occurs.

The covariance matrixes for th individual clusters are all biased in the direction leading to underestimation of expected variance. This effect occurs because observations are included in each cluster in the order of decreasing probability density. Fortunately, a suitable correction for the variance is available in the expression:

$$K = 1 - \left(\frac{f(\vec{x})}{f(\vec{\underline{x}}_i)} \right)^{\frac{2}{n+2}}$$

where

K = the fractional amount by which variance is underestimated

n = the number of variables in each multivariate observation.

1.4.2 Flow Chart of the Cluster Analysis Program

The following flow chart of the Cluster Analysis Program, Figure 1-14, makes reference to Subroutine EIGEN whose chart is presented in Section 1.2.2.

KEY TO NOTATION

Main Program, Cluster Analysis

IIN - input unit for data set to be cluster analyzed
IOUT - output unit for cluster analyzed data
LBL - data set identification
KD - data type
NV - number of variables
NG - number of groups
SL - smoothing parameters
OPT - option for space in which results of cluster analysis are to be printed
  MEAS - measurement space
  FACT - factor space
  YM - constant vector in transformation $(y = \tilde{A}\, x + \bar{b})$
  SA - transformation matrix $(y = \tilde{A}\, x + \bar{b})$
X - data point
NC - number of cases
N_k - number of vectors belonging to kth cluster
VF_k - factor by which variance in cluster k has been misestimated
PS - probability density
NOG - number of clusters
XM_{ij} - mean of cluster j
NCR - number of data points which have been read into program
PM_j - probability density at mode of cluster j
COV_{ij} - jth component of covariance matrix from cluster i
DF - number of data points used in computing unexplained covariance matrix
ESDS - expected square of Mahalanobis distance between vector arising from cluster under hypothesis and cluster mean
CHISQ - unexpected portion of square of Mahalanobis distance from mode of cluster under hypothesis
GMDS - Mahalanobis distance between vector and mode of cluster under hypothesis
CSMIN - minimum value of CHISQ under all hypotheses
CSMAX - maximum value which CSMIN can assume without causing a new cluster to be formed
V_{ijk} - diagonalized scatter matrix for cluster $_{k-1}$
U_{ijk} - unitary matrix used to diagonalize scatter matrix for cluster $_{k-1}$
V_{ijl} - diagonalized scatter matrix for unexpected variance
U_{ijl} - unitary matrix used to diagonalize scatter matrix of unexpected variance.

