IBM

Application Program

GY20-0092-2

System/360 Scientific Subroutine Package

(360A-CM-03X) – Version 3

System Manual

This manual provides flowchart information on the logic used in each subroutine of the System/360 Scientific Subroutine Package. All subroutine descriptions and listings are contained in the User's Manual.

Third Edition

This edition, Y20-0092-2, is a major revision of, and obsoletes, Y20-0092-1.

Significant changes or additions to the specifications contained in this publication will be reported in subsequent revisions or Technical Newsletters.

Address comments concerning the contents of this publication to IBM, Technical Publications Department, 1133 Westchester Avenue, White Plains, New York 10604.

CONTENTS

INTRODUCTION

Each subroutine in the Scientific Subroutine Package is documented by means of a flowchart. In the Guide to the flowcharts each flowchart is listed under the general heading "Statistics", "Mathematics" and under a subheading relating to its basic function.

SYMBOLS USED

Figure 1 illustrates the various blocks used in the flowcharts and their particular meaning. Lines connecting these blocks are made up of asterisks or periods. Arrows showing the direction of flow are represented by a "V" for downward, a left parenthesis "(" for right to left, and a right parenthesis ")" for left to right. In some cases an X is used to show flow direction. Special symbols are used for mathematical operators within the flowchart boxes. These are:

* for multiplication
& or + for addition
— for subtraction

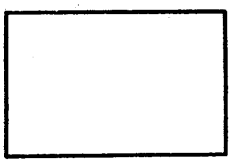
BLOCK NUMBERING

Numbers appearing inside the connector circles refer to the chart block location on the page; these are numbered vertically zero to nine and horizontally A to C. Numbers shown outside the blocks correspond to FORTRAN statement numbers in the programs.

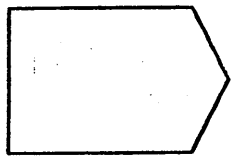
EQUATIONS

Many of the flowcharts contain references to equation numbers. These equations can be found in the mathematical description for the particular subroutines in the User's Manual (H20-0205).