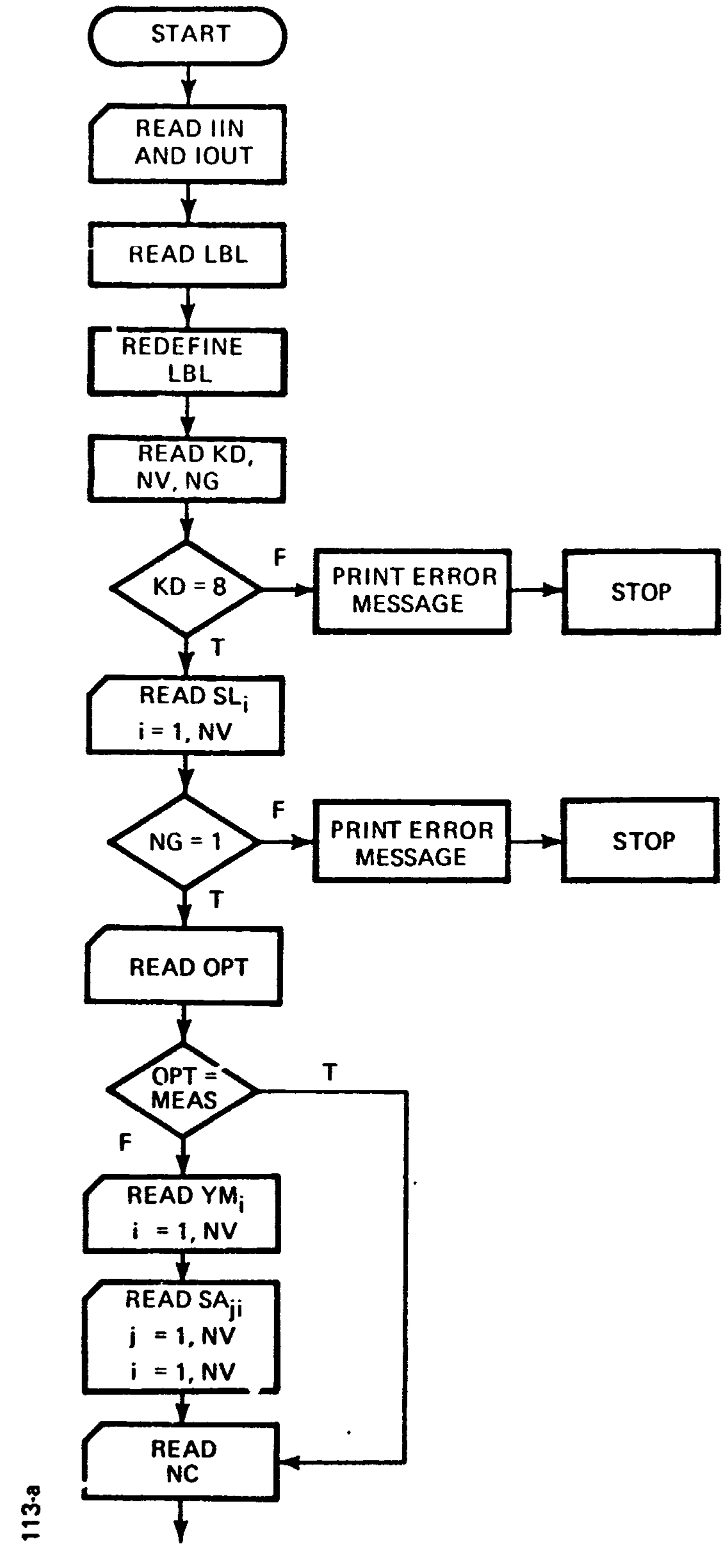


Figure 1-14 Main Program, Cluster Analysis

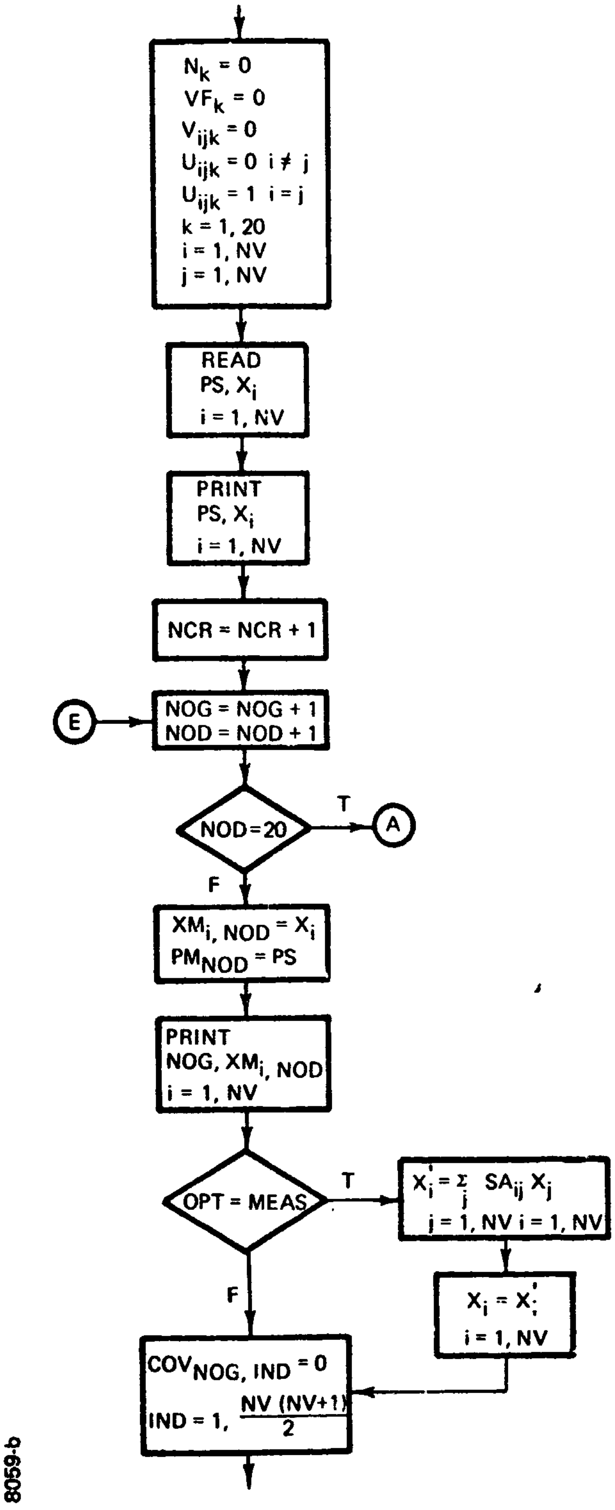


Figure 1-14 (Cont.)

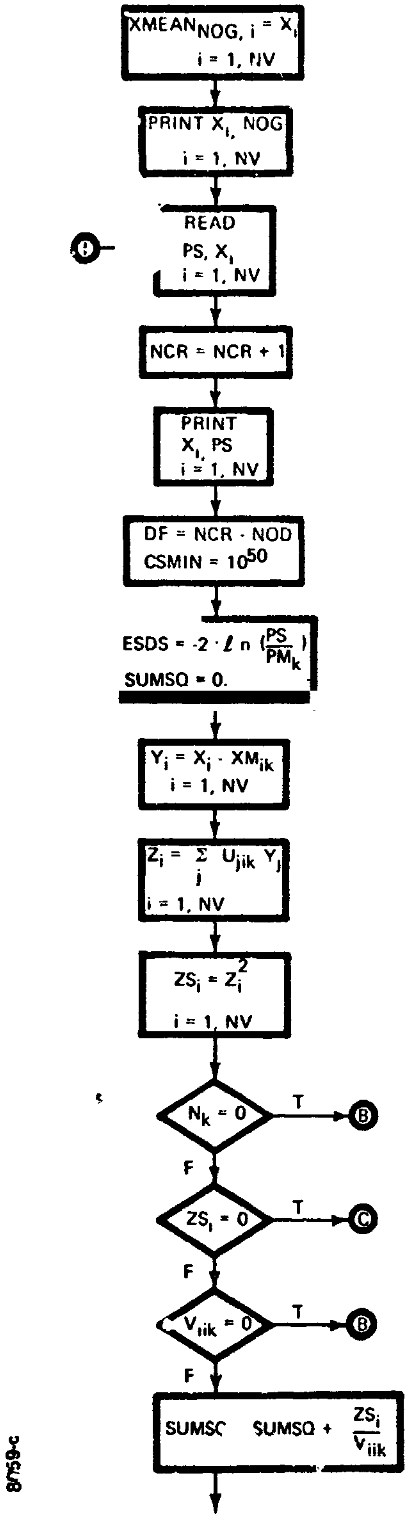


Figure 1-14 (Cont.)