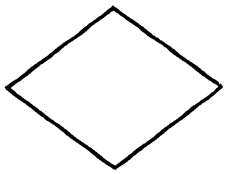
Enter or exit block



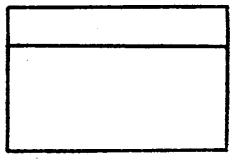
Processor block



Modification block



Decision block



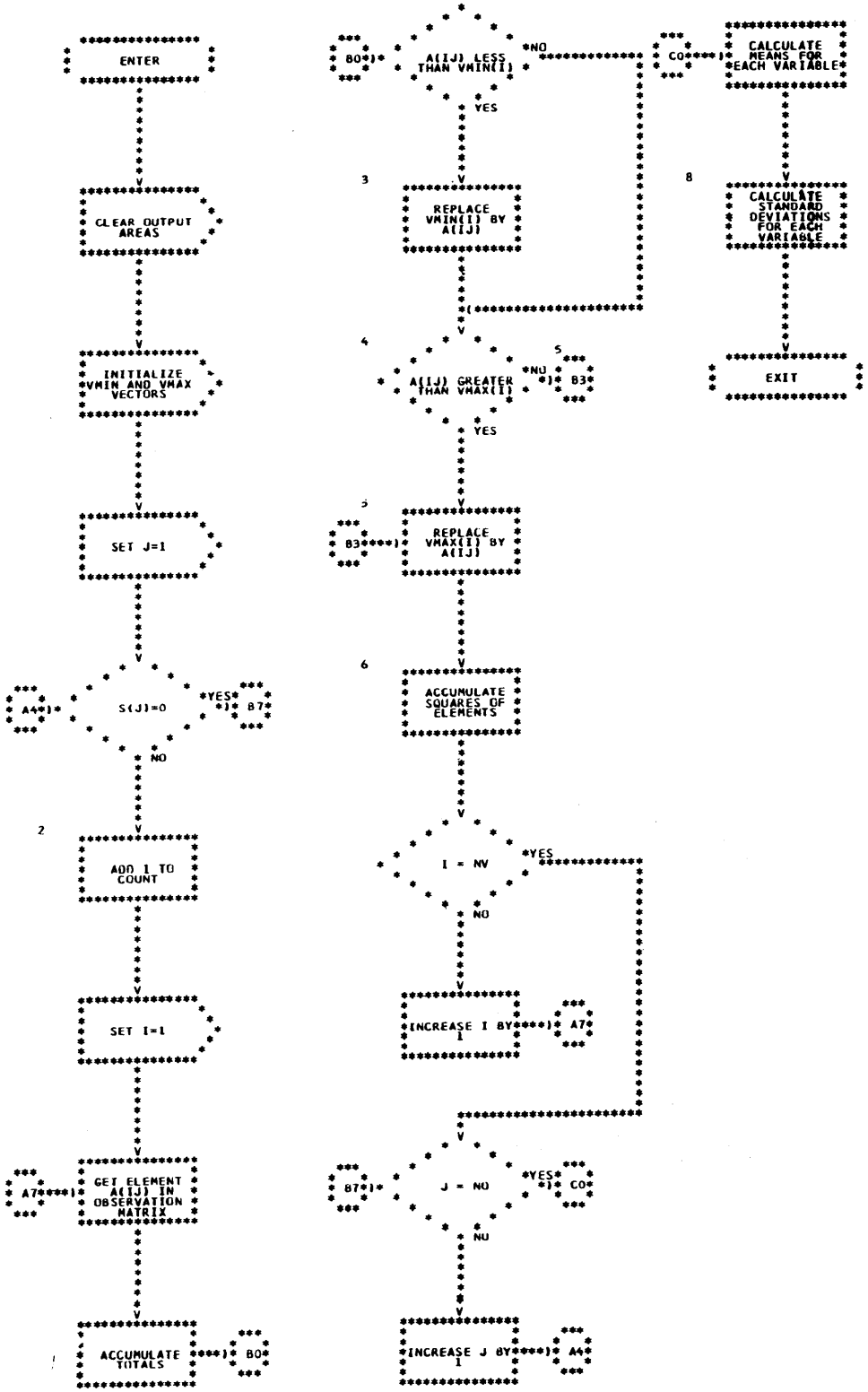
Call to subroutine block

Connector to block within page

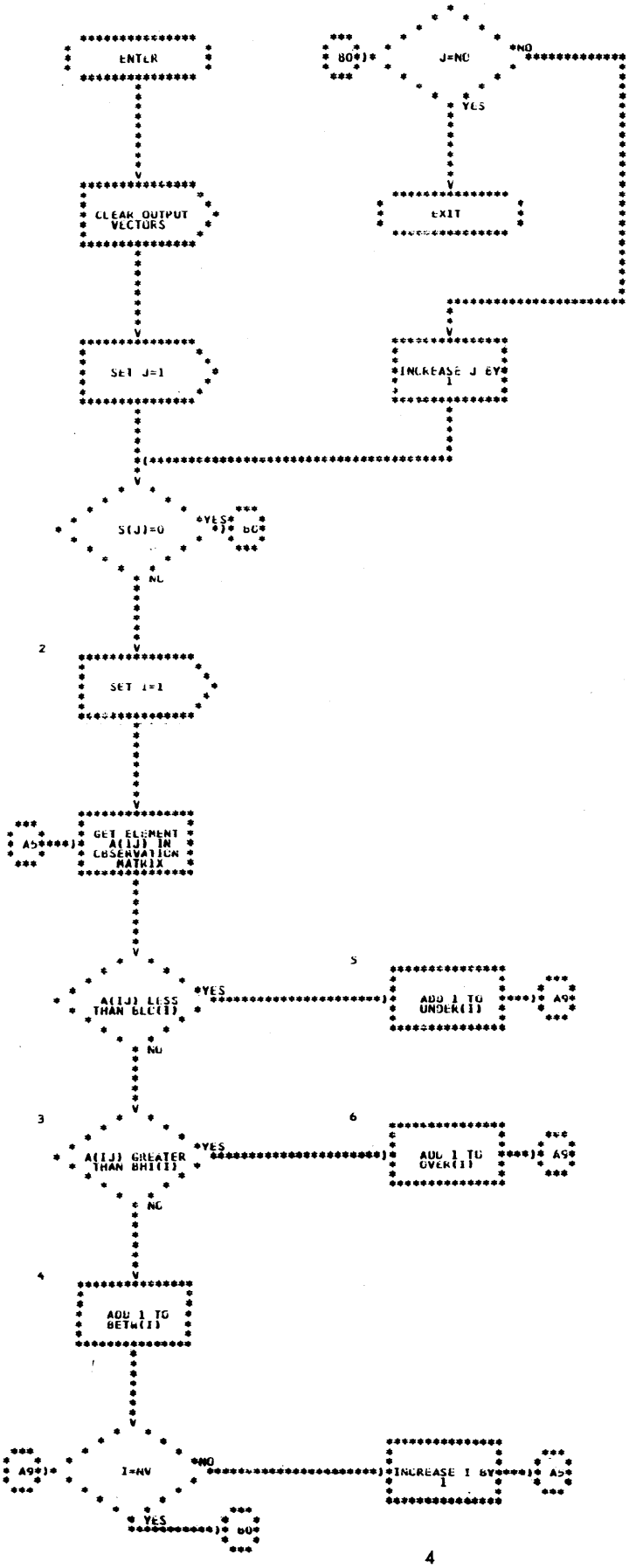
Connector to block on another page

Figure 1. Flowchart blocks

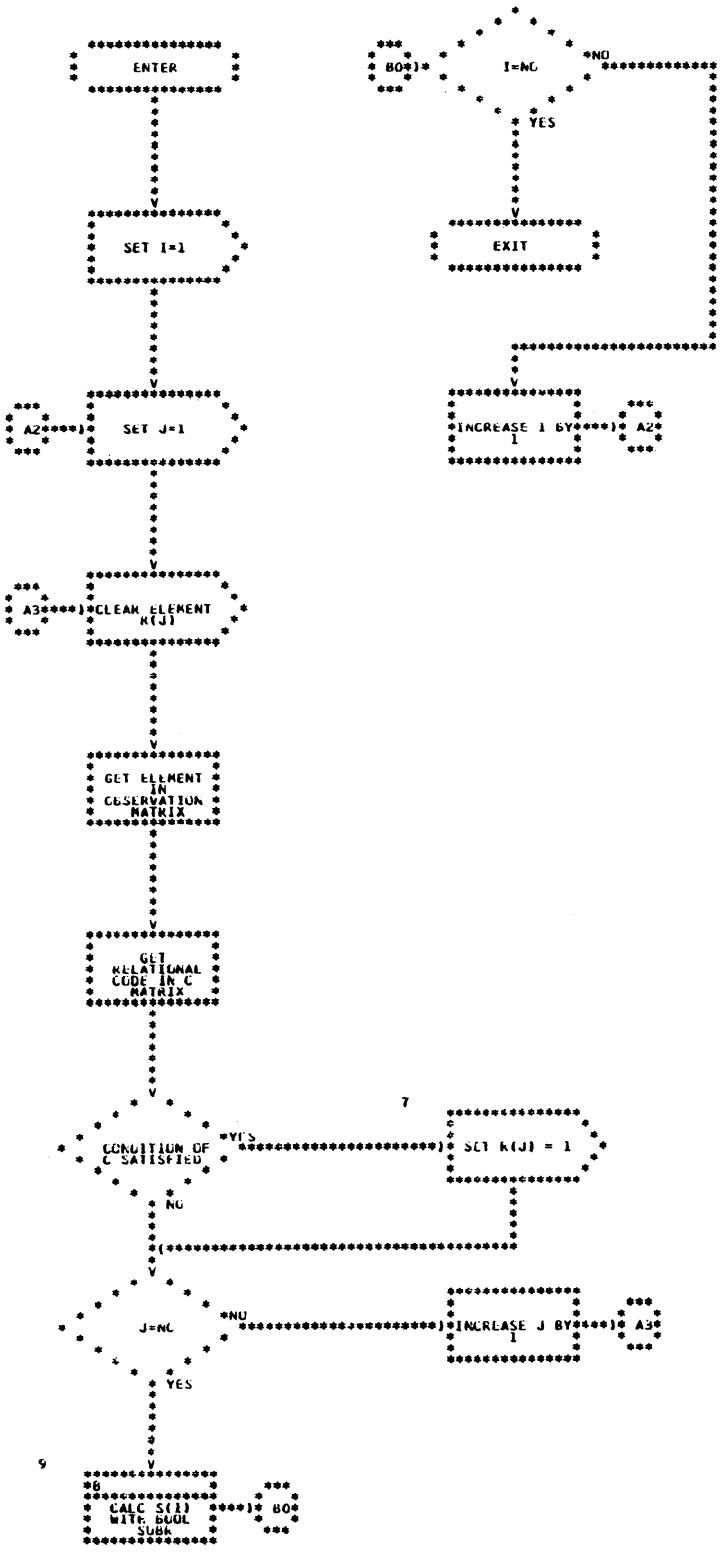
SUBROUTINE TALLY

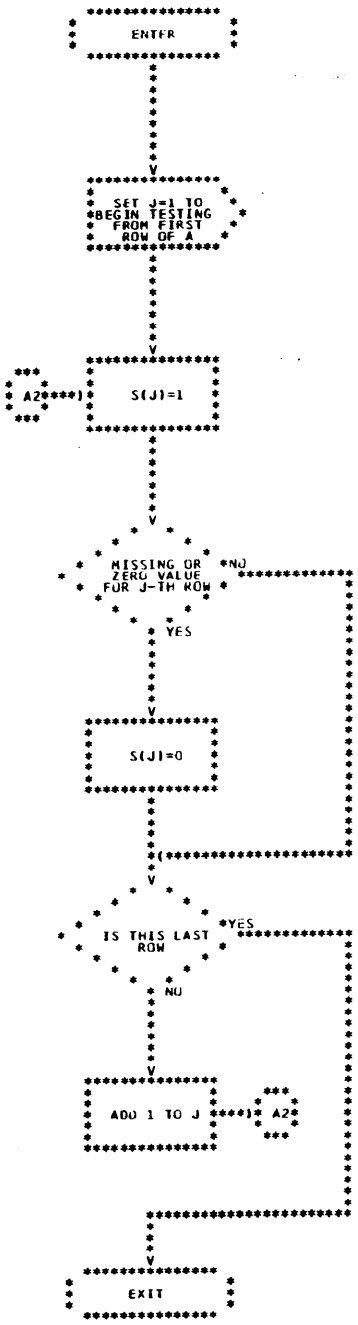


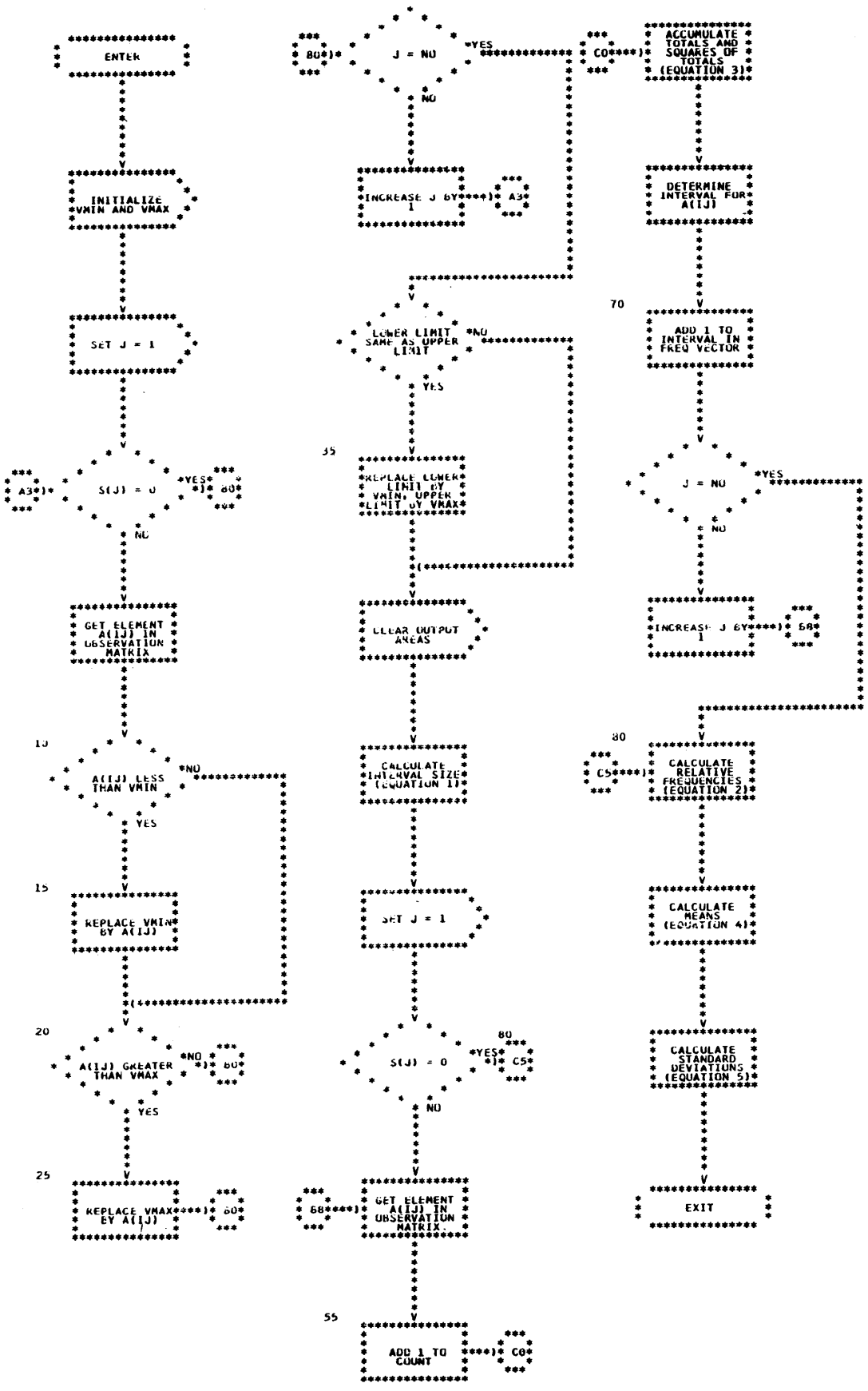
SUBROUTINE BOUND

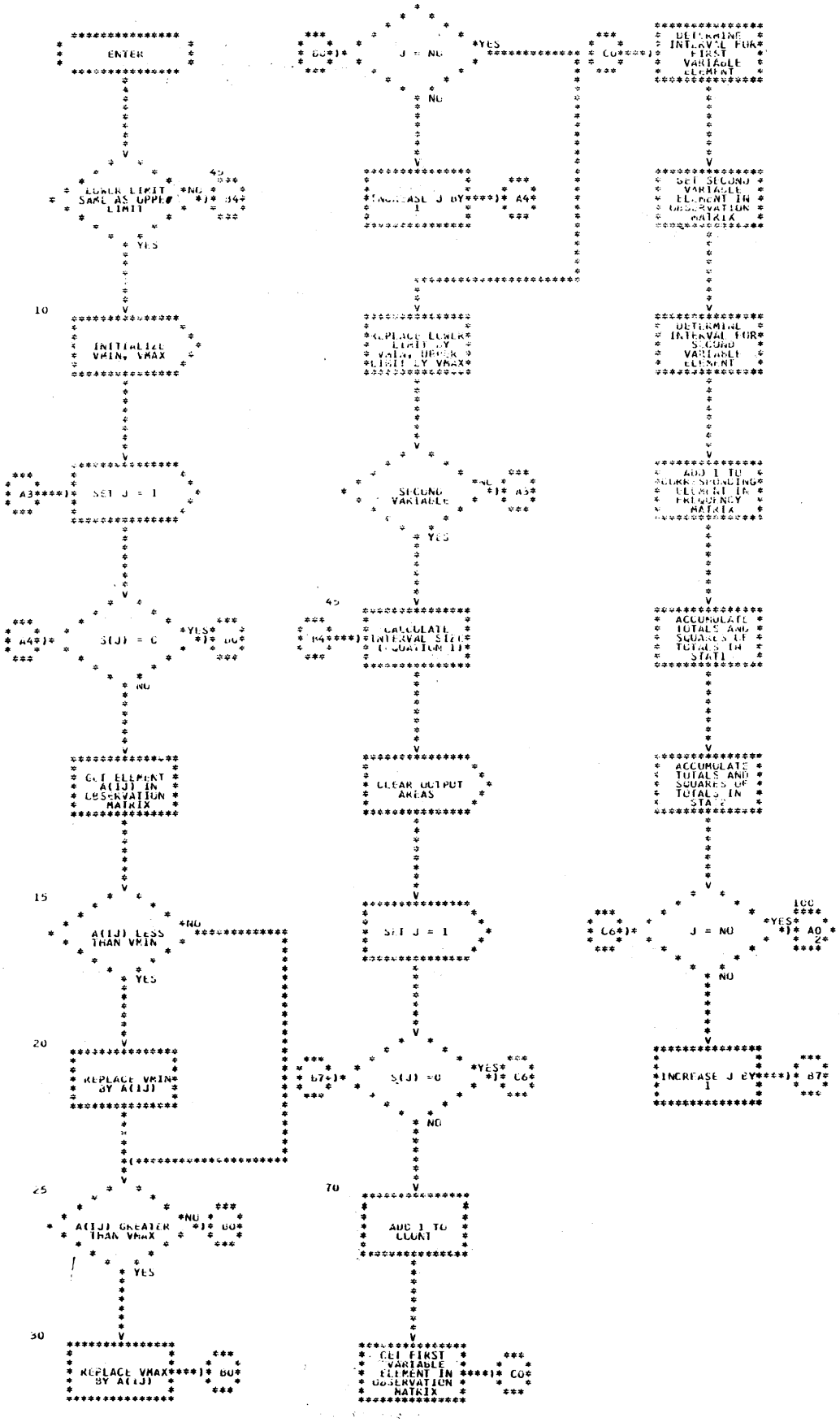


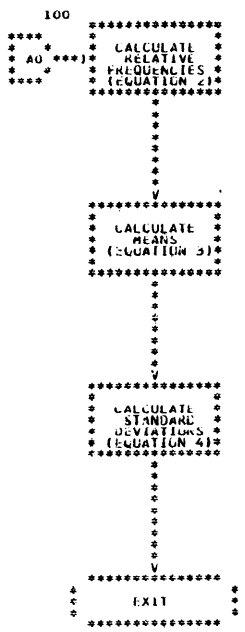
SUBROUTINE SUBST

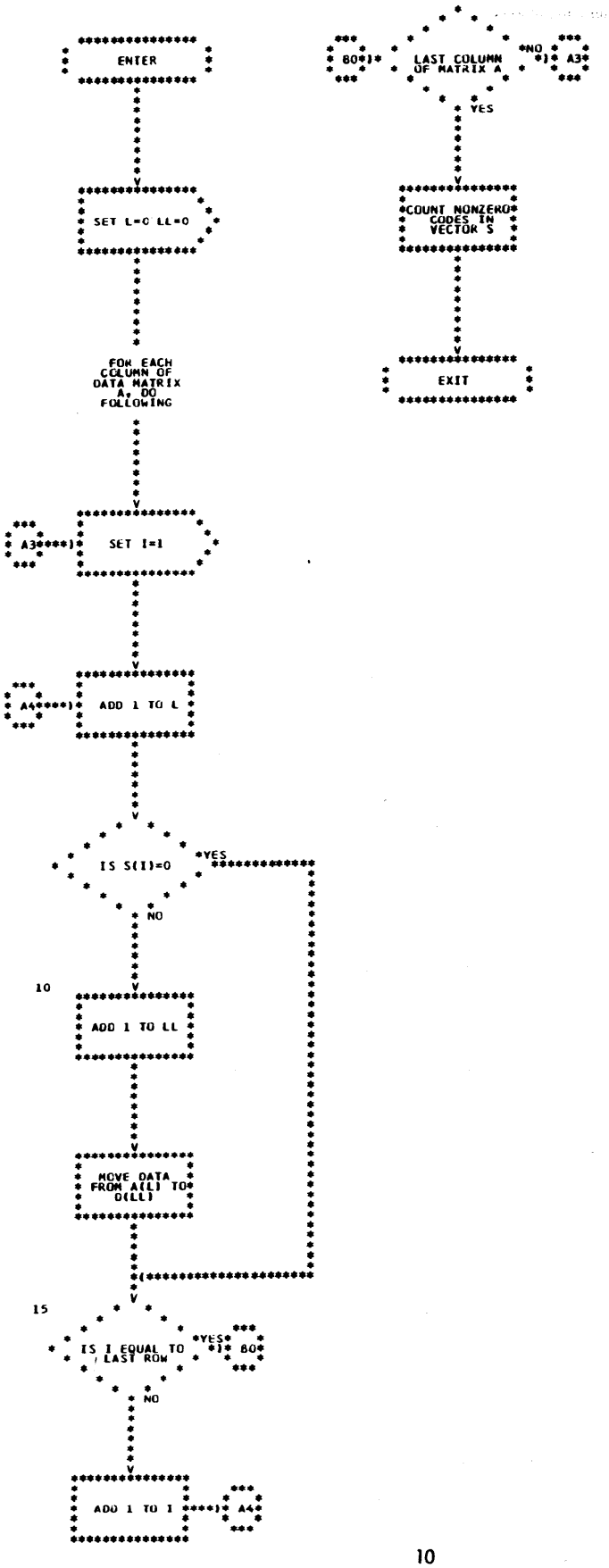


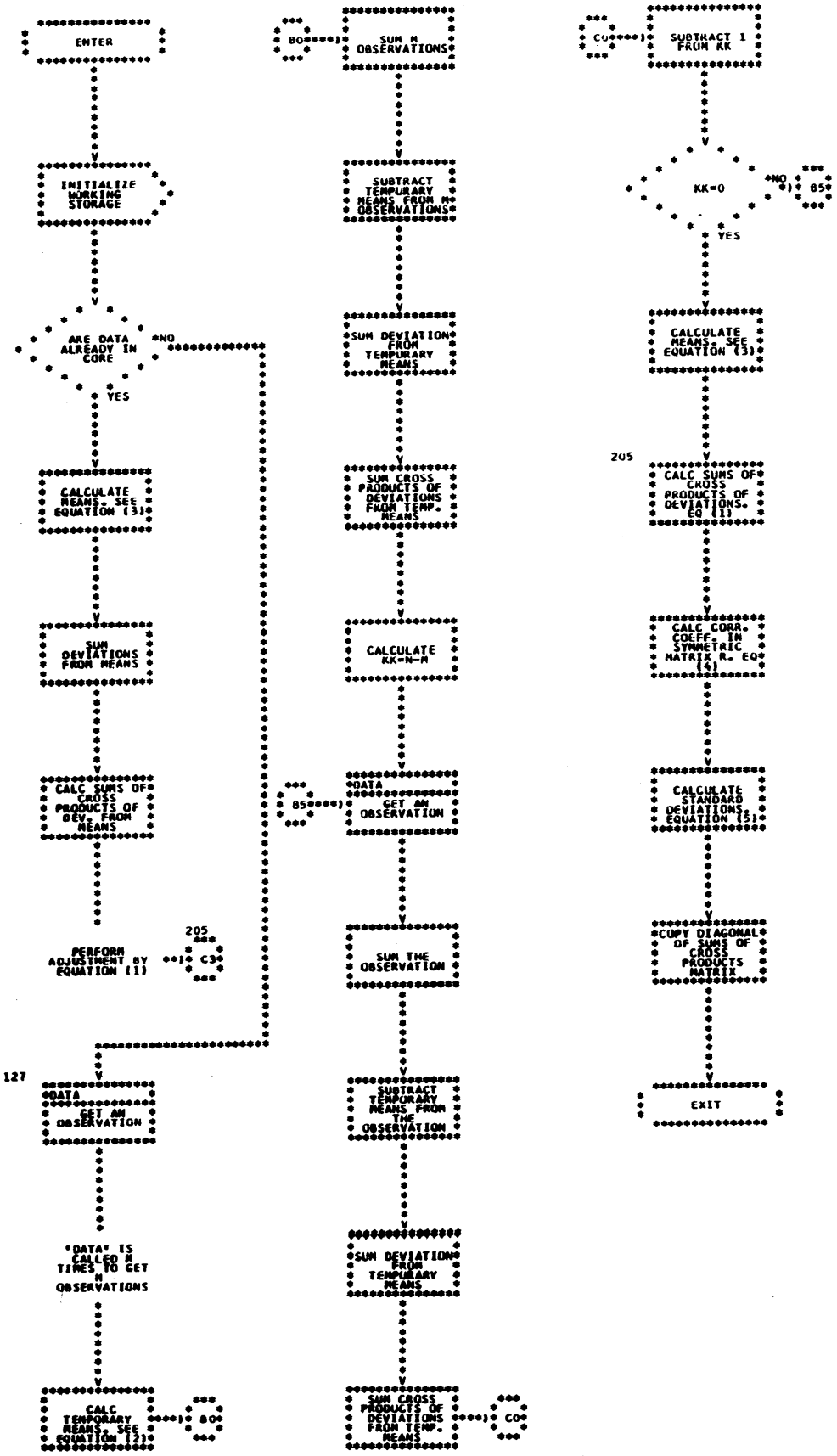

ENTER
SET J=1 TO BEGIN TESTING FROM FIRST ROW OF A
A2
S(J)=1
MISSING OR ZERO VALUE FOR J-TH ROW
NO
YES
S(J)=0
IS THIS LAST ROW
YES
NO
ADD 1 TO J
A2
EXIT

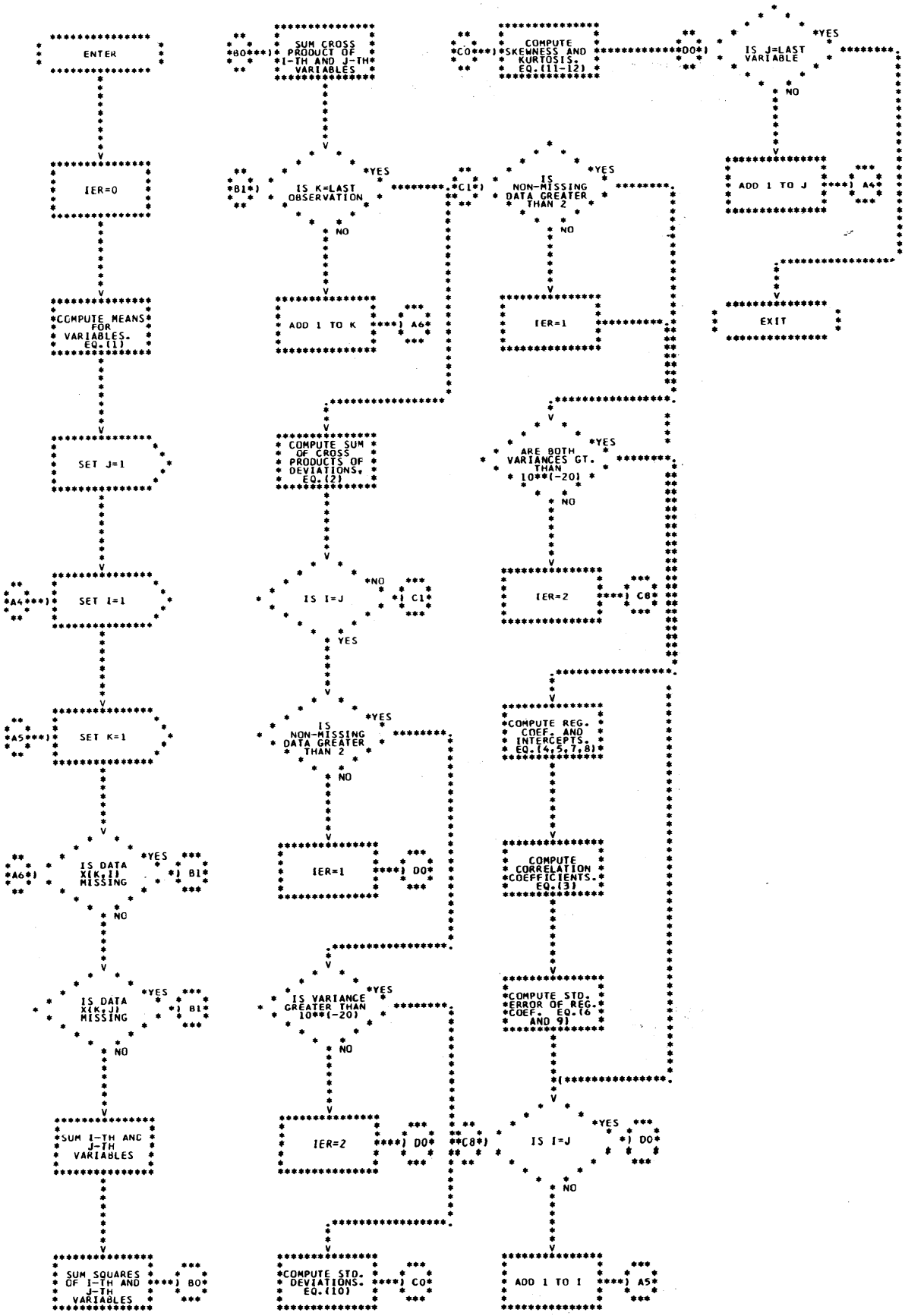

ENTER
INITIALIZE VMIN AND VMAX
SET J = 1
S(J) = 0
YES
NO
A3
B0
GET ELEMENT A(I,J) IN OBSERVATION MATRIX
10
A(I,J) LESS THAN VMIN
NO
YES
15
REPLACE VMIN BY A(I,J)
20
A(I,J) GREATER THAN VMAX
NO
YES
25
REPLACE VMAX BY A(I,J)
B0
J = NO
YES
NO
INCREASE J BY 1
A3
C0
LOWER LIMIT SAME AS UPPER LIMIT
NO
YES
35
REPLACE LOWER LIMIT BY VMIN, UPPER LIMIT BY VMAX
CLEAR OUTPUT AREAS
CALCULATE INTERVAL SIZE (EQUATION 1)
SET J = 1
S(J) = 0
YES
NO
C5
B0
GET ELEMENT A(I,J) IN OBSERVATION MATRIX
B8
55
ADD 1 TO COUNT
C0
ACCUMULATE TOTALS AND SQUARES OF TOTALS (EQUATION 3)
DETERMINE INTERVAL FOR A(I,J)
70
ADD 1 TO INTERVAL IN FREQ VECTOR
J = NO
YES
NO
INCREASE J BY 1
B8
80
C5
CALCULATE RELATIVE FREQUENCIES (EQUATION 2)
CALCULATE MEANS (EQUATION 4)
CALCULATE STANDARD DEVIATIONS (EQUATION 5)
EXIT

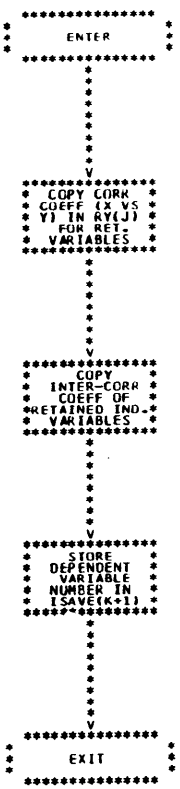

ENTER
LOWER LIMIT SAME AS UPPER LIMIT
NO
YES
INITIALIZE VMIN, VMAX
SET J = 1
S(J) = 0
YES
NO
GET ELEMENT A(I,J) IN OBSERVATION MATRIX
A(I,J) LESS THAN VMIN
YES
NO
REPLACE VMIN BY A(I,J)
A(I,J) GREATER THAN VMAX
YES
NO
REPLACE VMAX BY A(I,J)
J = NO
YES
NO
INCREASE J BY 1
REPLACE LOWER LIMIT BY VMIN, UPPER LIMIT BY VMAX
SECOND VARIABLE
YES
NO
CALCULATE INTERVAL SIZE (EQUATION 1)
CLEAR OUTPUT AREAS
SET J = 1
S(J) =0
YES
NO
ADD 1 TO COUNT
GET FIRST VARIABLE ELEMENT IN OBSERVATION MATRIX
DETERMINE INTERVAL FOR FIRST VARIABLE ELEMENT
GET SECOND VARIABLE ELEMENT IN OBSERVATION MATRIX
DETERMINE INTERVAL FOR SECOND VARIABLE ELEMENT
ADD 1 TO CORRESPONDING ELEMENT IN FREQUENCY MATRIX
ACCUMULATE TOTALS AND SQUARES OF TOTALS IN STAT
ACCUMULATE TOTALS AND SQUARES OF TOTALS IN STAT
J = NO
YES
NO
INCREASE J BY 1
10
15
20
25
30
45
70
100

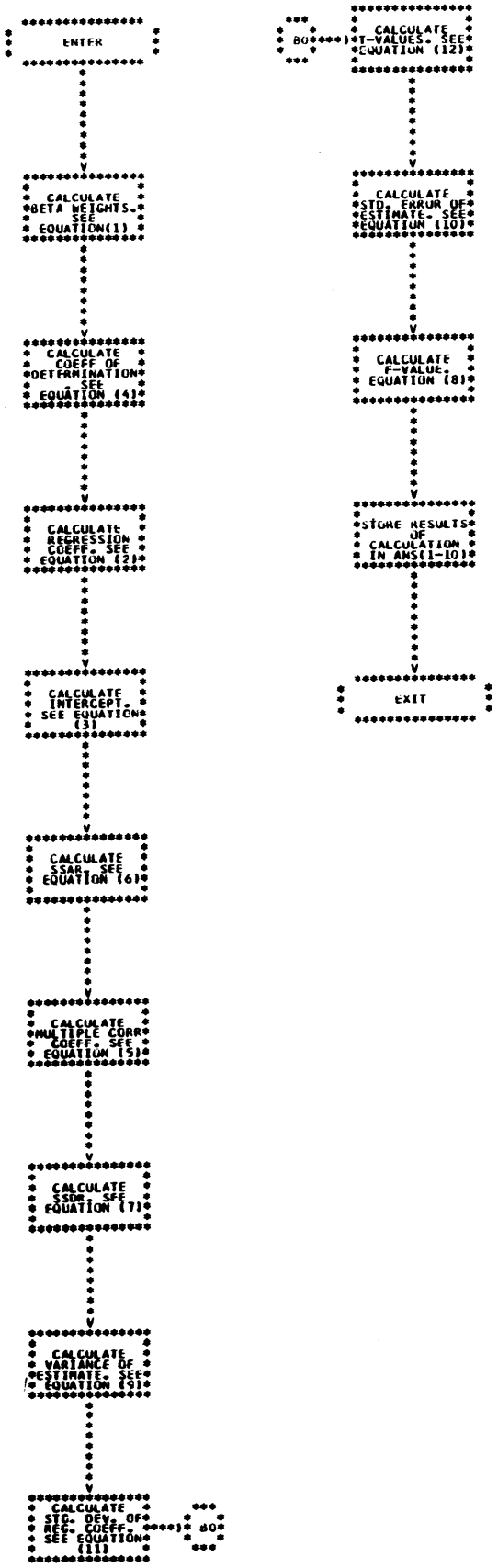

100
CALCULATE RELATIVE FREQUENCIES (EQUATION 2)
A0
CALCULATE MEANS (EQUATION 3)
CALCULATE STANDARD DEVIATIONS (EQUATION 4)
EXIT

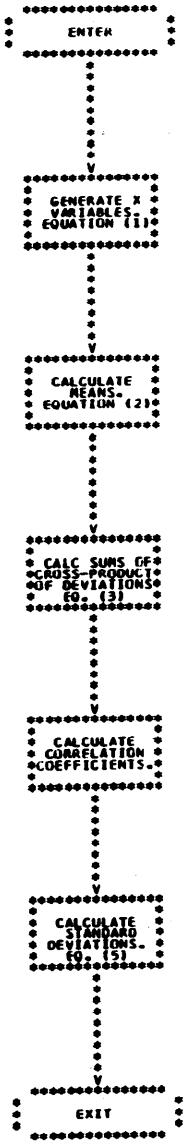

ENTER
SET L=0 LL=0
FOR EACH COLUMN OF DATA MATRIX A, DO FOLLOWING
A3
SET I=1
A4
ADD 1 TO L
IS S(I)=0
YES
NO
10
ADD 1 TO LL
MOVE DATA FROM A(L) TO D(LL)
15
IS I EQUAL TO LAST ROW
YES
B0
NO
ADD 1 TO I
A4
B0
LAST COLUMN OF MATRIX A
NO
A3
YES
COUNT NONZERO CODES IN VECTOR S
EXIT

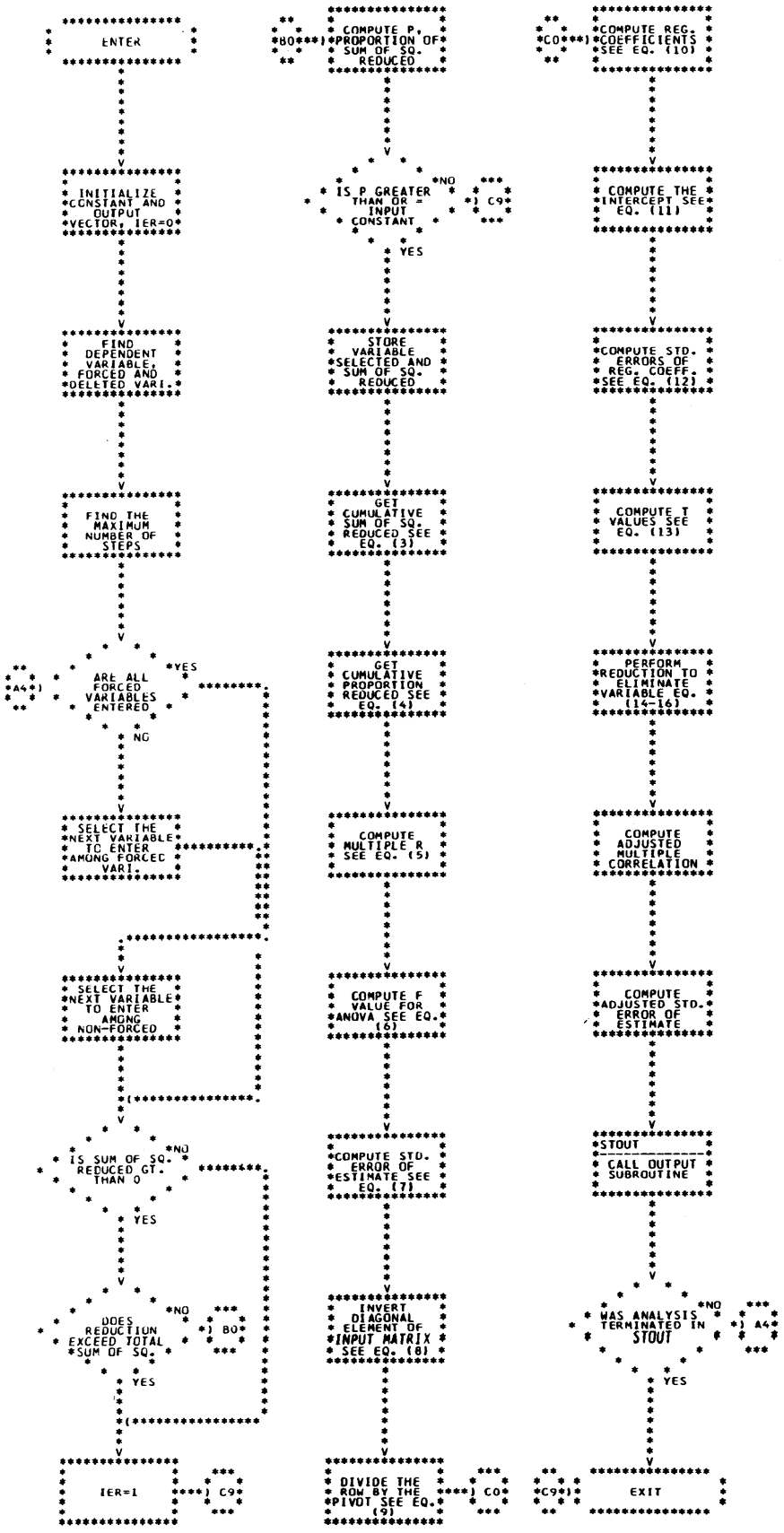

ENTER
INITIALIZE WORKING STORAGE
ARE DATA ALREADY IN CORE
NO
YES
CALCULATE MEANS, SEE EQUATION (3)
SUM DEVIATIONS FROM MEANS
CALC SUMS OF CROSS PRODUCTS OF DEV. FROM MEANS
PERFORM ADJUSTMENT BY EQUATION (1)
205
C3
127
DATA GET AN OBSERVATION
DATA IS CALLED M TIMES TO GET N OBSERVATIONS
CALC TEMPORARY MEANS, SEE EQUATION (2)
80
SUM M OBSERVATIONS
SUBTRACT TEMPURARY MEANS FROM M OBSERVATIONS
SUM DEVIATION FROM TEMPURARY MEANS
SUM CROSS PRODUCTS OF DEVIATIONS FROM TEMP. MEANS
CALCULATE KK=N-M
85
DATA GET AN OBSERVATION
SUM THE OBSERVATION
SUBTRACT TEMPURARY MEANS FROM THE OBSERVATION
SUM DEVIATION FROM TEMPURARY MEANS
SUM CROSS PRODUCTS OF DEVIATIONS FROM TEMP. MEANS
CO
SUBTRACT 1 FROM KK
KK=0
NO
85
YES
CALCULATE MEANS, SEE EQUATION (3)
205
CALC SUMS OF CROSS PRODUCTS OF DEVIATIONS, EQ (1)
CALC CORR. COEFF. IN SYMMETRIC MATRIX R, EQ (4)
CALCULATE STANDARD DEVIATIONS, EQUATION (5)
COPY DIAGONAL OF SUMS OF CROSS PRODUCTS MATRIX
EXIT