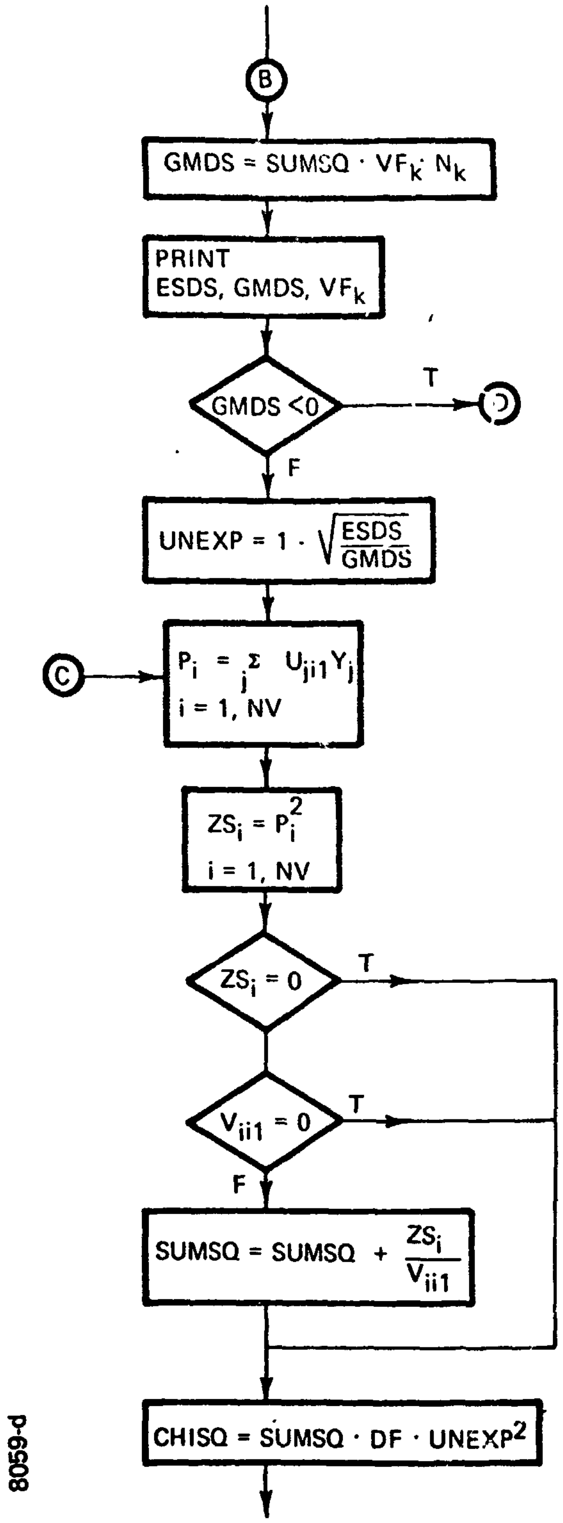

B
$GMDS = SUMSQ \cdot VF_k \cdot N_k$
PRINT
$ESDS, GMDS, VF_k$
$GMDS < 0$
T
D
F
$UNEXP = 1 \cdot \sqrt{\frac{ESDS}{GMDS}}$
C
$P_i = \sum_j U_{ji1} Y_j$
$i = 1, NV$
$ZS_i = P_i^2$
$i = 1, NV$
$ZS_i = 0$
T
$V_{ii1} = 0$
T
F
$SUMSQ = SUMSQ + \frac{ZS_i}{V_{ii1}}$
$CHISQ = SUMSQ \cdot DF \cdot UNEXP^2$
8059-d

Figure 1-14 (Cont.)

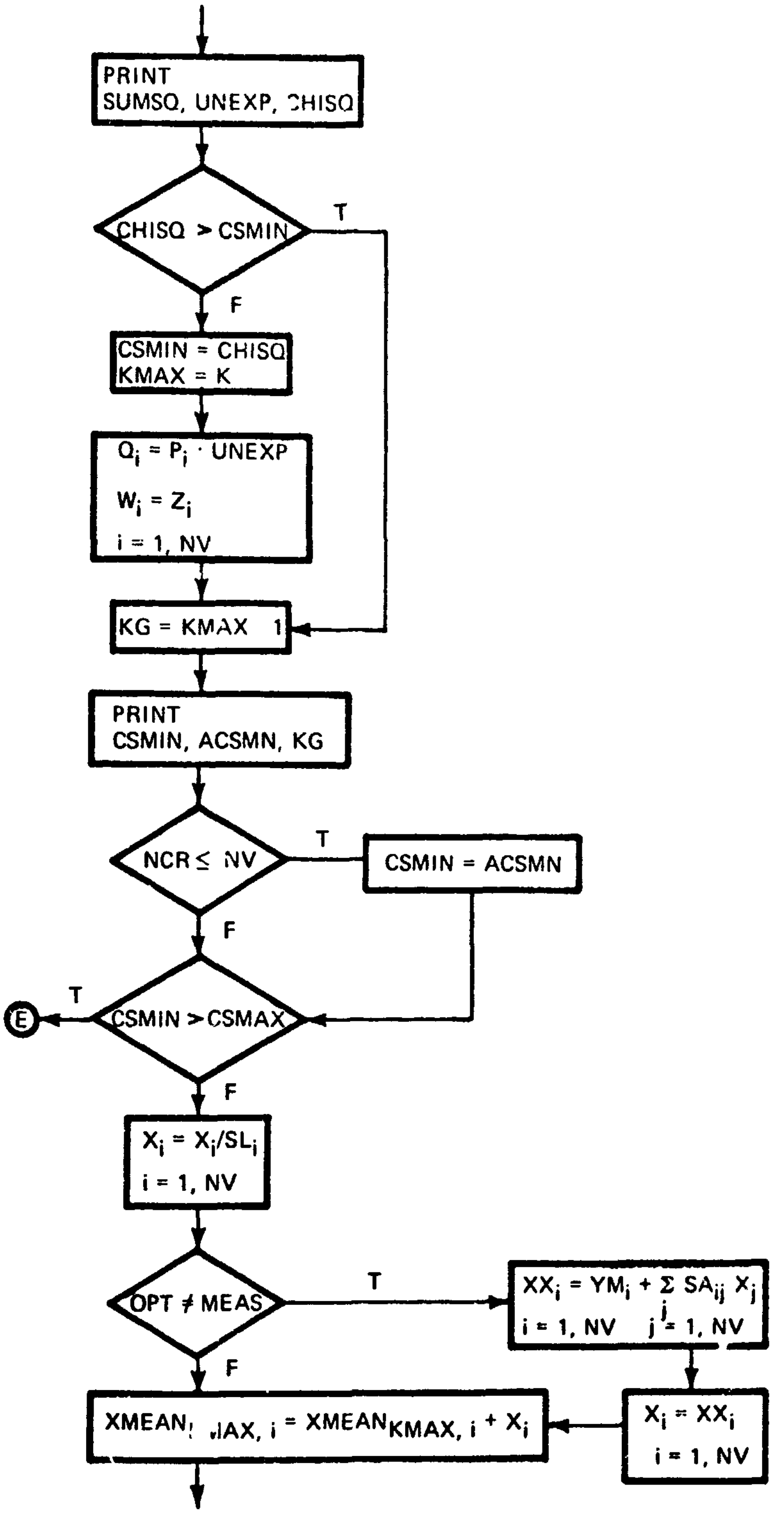


Figure 1-14 (Cont.)

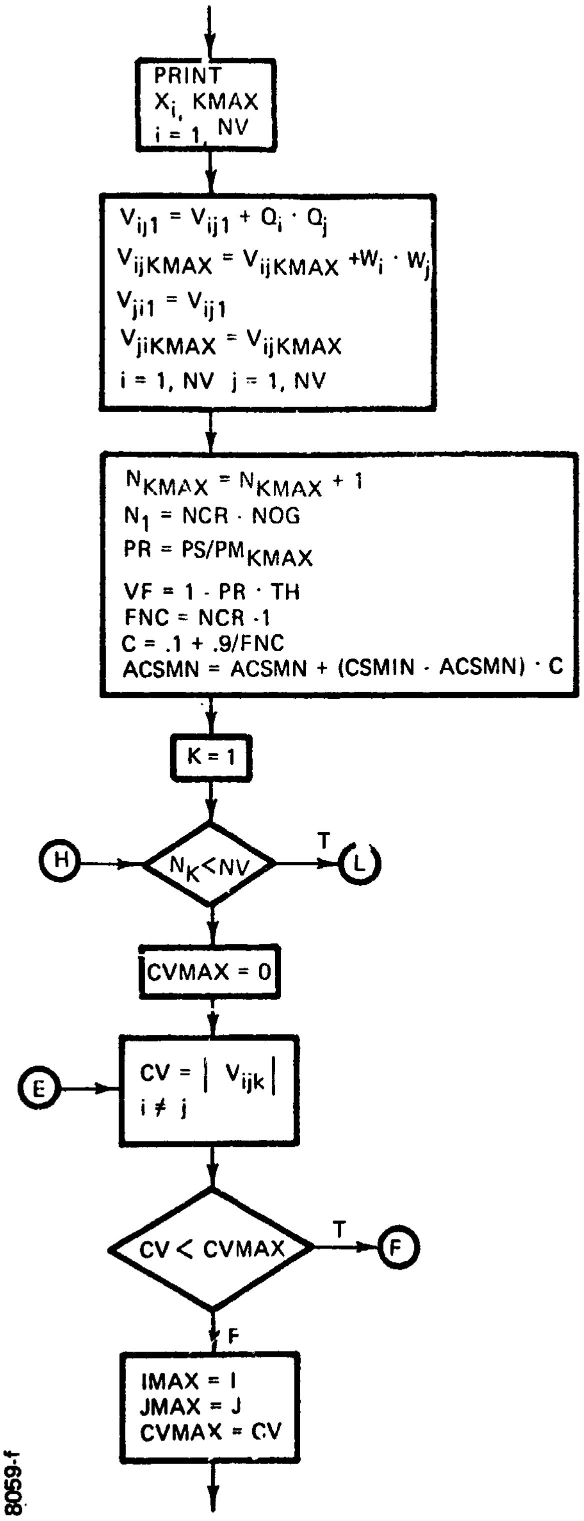


Figure 1-14 (Cont.)