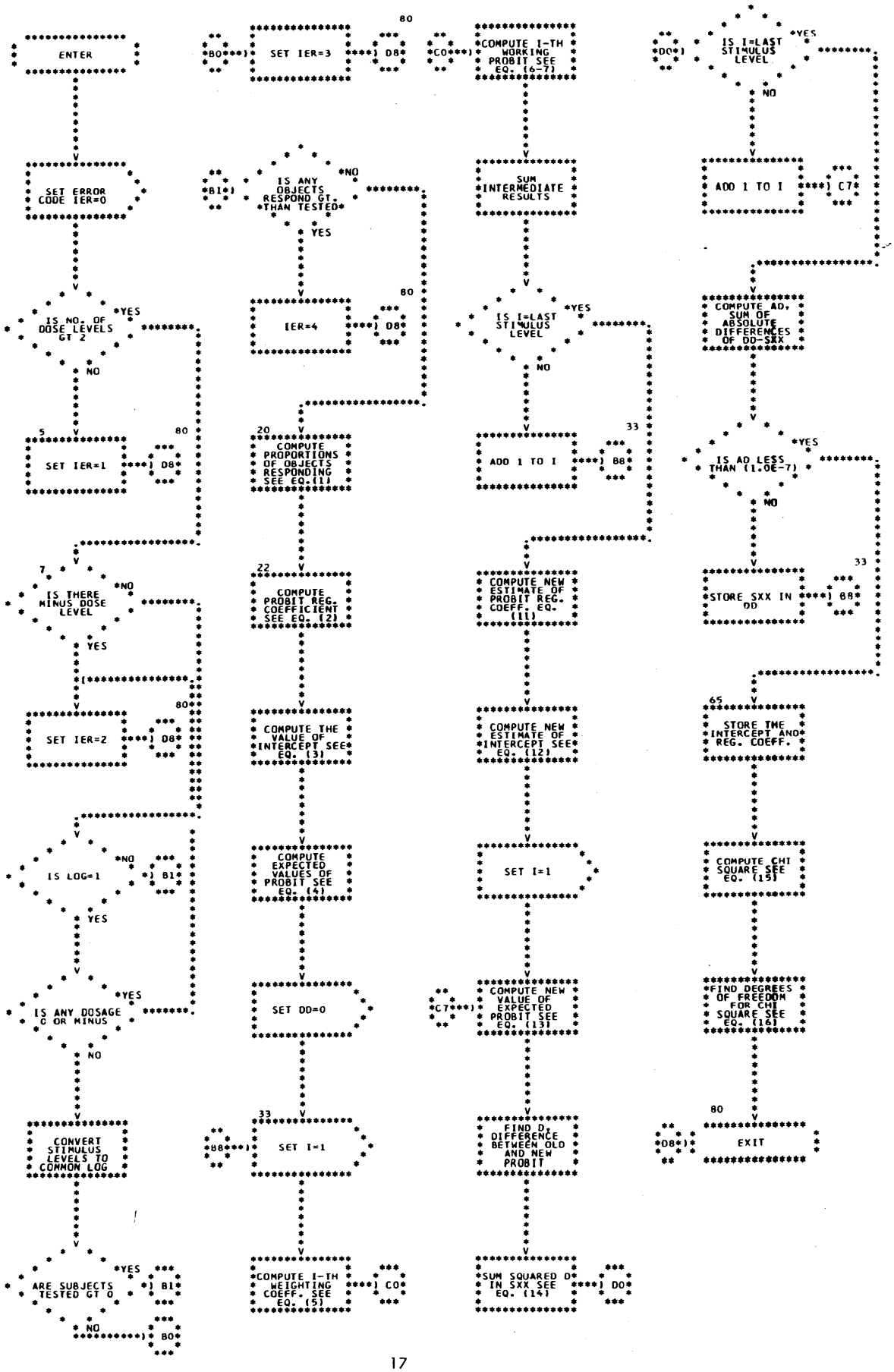

ENTER
IER=0
COMPUTE MEANS FOR VARIABLES. EQ.(1)
SET J=1
SET I=1
SET K=1
IS DATA X(K,I) MISSING
YES
B1
NO
IS DATA X(K,J) MISSING
YES
B1
NO
SUM I-TH AND J-TH VARIABLES
SUM SQUARES OF I-TH AND J-TH VARIABLES
B0
A4
A5
A6
SUM CROSS PRODUCT OF I-TH AND J-TH VARIABLES
IS K=LAST OBSERVATION
YES
NO
ADD 1 TO K
A6
B1
C1
COMPUTE SUM OF CROSS PRODUCTS OF DEVIATIONS. EQ.(2)
IS I=J
NO
C1
YES
IS NON-MISSING DATA GREATER THAN 2
YES
NO
IER=1
D0
IS VARIANCE GREATER THAN 10**(-20)
YES
NO
IER=2
D0
COMPUTE STD. DEVIATIONS. EQ.(10)
C0
C1
C8
COMPUTE SKEWNESS AND KURTOSIS. EQ.(11-12)
C0
D0
IS NON-MISSING DATA GREATER THAN 2
YES
NO
IER=1
ARE BOTH VARIANCES GT. THAN 10**(-20)
YES
NO
IER=2
C8
COMPUTE REG. COEF. AND INTERCEPTS. EQ.(4,5,7,8)
COMPUTE CORRELATION COEFFICIENTS. EQ.(3)
COMPUTE STD. ERROR OF REG. COEF. EQ.(6 AND 9)
C8
IS I=J
YES
D0
NO
ADD 1 TO I
A5
IS J=LAST VARIABLE
YES
NO
ADD 1 TO J
A4
EXIT

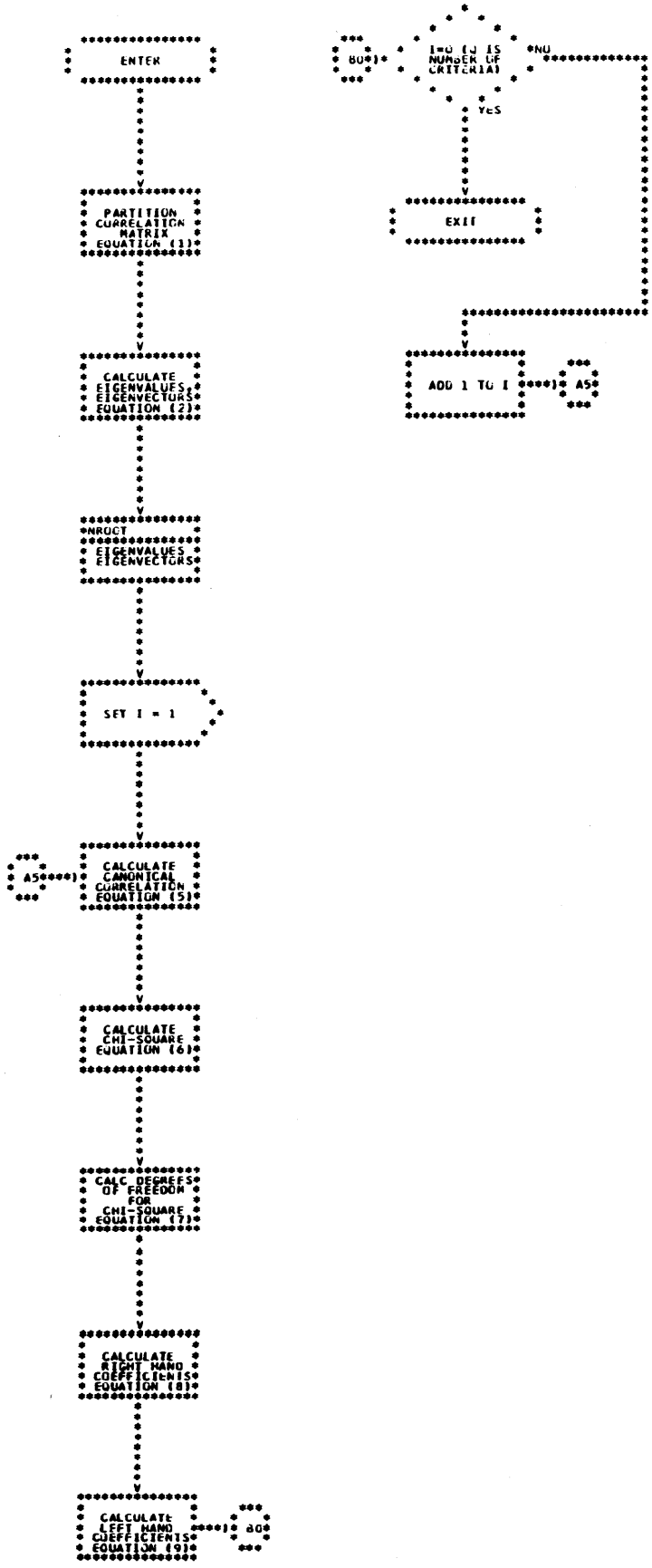

ENTER
COPY CORR COEFF (X VS Y) IN RY(J) FOR RET. VARIABLES
COPY INTER-CORR COEFF OF RETAINED IND. VARIABLES
STORE DEPENDENT VARIABLE NUMBER IN ISAVE(K+1)
EXIT

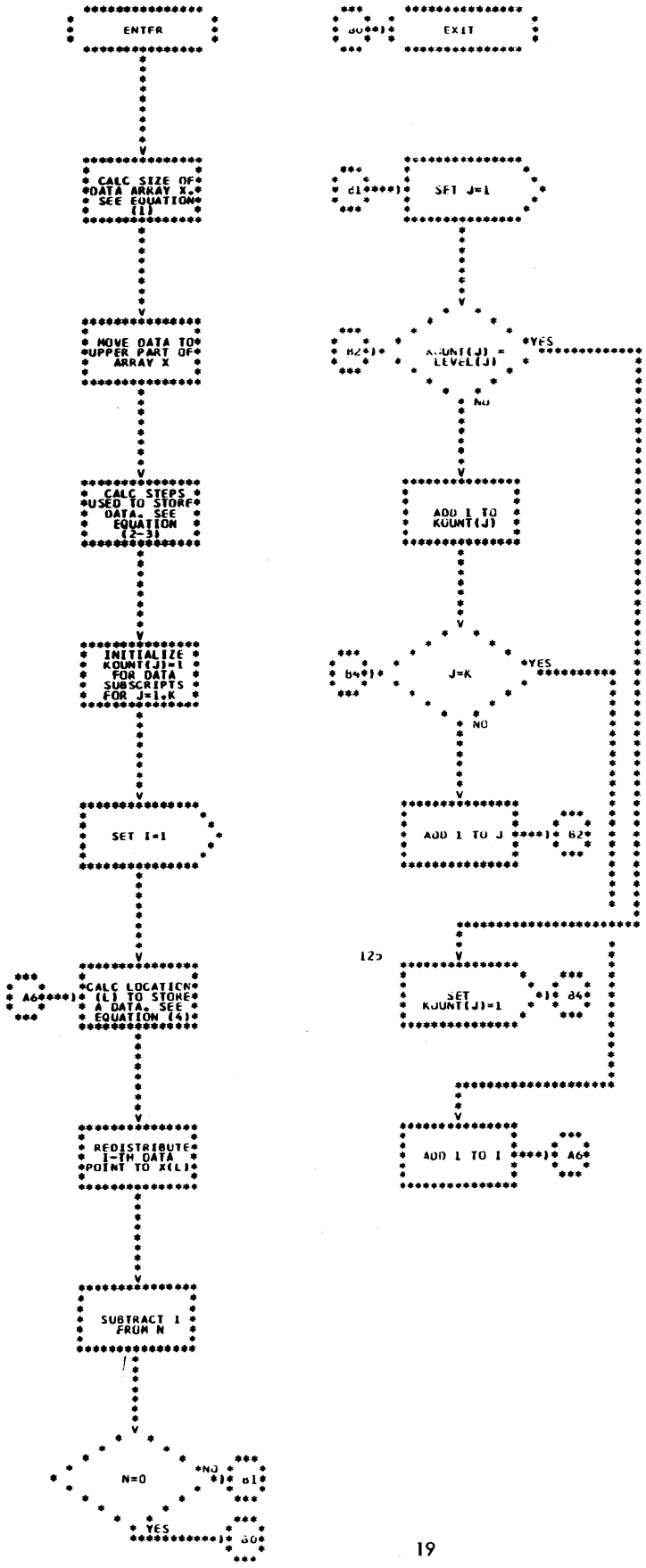

ENTER
CALCULATE BETA WEIGHTS. SEE EQUATION(1)
CALCULATE COEFF OF DETERMINATION SEE EQUATION (4)
CALCULATE REGRESSION COEFF. SEE EQUATION (2)
CALCULATE INTERCEPT. SEE EQUATION (3)
CALCULATE SSAR. SEE EQUATION (6)
CALCULATE MULTIPLE CORR COEFF. SEE EQUATION (5)
CALCULATE SSDR. SEE EQUATION (7)
CALCULATE VARIANCE OF ESTIMATE. SEE EQUATION (9)
CALCULATE STD. DEV. OF REG. COEFF. SEE EQUATION (11)
80
80
CALCULATE T-VALUES. SEE EQUATION (12)
CALCULATE STD. ERROR OF ESTIMATE. SEE EQUATION (10)
CALCULATE F-VALUE. EQUATION (8)
STORE RESULTS OF CALCULATION IN ANS(1-10)
EXIT