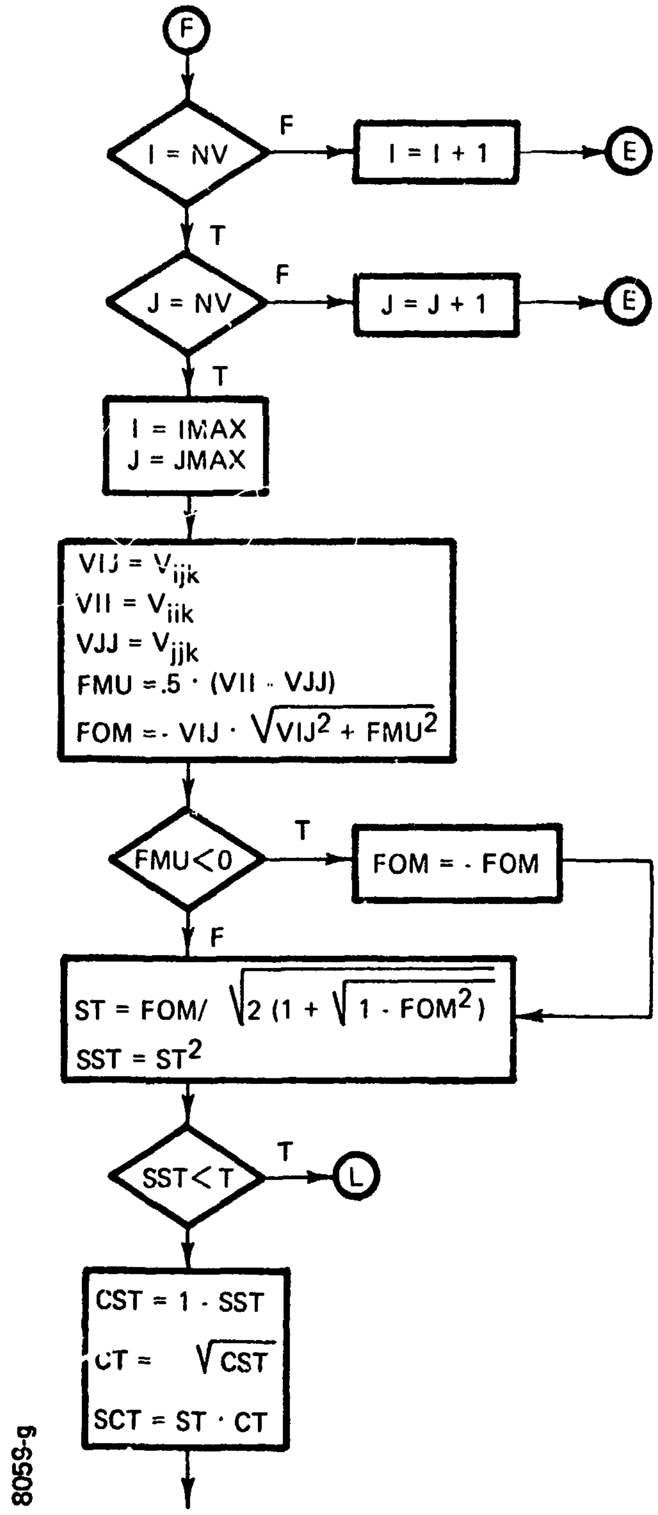


Figure 1-14 (Cont.)

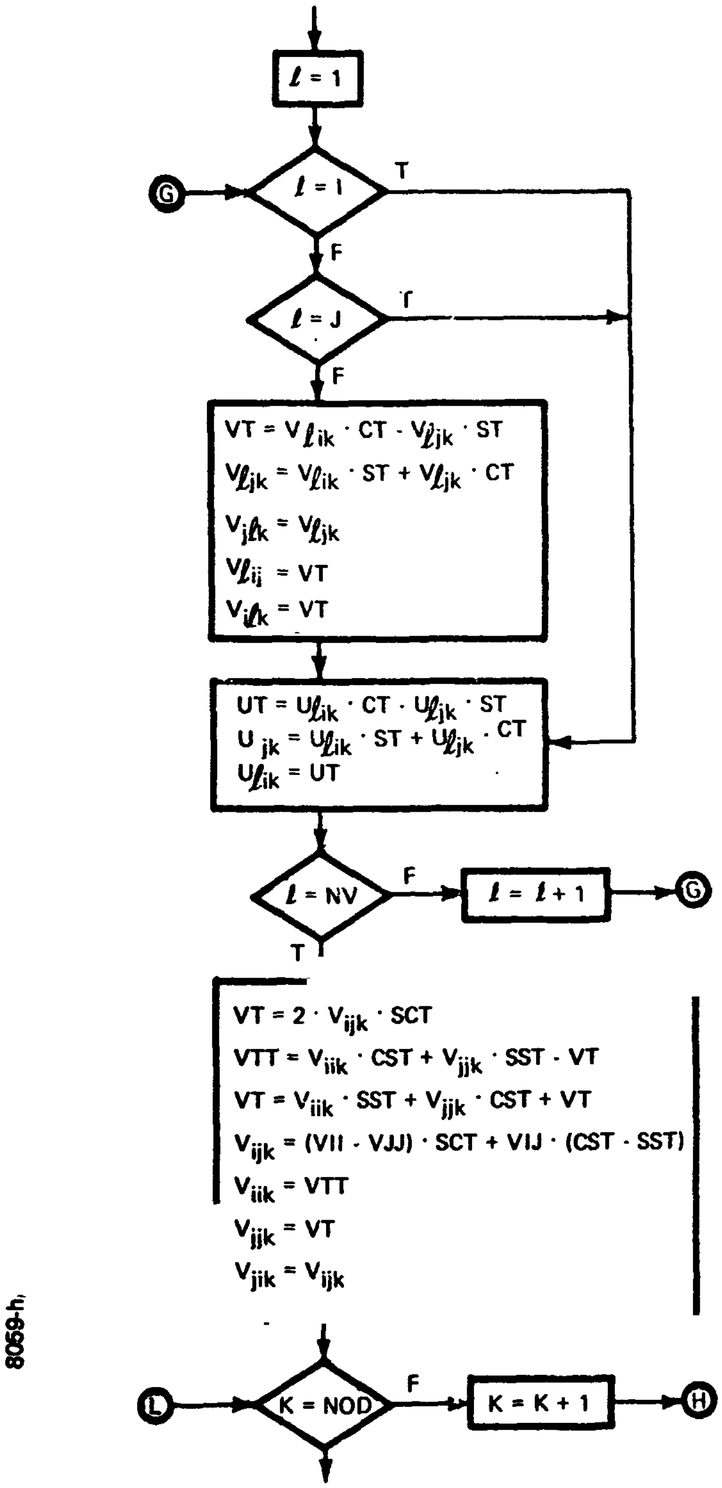

$\ell = 1$
G
$\ell = I$
T
F
$\ell = J$
T
F
$VT = V_{\ell ik} \cdot CT - V_{\ell jk} \cdot ST$
$V_{\ell jk} = V_{\ell ik} \cdot ST + V_{\ell jk} \cdot CT$
$V_{j\ell k} = V_{\ell jk}$
$V_{\ell ij} = VT$
$V_{i\ell k} = VT$
$UT = U_{\ell ik} \cdot CT - U_{\ell jk} \cdot ST$
$U_{jk} = U_{\ell ik} \cdot ST + U_{\ell jk} \cdot CT$
$U_{\ell ik} = UT$
$\ell = NV$
F
$\ell = \ell + 1$
G
T
$VT = 2 \cdot V_{ijk} \cdot SCT$
$VTT = V_{iik} \cdot CST + V_{jjk} \cdot SST - VT$
$VT = V_{iik} \cdot SST + V_{jjk} \cdot CST + VT$
$V_{ijk} = (VII - VJJ) \cdot SCT + VIJ \cdot (CST - SST)$
$V_{iik} = VTT$
$V_{jjk} = VT$
$V_{jik} = V_{ijk}$
L
K = NOD
F
K = K + 1
H
8059-h

Figure 1-14 (Cont.)