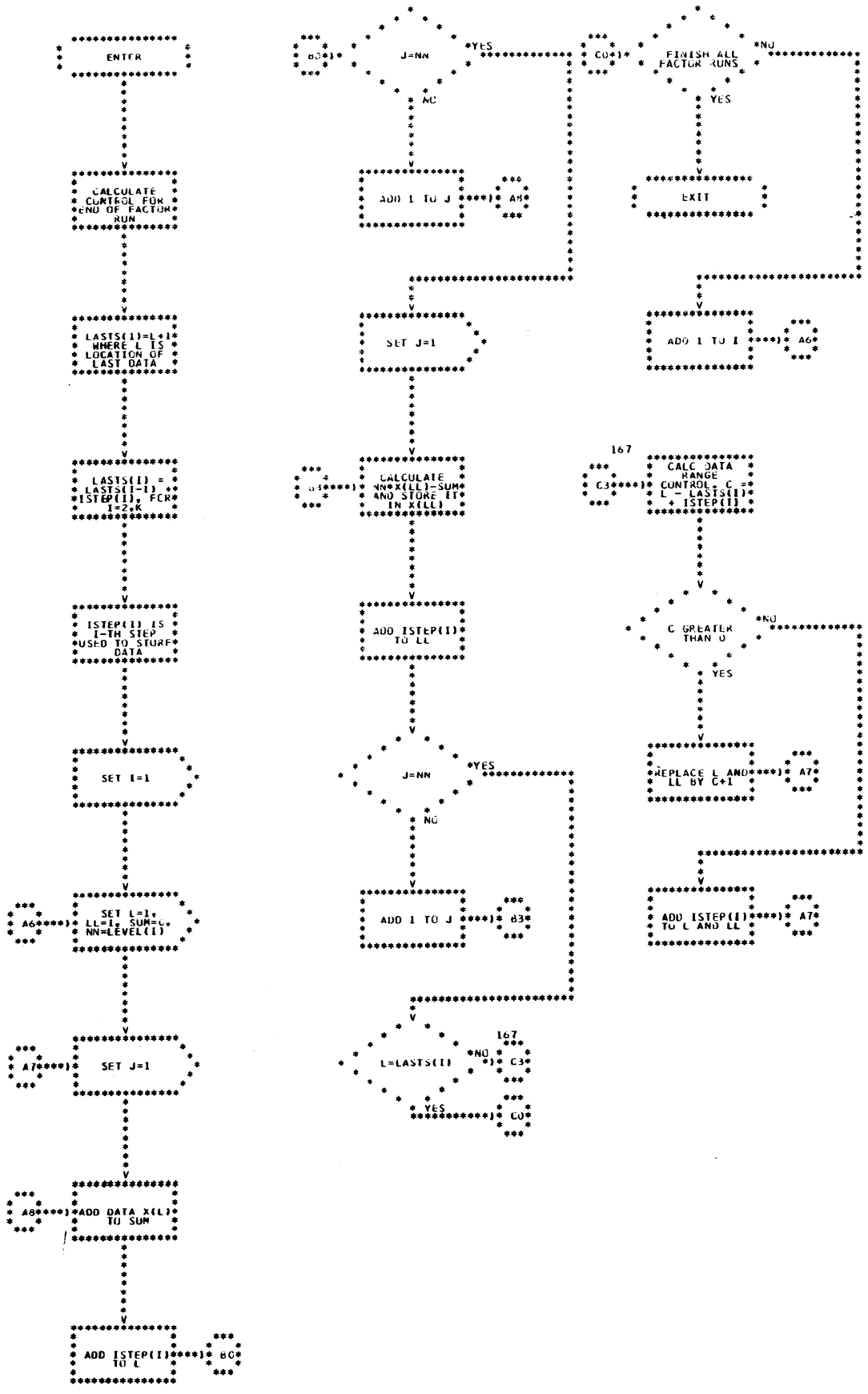

ENTER
GENERATE X VARIABLES. EQUATION (1)
CALCULATE MEANS. EQUATION (2)
CALC SUMS OF CROSS-PRODUCT OF DEVIATIONS EQ. (3)
CALCULATE CORRELATION COEFFICIENTS.
CALCULATE STANDARD DEVIATIONS. EQ. (5)
EXIT

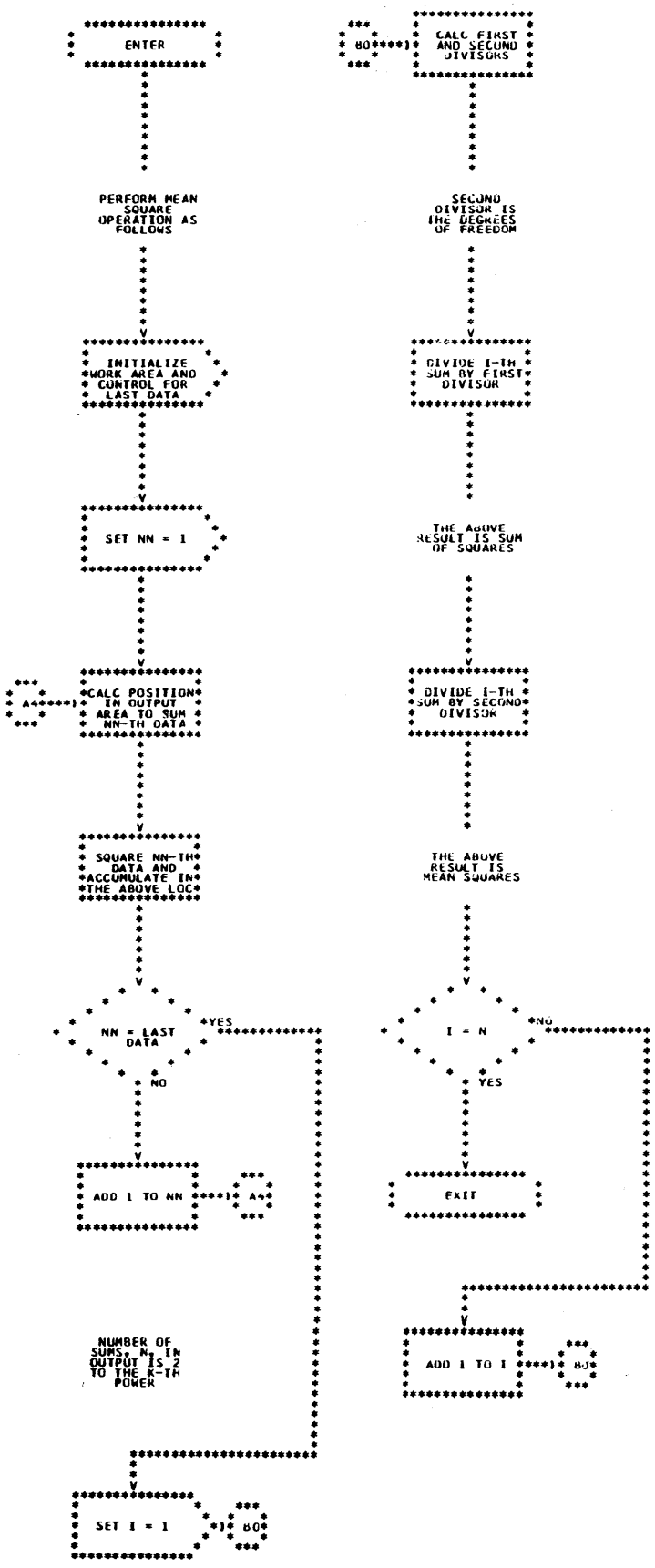

ENTER
INITIALIZE CONSTANT AND OUTPUT VECTOR, IER=0
FIND DEPENDENT VARIABLE, FORCED AND DELETED VARI.
FIND THE MAXIMUM NUMBER OF STEPS
ARE ALL FORCED VARIABLES ENTERED
YES
NO
A4
SELECT THE NEXT VARIABLE TO ENTER AMONG FORCED VARI.
SELECT THE NEXT VARIABLE TO ENTER AMONG NON-FORCED
IS SUM OF SQ. REDUCED GT. THAN 0
NO
YES
DOES REDUCTION EXCEED TOTAL SUM OF SQ.
NO
YES
B0
IER=1
C9
COMPUTE P, PROPORTION OF SUM OF SQ. REDUCED
B0
IS P GREATER THAN OR = INPUT CONSTANT
NO
YES
C9
STORE VARIABLE SELECTED AND SUM OF SQ. REDUCED
GET CUMULATIVE SUM OF SQ. REDUCED SEE EQ. (3)
GET CUMULATIVE PROPORTION REDUCED SEE EQ. (4)
COMPUTE MULTIPLE R SEE EQ. (5)
COMPUTE F VALUE FOR ANOVA SEE EQ. (6)
COMPUTE STD. ERROR OF ESTIMATE SEE EQ. (7)
INVERT DIAGONAL ELEMENT OF INPUT MATRIX SEE EQ. (8)
DIVIDE THE ROW BY THE PIVOT SEE EQ. (9)
C0
COMPUTE REG. COEFFICIENTS SEE EQ. (10)
C0
COMPUTE THE INTERCEPT SEE EQ. (11)
COMPUTE STD. ERRORS OF REG. COEFF. SEE EQ. (12)
COMPUTE T VALUES SEE EQ. (13)
PERFORM REDUCTION TO ELIMINATE VARIABLE EQ. (14-16)
COMPUTE ADJUSTED MULTIPLE CORRELATION
COMPUTE ADJUSTED STD. ERROR OF ESTIMATE
STOUT
CALL OUTPUT SUBROUTINE
WAS ANALYSIS TERMINATED IN STOUT
NO
YES
A4
C9
EXIT

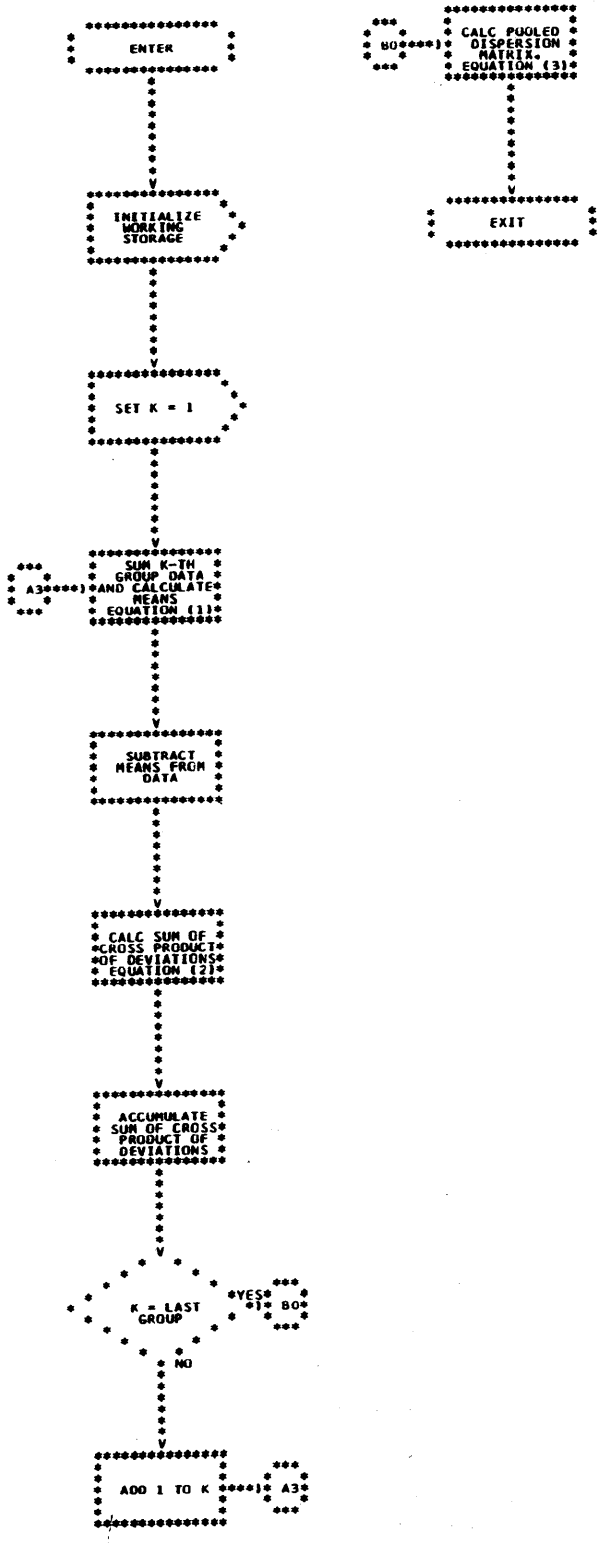

ENTER
SET ERROR CODE IER=0
IS NO. OF DOSE LEVELS GT 2
*YES
*NO
SET IER=1
5
80
*** D8 ***
IS THERE MINUS DOSE LEVEL
7
*NO
*YES
SET IER=2
80
*** D8 ***
IS LOG=1
*NO
*** B1 ***
*YES
IS ANY DOSAGE 0 OR MINUS
*YES
*NO
CONVERT STIMULUS LEVELS TO COMMON LOG
ARE SUBJECTS TESTED GT 0
*YES
*** B1 ***
*NO
*** B0 ***
*B0***
SET IER=3
80
*** D8 ***
B1
IS ANY OBJECTS RESPOND GT. *THAN TESTED*
*NO
*YES
IER=4
80
*** D8 ***
20
COMPUTE PROPORTIONS OF OBJECTS RESPONDING SEE EQ.(1)
22
COMPUTE PROBIT REG. COEFFICIENT SEE EQ. (2)
COMPUTE THE VALUE OF *INTERCEPT SEE* EQ. (3)
COMPUTE EXPECTED VALUES OF PROBIT SEE EQ. (4)
SET DD=0
33
*B8***
SET I=1
*COMPUTE I-TH WEIGHTING COEFF. SEE EQ. (5)
*** C0 ***
*C0***
*COMPUTE I-TH WORKING PROBIT SEE EQ. (6-7)
SUM *INTERMEDIATE RESULTS
IS I=LAST STIMULUS LEVEL
*YES
*NO
ADD 1 TO I
33
*** B8 ***
COMPUTE NEW ESTIMATE OF PROBIT REG. COEFF. EQ. (11)
COMPUTE NEW ESTIMATE OF *INTERCEPT SEE* EQ. (12)
SET I=1
*C7***
COMPUTE NEW VALUE OF EXPECTED PROBIT SEE EQ. (13)
FIND D, DIFFERENCE BETWEEN OLD AND NEW PROBIT
SUM SQUARED D IN SXX SEE EQ. (14)
*** D0 ***
D0
IS I=LAST STIMULUS LEVEL
*YES
*NO
ADD 1 TO I
*** C7 ***
COMPUTE AD, SUM OF ABSOLUTE DIFFERENCES OF DD-SXX
IS AD LESS THAN (1.0E-7)
*YES
*NO
STORE SXX IN DD
33
*** B8 ***
65
STORE THE *INTERCEPT AND* REG. COEFF.
COMPUTE CHI SQUARE SEE EQ. (15)
*FIND DEGREES OF FREEDOM FOR CHI SQUARE SEE EQ. (16)
80
D8
EXIT

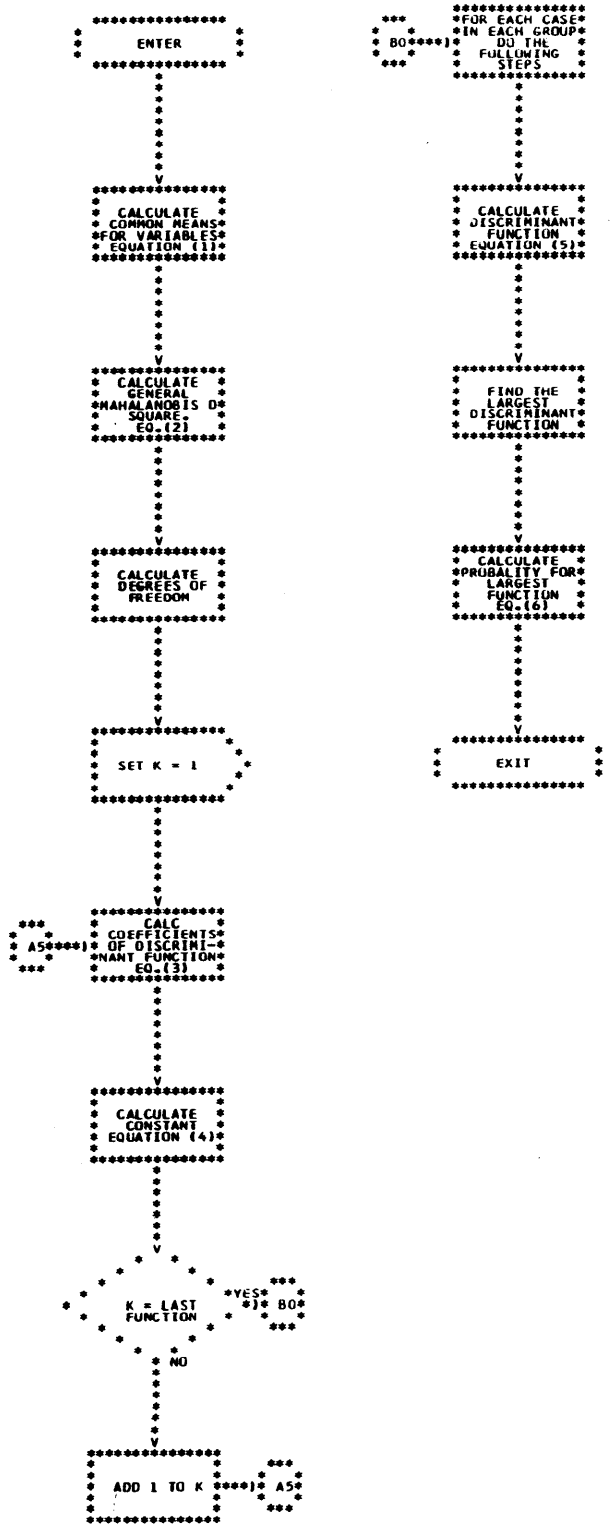

ENTER
PARTITION CORRELATION MATRIX EQUATION (1)
CALCULATE EIGENVALUES, EIGENVECTORS EQUATION (2)
*NROOT EIGENVALUES EIGENVECTORS
SET I = 1
A5
CALCULATE CANONICAL CORRELATION EQUATION (5)
CALCULATE CHI-SQUARE EQUATION (6)
CALC DEGREES OF FREEDOM FOR CHI-SQUARE EQUATION (7)
CALCULATE RIGHT HAND COEFFICIENTS EQUATION (8)
CALCULATE LEFT HAND COEFFICIENTS EQUATION (9)
A0
A0
I=O (J IS NUMBER OF CRITERIA)
*NO
YES
EXIT
ADD 1 TO I
A5