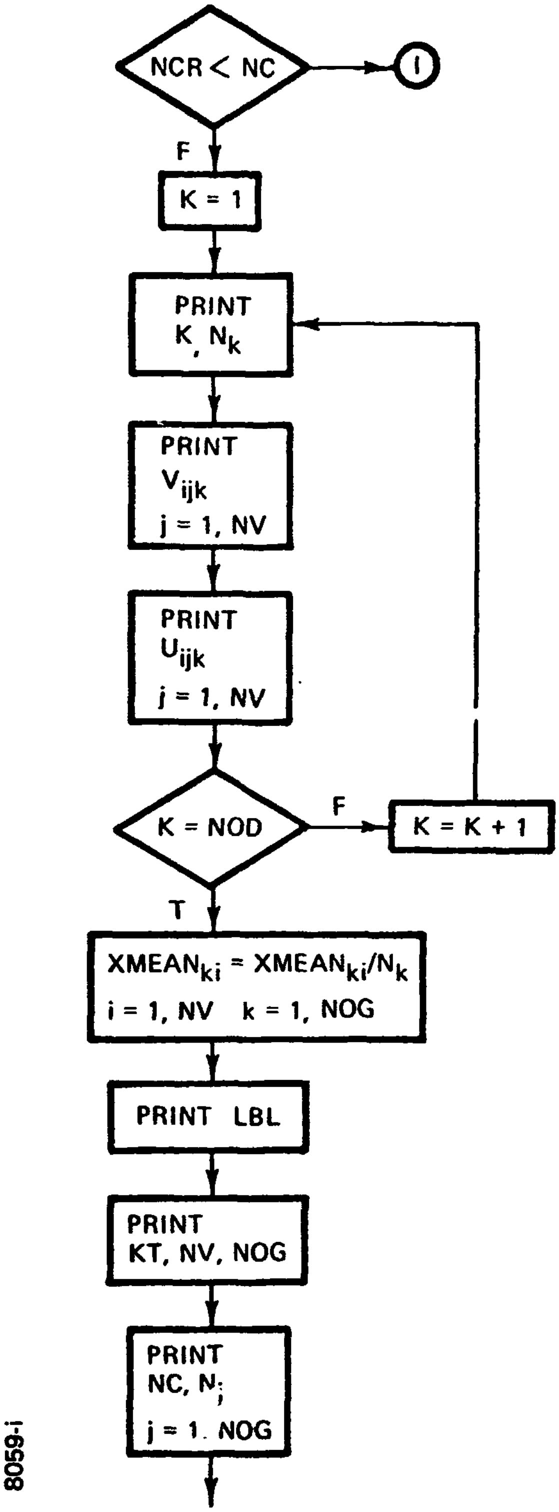
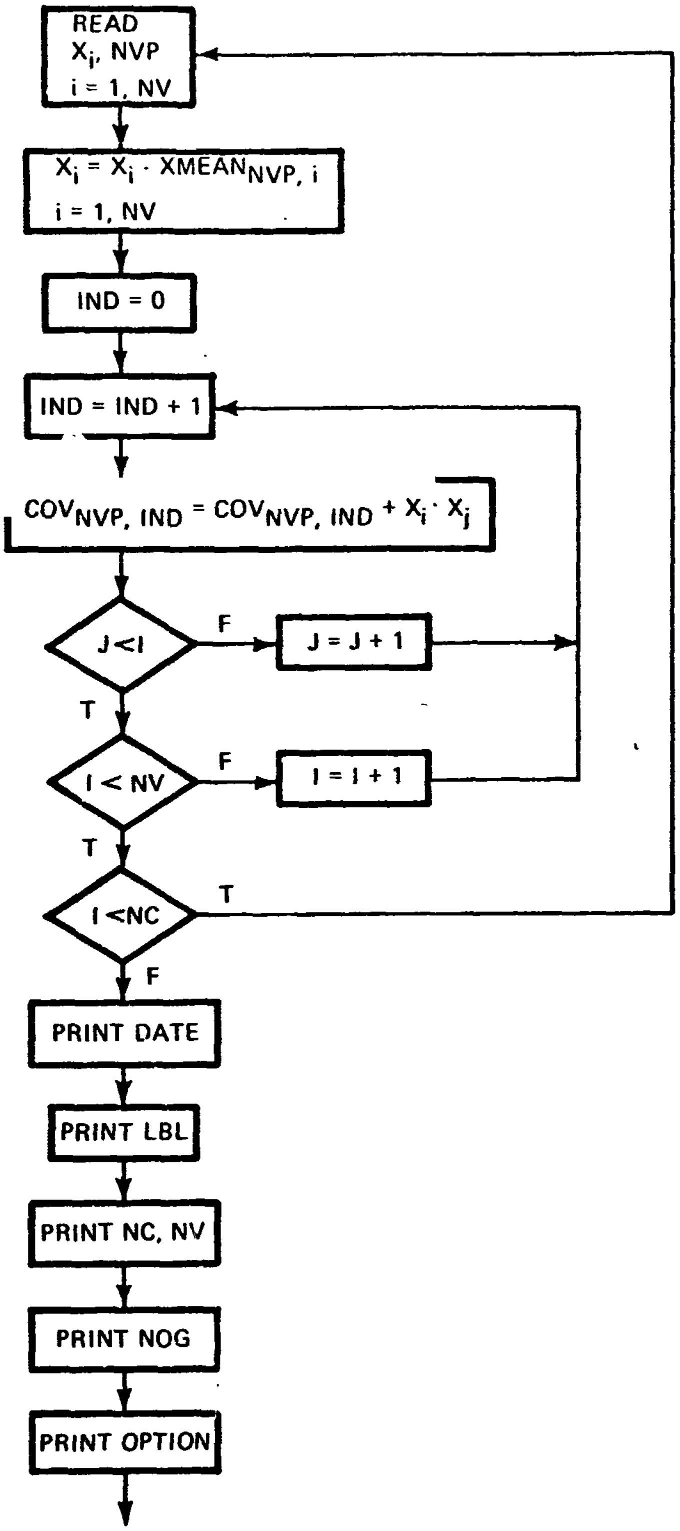


Figure 1-14 (Cont.)



8059-j

Figure 1-14 (Cont.)