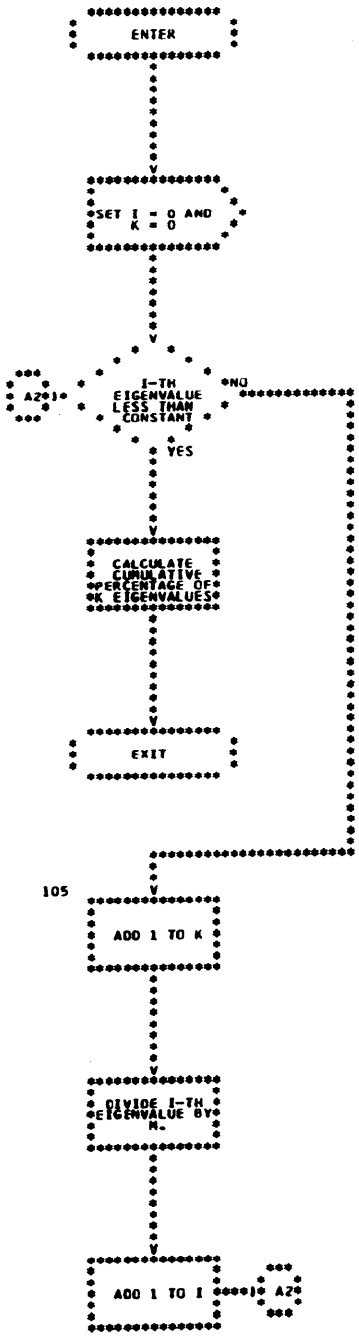

ENTER
CALC SIZE OF DATA ARRAY X. SEE EQUATION (1)
MOVE DATA TO UPPER PART OF ARRAY X
CALC STEPS USED TO STORE DATA. SEE EQUATION (2-3)
INITIALIZE KOUNT(J)=1 FOR DATA SUBSCRIPTS FOR J=1,K
SET I=1
CALC LOCATION (L) TO STORE A DATA. SEE EQUATION (4)
REDISTRIBUTE I-TH DATA POINT TO X(L)
SUBTRACT 1 FROM N
N=0
NO
YES
B1
B0
EXIT
SET J=1
KOUNT(J) = LEVEL(J)
YES
NO
ADD 1 TO KOUNT(J)
J=K
YES
NO
ADD 1 TO J
B2
SET KOUNT(J)=1
B4
ADD 1 TO I
A6
A6
B4
B2
B1
B0

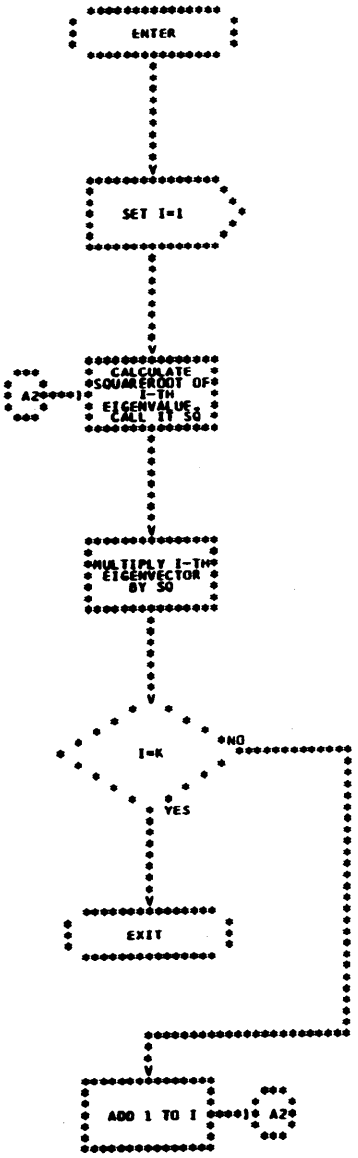

ENTER
CALCULATE CONTROL FOR END OF FACTOR RUN
LASTS(1)=L+1 WHERE L IS LOCATION OF LAST DATA
LASTS(I) = LASTS(I-1) + ISTEP(I), FOR I=2,K
ISTEP(I) IS I-TH STEP USED TO STORE DATA
SET I=1
SET L=1, LL=1, SUM=0, NN=LEVEL(1)
SET J=1
ADD DATA X(L) TO SUM
ADD ISTEP(I) TO L
B0
A6
A7
A8
J=NN
YES
NO
ADD 1 TO J
A4
B3
CALCULATE NN*X(LL)=SUM AND STORE IT IN X(LL)
ADD ISTEP(I) TO LL
L=LASTS(I)
167
C3
C0
FINISH ALL FACTOR RUNS
EXIT
ADD 1 TO I
A6
CALC DATA RANGE CONTROL, C = L - LASTS(I) + ISTEP(I)
C GREATER THAN 0
REPLACE L AND LL BY C+1
A7
ADD ISTEP(I) TO L AND LL
A7

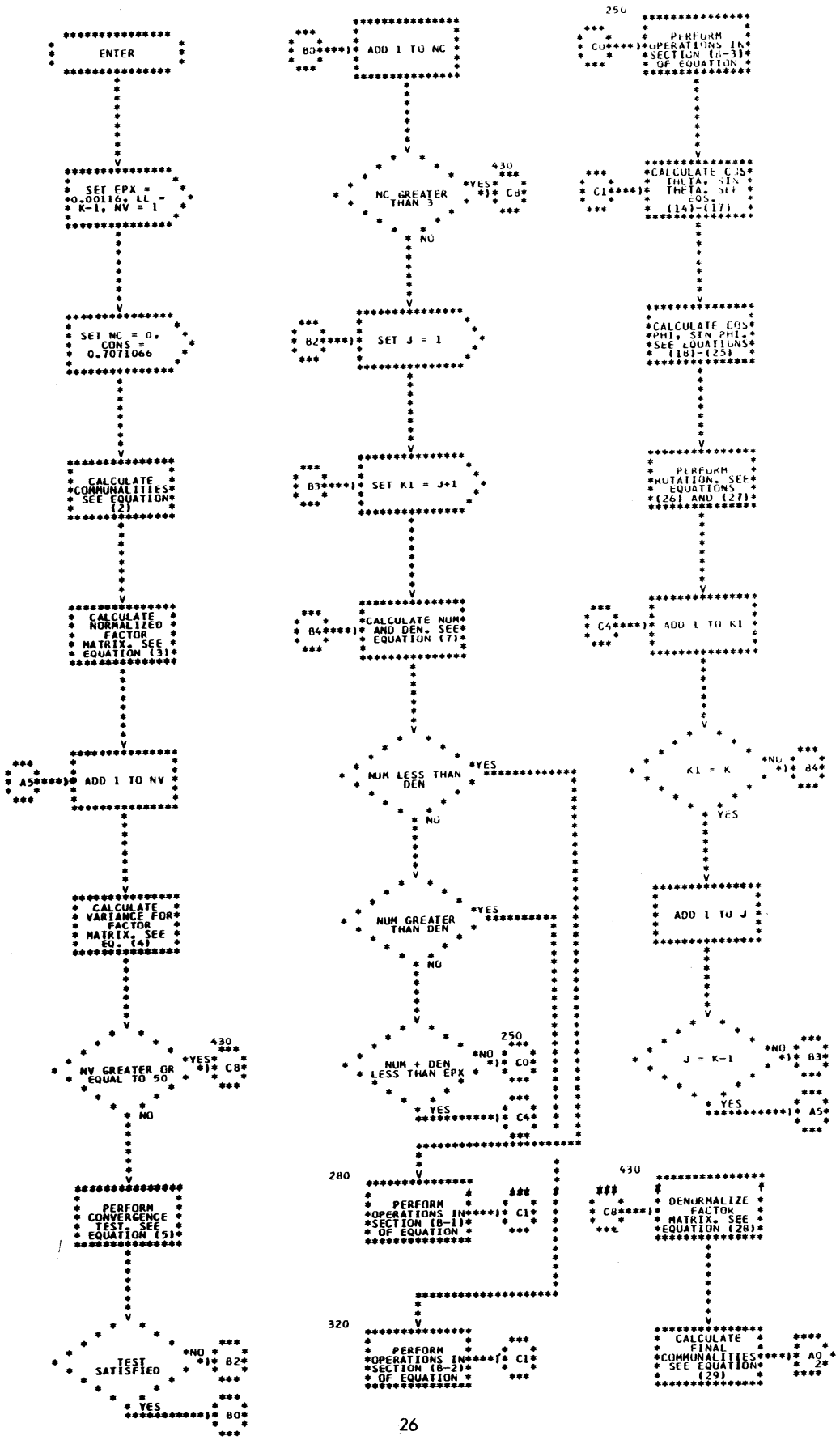

ENTER
PERFORM MEAN SQUARE OPERATION AS FOLLOWS
INITIALIZE WORK AREA AND CONTROL FOR LAST DATA
SET NN = 1
CALC POSITION IN OUTPUT AREA TO SUM NN-TH DATA
A4
SQUARE NN-TH DATA AND ACCUMULATE IN THE ABOVE LOC
NN = LAST DATA
YES
NO
ADD 1 TO NN
A4
NUMBER OF SUMS, N, IN OUTPUT IS 2 TO THE K-TH POWER
SET I = 1
B0
B0
CALC FIRST AND SECOND DIVISORS
SECOND DIVISOR IS THE DEGREES OF FREEDOM
DIVIDE I-TH SUM BY FIRST DIVISOR
THE ABOVE RESULT IS SUM OF SQUARES
DIVIDE I-TH SUM BY SECOND DIVISOR
THE ABOVE RESULT IS MEAN SQUARES
I = N
NO
YES
EXIT
ADD 1 TO I
B0

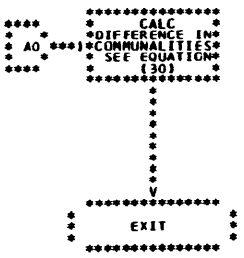

ENTER
INITIALIZE WORKING STORAGE
SET K = 1
A3
SUM K-TH GROUP DATA AND CALCULATE MEANS EQUATION (1)
SUBTRACT MEANS FROM DATA
CALC SUM OF CROSS PRODUCT OF DEVIATIONS EQUATION (2)
ACCUMULATE SUM OF CROSS PRODUCT OF DEVIATIONS
K = LAST GROUP
YES
1
B0
NO
ADD 1 TO K
A3
B0
CALC POOLED DISPERSION MATRIX. EQUATION (3)
EXIT

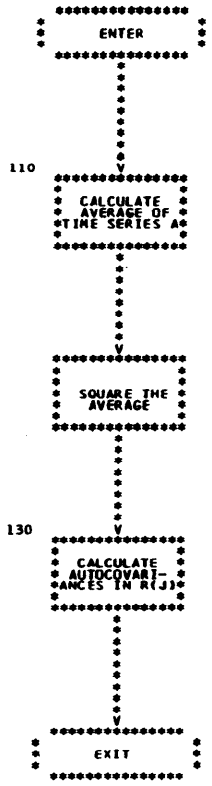

ENTER
CALCULATE COMMON MEANS FOR VARIABLES EQUATION (1)
CALCULATE GENERAL MAHALANOBIS D SQUARE. EQ.(2)
CALCULATE DEGREES OF FREEDOM
SET K = 1
CALC COEFFICIENTS OF DISCRIMINANT FUNCTION EQ.(3)
CALCULATE CONSTANT EQUATION (4)
K = LAST FUNCTION
YES
NO
ADD 1 TO K
A5
B0
FOR EACH CASE IN EACH GROUP DO THE FULLOWING STEPS
CALCULATE DISCRIMINANT FUNCTION EQUATION (5)
FIND THE LARGEST DISCRIMINANT FUNCTION
CALCULATE PROBALITY FOR LARGEST FUNCTION EQ.(6)
EXIT

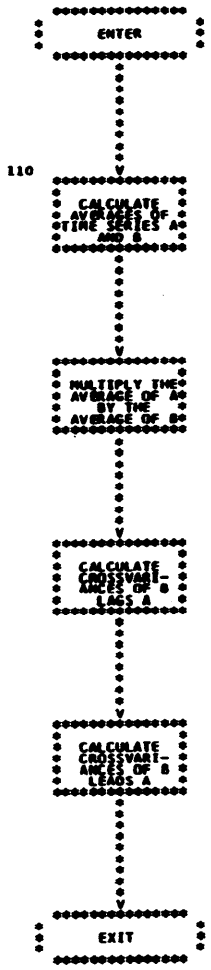

ENTER
SET I = 0 AND K = 0
A2
I-TH EIGENVALUE LESS THAN CONSTANT
NO
YES
CALCULATE CUMULATIVE PERCENTAGE OF K EIGENVALUES
EXIT
105
ADD 1 TO K
DIVIDE I-TH EIGENVALUE BY M.
ADD 1 TO I
A2

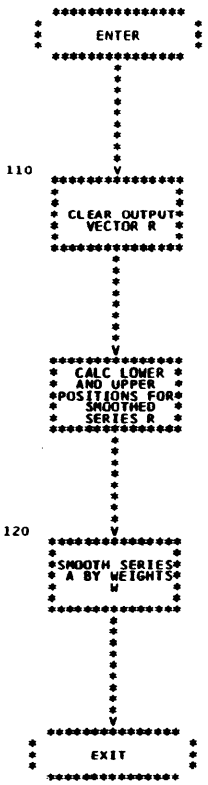

ENTER
SET I=1
CALCULATE SQUAREROOT OF I-TH EIGENVALUE CALL IT SO
A2
MULTIPLY I-TH EIGENVECTOR BY SO
I=K
NO
YES
EXIT
ADD 1 TO I
A2