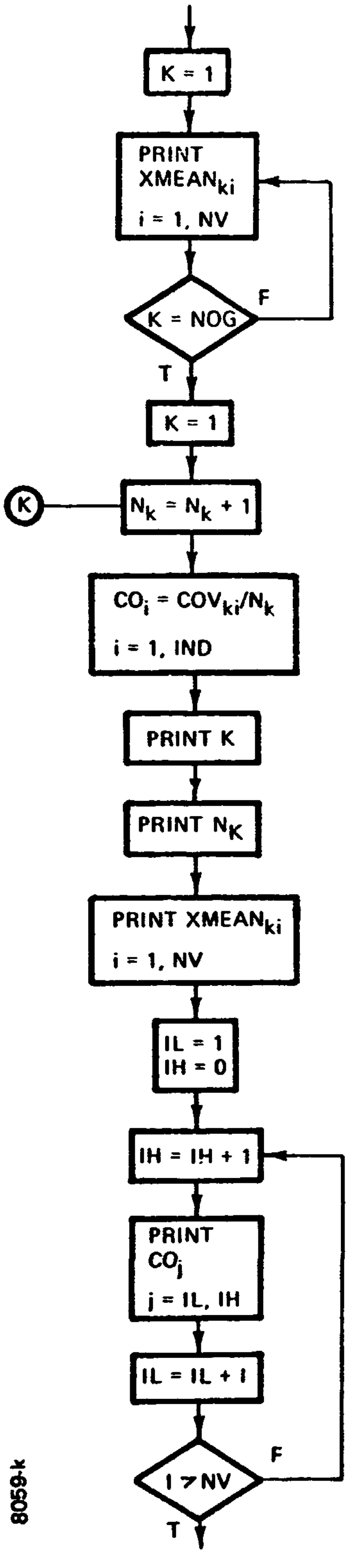
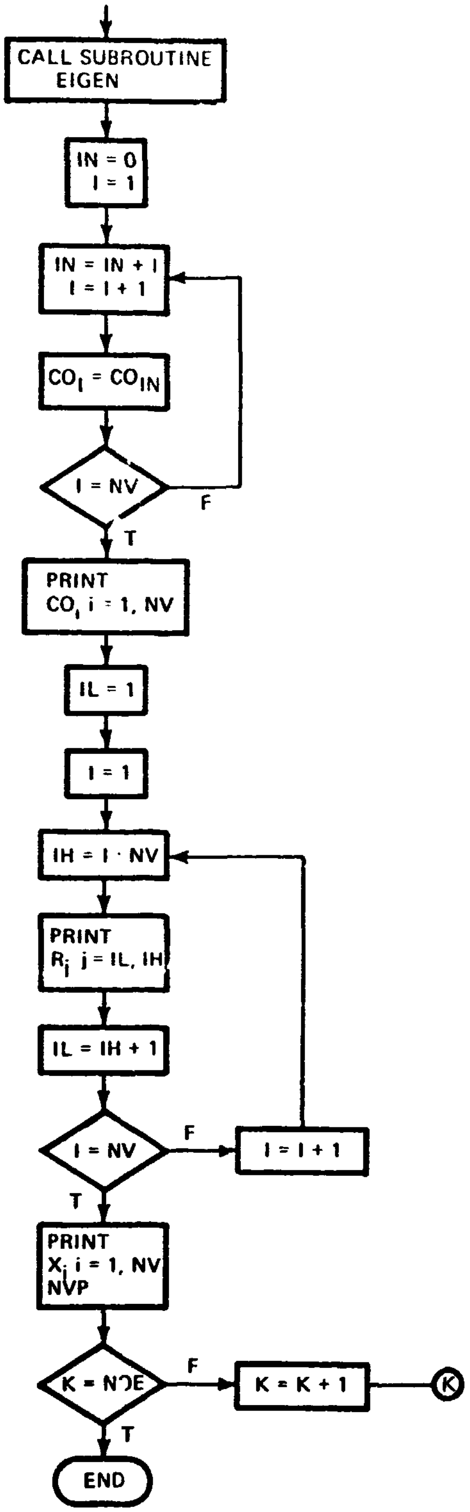


Figure 1-14 (Cont.)



8059-I

Figure 1-14 (Cont.)

SECTION 2

UTILITY PROGRAMS

Several of the computer programs produced during the study are not of any real theoretical interest in terms of advancing the ability to perform spectral pattern recognition. Hoever, they perform necessary clerical and data handling services in preparation for the programs that actually perform multivariate data analysis and classification. The functions performed by these programs will be required by any signature data processing system, although the details of those discussed here are peculiar to the requirements of the hardware used. Thus, only the functions performed are described.

2.1 TAPE EDITOR PROGRAM

Analog video is sampled with an eight-channel A/D converter under push-button control by the operator. The results are recorded on a computer-compatible incremental tape recorder. While the tapes produced in this way are computer-compatible, the format is not Fortran-compatible. The Tape Editor Program accepts the output of the A/D converter and converts it to a record length and format which can be read by Fortran programs. During this process, incomplete records and records containing readings outside the scanner range specified by the user are deleted.

2.2 DATA SELECTION PROGRAM

The Data Selection Program regroups existing sets of data, as in selecting homogeneous samples for collating signatures with imagery. Input to the program may be the output of the Tape Editor or output from a previous run of the Data Selection or Subsampling Programs, along with the case numbers of the signatures to be extracted to form a new data set. The user may choose an option of recording the original Tape Editor output observation number along with the signature selected, for ease in associating ground truth information.

2.3 SUBSAMPLING PROGRAM

The Subsampling Program selects random subsamples of a specified length from existing sets of data. Input may be in the form of digitized scanner data or continuous data with or without case numbers of the Tape Editor output appended. If input does not already have case numbers appended, the user may choose an option to append them in this program.

2.4 DATA TRANSFORMATION PROGRAM

The Data Transformation Program is designed to accept any number of factor score coefficient matrices produced by analysis about the mean or origin, for transforming any number of sets of data in digitized scanner or continuous form. Several computation options are available to the user as well:

1. A factor score correlation matrix may be computed from resultant transformed values.

2. A data correlation matrix may be computed from raw input data.

3. A data covariance matrix may be computed from raw input data.

2.5 HISTOGRAM TALLY PROGRAM

The Histogram Tally Program produces univariate histograms for each channel in the form of a frequency distribution by interval. Input may be either continuous data, digitized scanner data, or factor scores along with the number of intervals desired in the histogram. Output includes sample size, number of intervals, midpoint of first interval, length of interval, and observed frequency in each interval.

2.6 PLOTTING ROUTINES

The Scatter Diagram Program produces a Calcomp Pen Plotter scatter diagram of values of any two specified channels. The user specifies the scale on the axes of the plot and may input continuous, digitized scanner, or factor score data. Digitized scanner data are converted to continuous data before plotting by use of the upper and lower bounds of scanner range.

The Histogram Plotter Program accepts as input the output of the Histogram Tally Program, and produces a Calcomp Pen Plotter graph per channel of the frequencies in each interval. Scaling of the axes is handled automatically by the program.