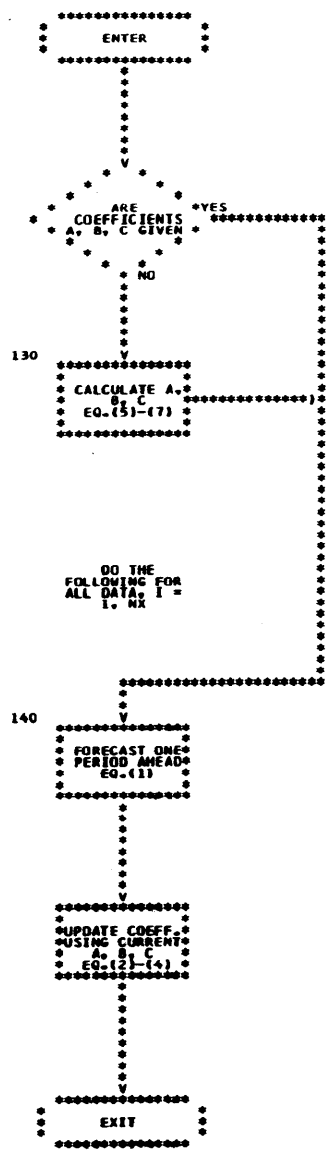

ENTER
SET EPX = 0.00116, LL = K-1, NV = 1
SET NC = 0, CONS = 0.7071066
CALCULATE COMMUNALITIES SEE EQUATION (2)
CALCULATE NORMALIZED FACTOR MATRIX, SEE EQUATION (3)
A5
ADD 1 TO NV
CALCULATE VARIANCE FOR FACTOR MATRIX, SEE EQ. (4)
NV GREATER OR EQUAL TO 50
YES
430
C8
NO
PERFORM CONVERGENCE TEST, SEE EQUATION (5)
TEST SATISFIED
NO
B2
YES
B0
B0
ADD 1 TO NC
NC GREATER THAN 3
YES
430
C8
NO
B2
SET J = 1
B3
SET K1 = J+1
B4
CALCULATE NUM AND DEN, SEE EQUATION (7)
NUM LESS THAN DEN
YES
NO
NUM GREATER THAN DEN
YES
NO
NUM + DEN LESS THAN EPX
NO
250
C0
YES
C4
280
PERFORM OPERATIONS IN SECTION (8-1) OF EQUATION
C1
320
PERFORM OPERATIONS IN SECTION (8-2) OF EQUATION
C1
C0
250
PERFORM OPERATIONS IN SECTION (8-3) OF EQUATION
C1
CALCULATE COS THETA, SIN THETA, SEE EQS. (14)-(17)
CALCULATE COS PHI, SIN PHI, SEE EQUATIONS (18)-(25)
PERFORM ROTATION, SEE EQUATIONS (26) AND (27)
C4
ADD 1 TO K1
K1 = K
NO
B4
YES
ADD 1 TO J
J = K-1
NO
B3
YES
A5
C8
430
DENURMALIZE FACTOR MATRIX, SEE EQUATION (28)
CALCULATE FINAL COMMUNALITIES SEE EQUATION (29)
A0
2

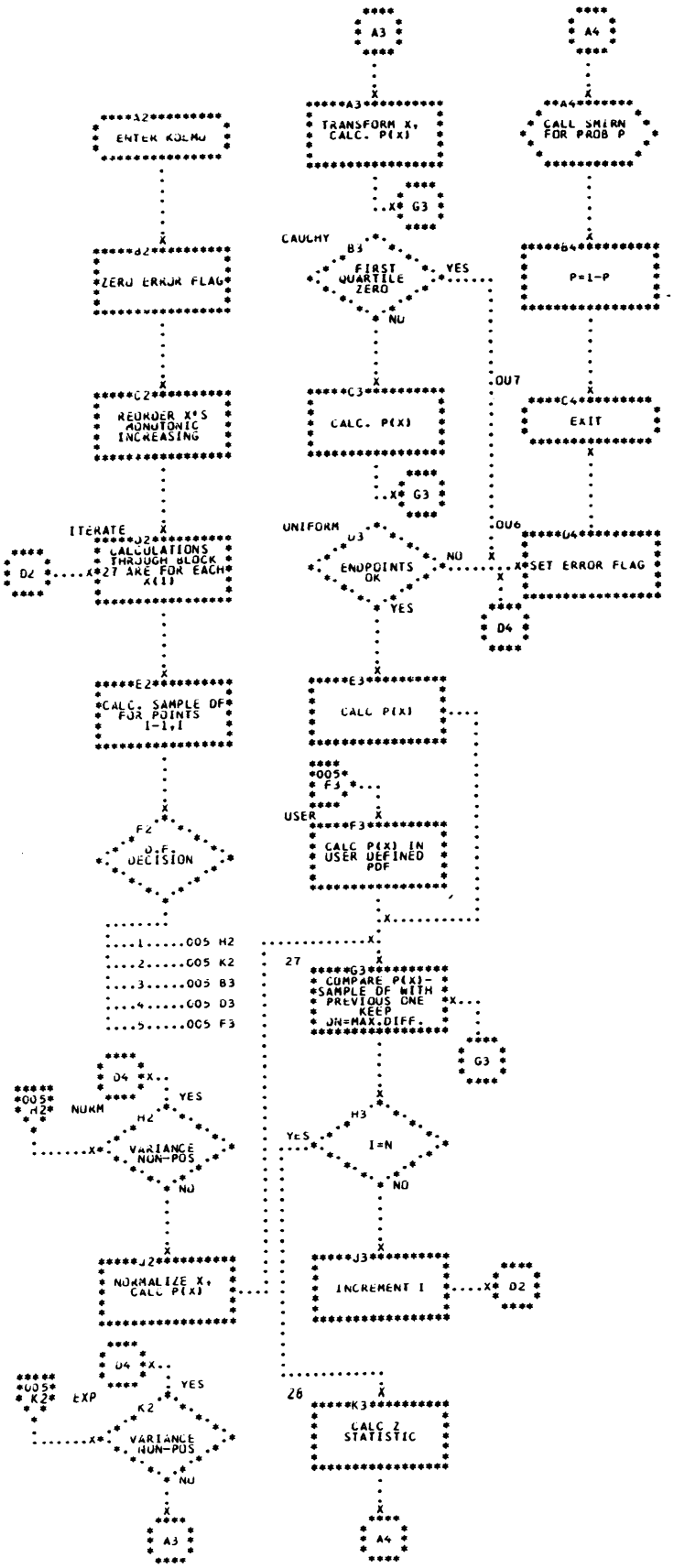

CALC
DIFFERENCE IN
COMMUNALITIES
SEE EQUATION
(30)
AO
EXIT

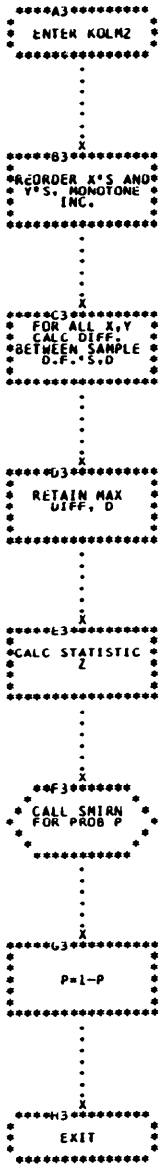

ENTER
110
CALCULATE AVERAGE OF TIME SERIES A
SQUARE THE AVERAGE
130
CALCULATE AUTOCOVARI- ANCES IN R(J)
EXIT

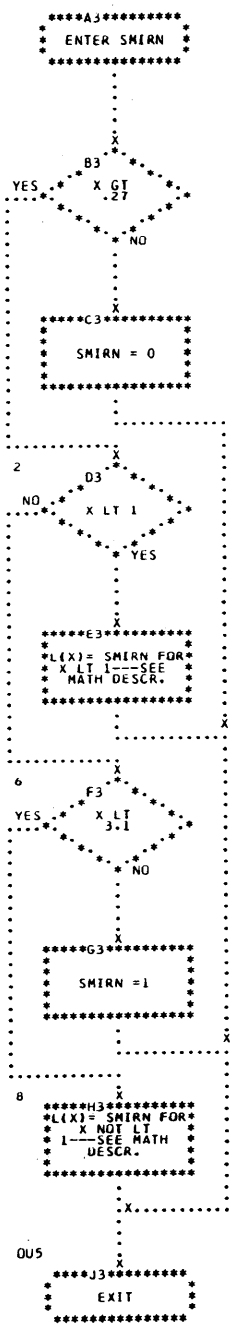

ENTER
110
CALCULATE AVERAGES OF TIME SERIES A AND B
MULTIPLY THE AVERAGE OF A BY THE AVERAGE OF B
CALCULATE CROSSVARI-ANCES OF B LAGS A
CALCULATE CROSSVARI-ANCES OF B LEADS A
EXIT

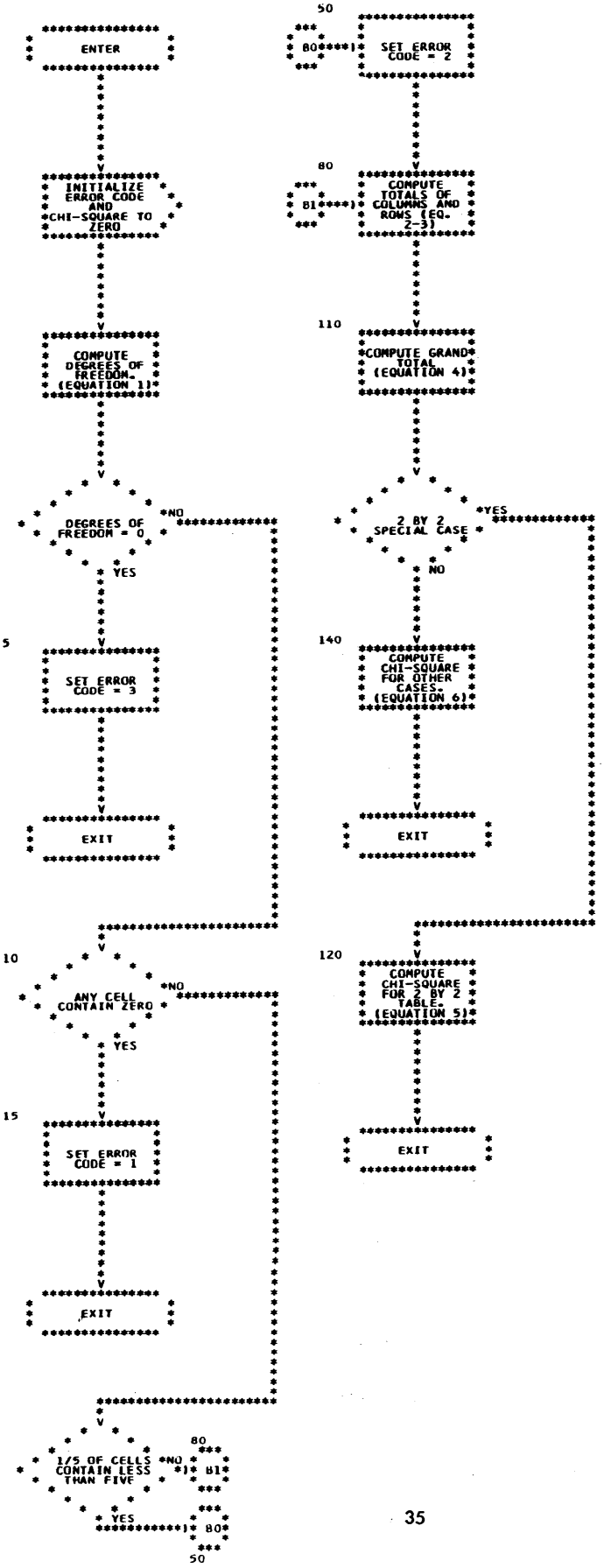

ENTER
110
CLEAR OUTPUT VECTOR R
CALC LOWER AND UPPER POSITIONS FOR SMOOTHED SERIES R
120
SMOOTH SERIES A BY WEIGHTS W
EXIT

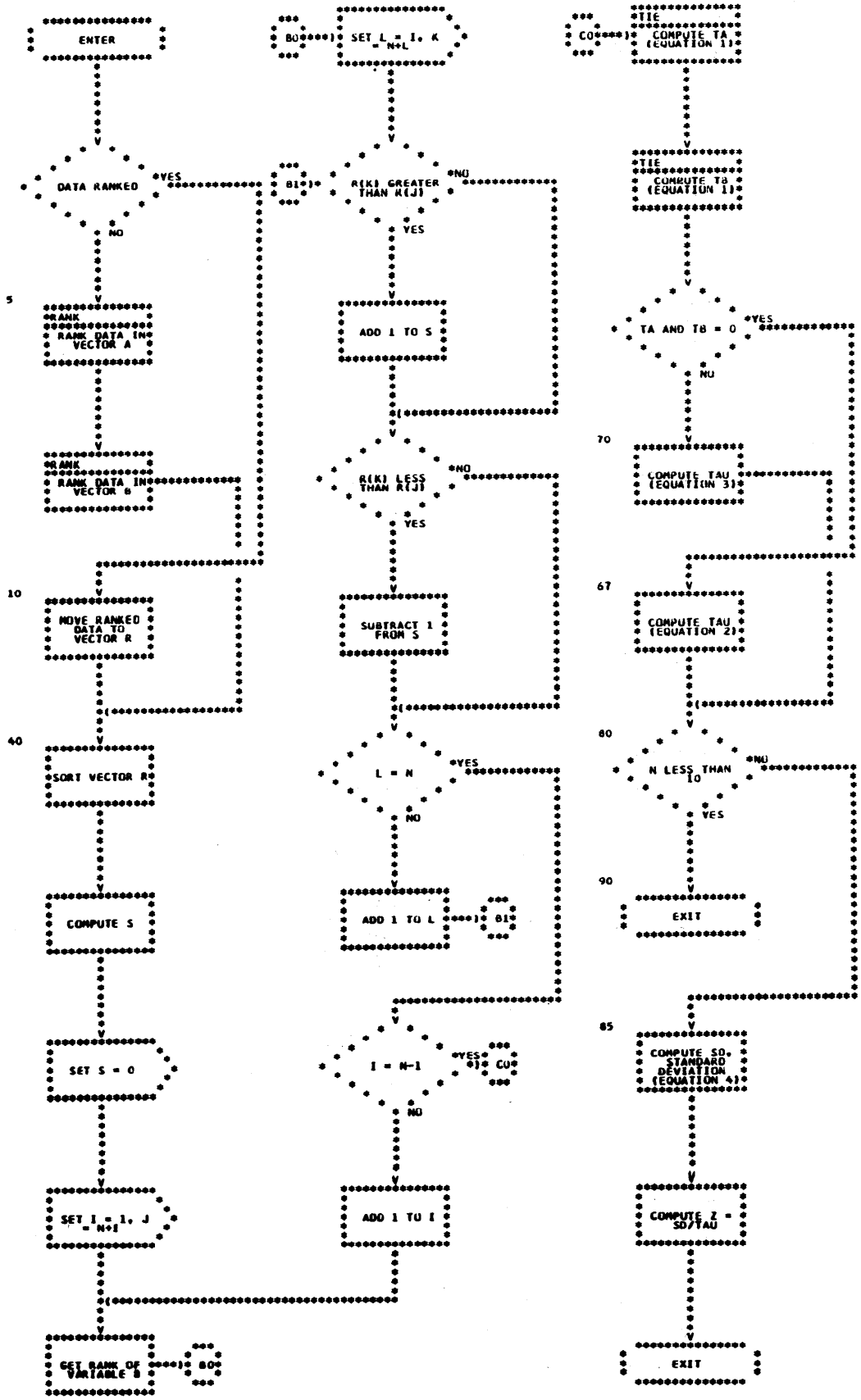

ENTER
ARE COEFFICIENTS A, B, C GIVEN
YES
NO
130
CALCULATE A, B, C EQ.(5)-(7)
DO THE FOLLOWING FOR ALL DATA, I = 1, NX
140
FORECAST ONE PERIOD AHEAD EQ.(1)
UPDATE COEFF. USING CURRENT A, B, C EQ.(2)-(4)
EXIT

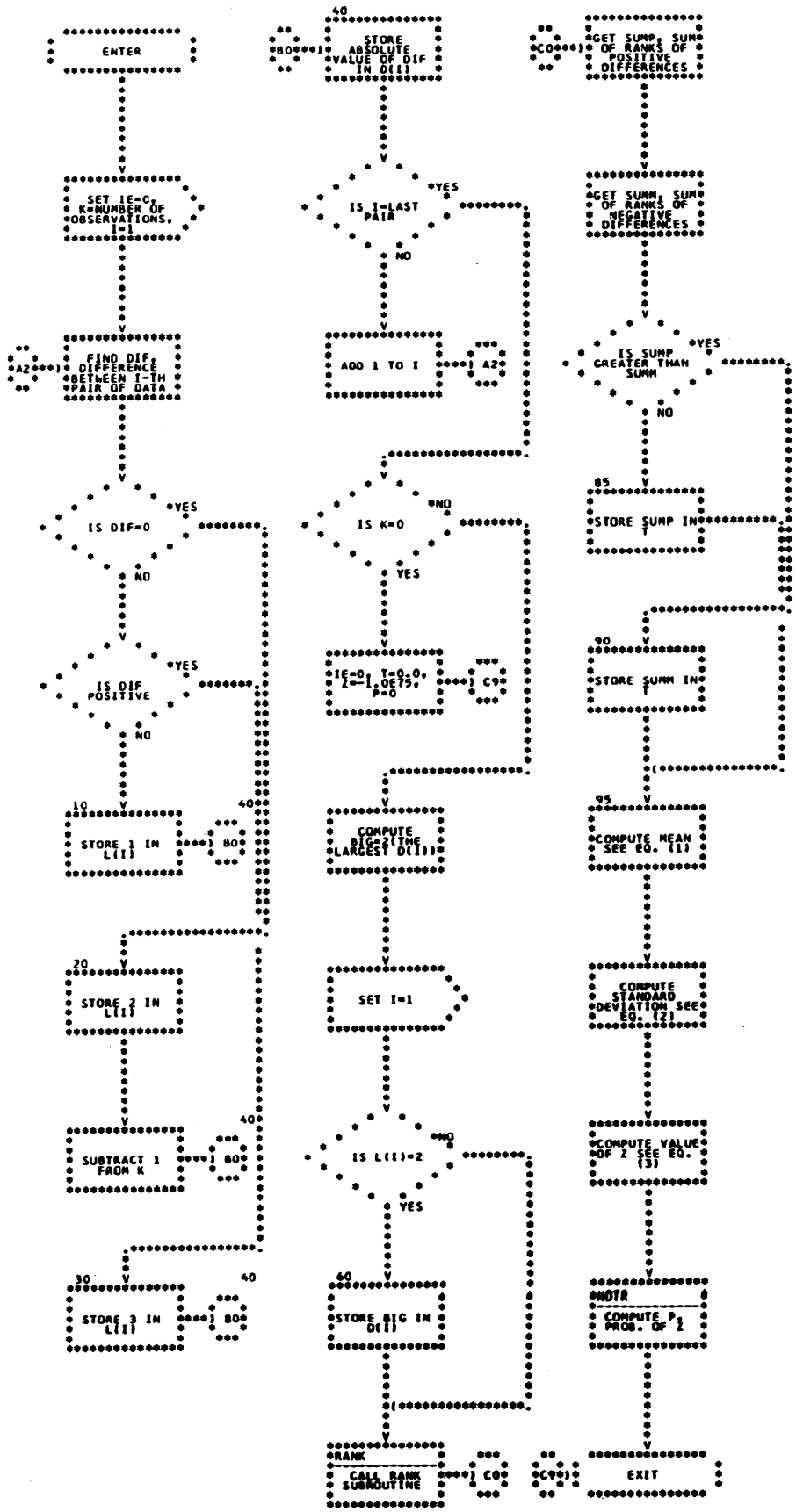

A3
A4
A2
ENTER KOLMO
A3
TRANSFORM X, CALC. P(X)
A4
CALL SMIRN FOR PROB P
B2
ZERO ERROR FLAG
G3
CAUCHY
B3
FIRST QUARTILE ZERO
YES
NO
B4
P=1-P
OUT
C2
REORDER X'S MONOTONIC INCREASING
C3
CALC. P(X)
C4
EXIT
ITERATE
J2
CALCULATIONS THROUGH BLOCK 27 ARE FOR EACH X(I)
D2
UNIFORM
D3
ENDPOINTS OK
NO
YES
OUT
D4
SET ERROR FLAG
D4
E2
CALC. SAMPLE DF FOR POINTS I-1,I
E3
CALC P(X)
005 F3
USER
F2
D.F. DECISION
F3
CALC P(X) IN USER DEFINED PDF
1.....005 H2
2.....005 K2
3.....005 B3
4.....005 D3
5.....005 F3
27
G3
COMPARE P(X)-SAMPLE DF WITH PREVIOUS ONE KEEP DN=MAX.DIFF.
G3
D4
005 H2
NUKM
YES
H2
VARIANCE NON-POS
NO
H3
I=N
YES
NO
J2
NORMALIZE X, CALC P(X)
J3
INCREMENT I
D2
D4
005 K2
EXP
YES
K2
VARIANCE NON-POS
NO
26
K3
CALC Z STATISTIC
A3
A4

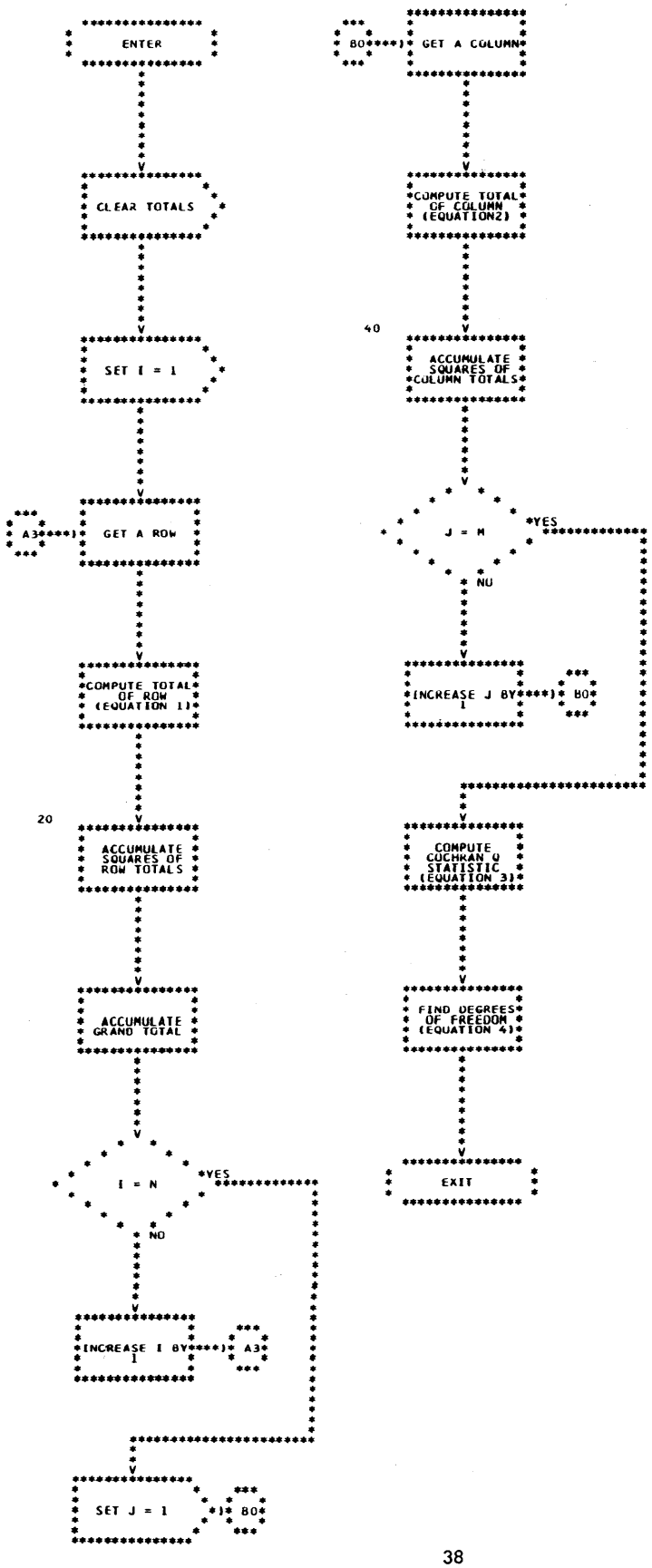

A3
ENTER KOLM2
B3
REORDER X'S AND Y'S, MONOTONE INC.
C3
FOR ALL X,Y CALC DIFF. BETWEEN SAMPLE D.F.'S,D
D3
RETAIN MAX DIFF, D
E3
CALC STATISTIC Z
F3
CALL SMIRN FOR PROB P
G3
P=1-P
H3
EXIT

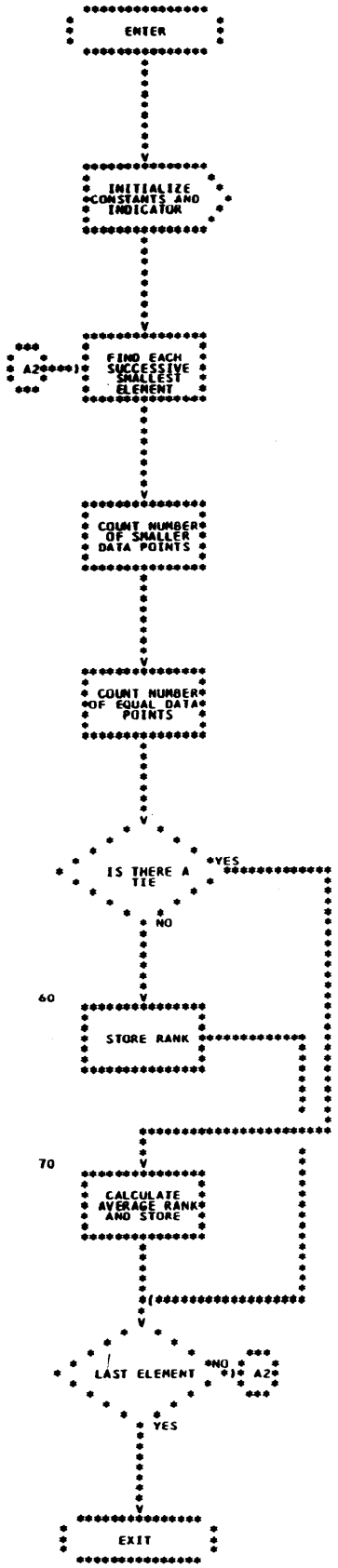

A3
ENTER SMIRN
B3
X GT .27
YES
NO
C3
SMIRN = 0
2
D3
X LT 1
NO
YES
E3
L(X)= SMIRN FOR X LT 1---SEE MATH DESCR.
6
F3
X LT 3.1
YES
NO
G3
SMIRN =1
8
H3
L(X)= SMIRN FOR X NOT LT 1---SEE MATH DESCR.
OU5
J3
EXIT

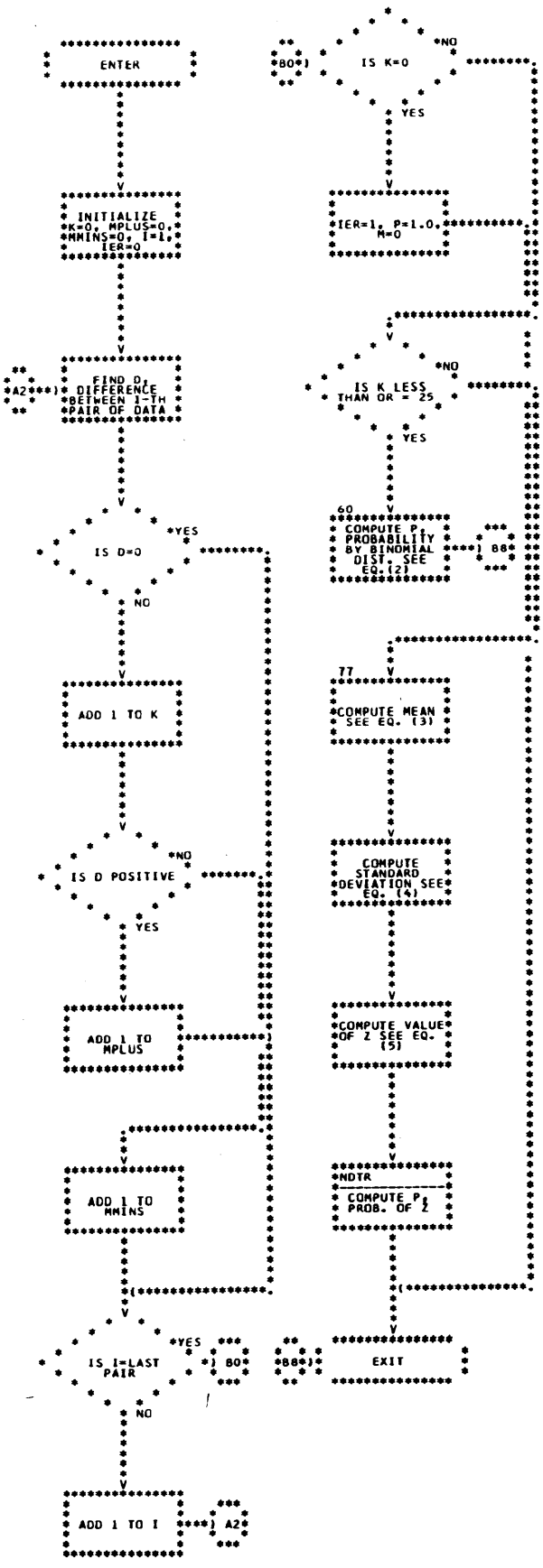

ENTER
INITIALIZE ERROR CODE AND CHI-SQUARE TO ZERO
COMPUTE DEGREES OF FREEDOM. (EQUATION 1)
DEGREES OF FREEDOM = 0
NO
YES
5
SET ERROR CODE = 3
EXIT
10
ANY CELL CONTAIN ZERO
NO
YES
15
SET ERROR CODE = 1
EXIT
80
1/5 OF CELLS CONTAIN LESS THAN FIVE
NO
YES
81
80
50
SET ERROR CODE = 2
80
COMPUTE TOTALS OF COLUMNS AND ROWS (EQ. 2-3)
110
COMPUTE GRAND TOTAL (EQUATION 4)
2 BY 2 SPECIAL CASE
YES
NO
140
COMPUTE CHI-SQUARE FOR OTHER CASES. (EQUATION 6)
EXIT
120
COMPUTE CHI-SQUARE FOR 2 BY 2 TABLE. (EQUATION 5)
EXIT

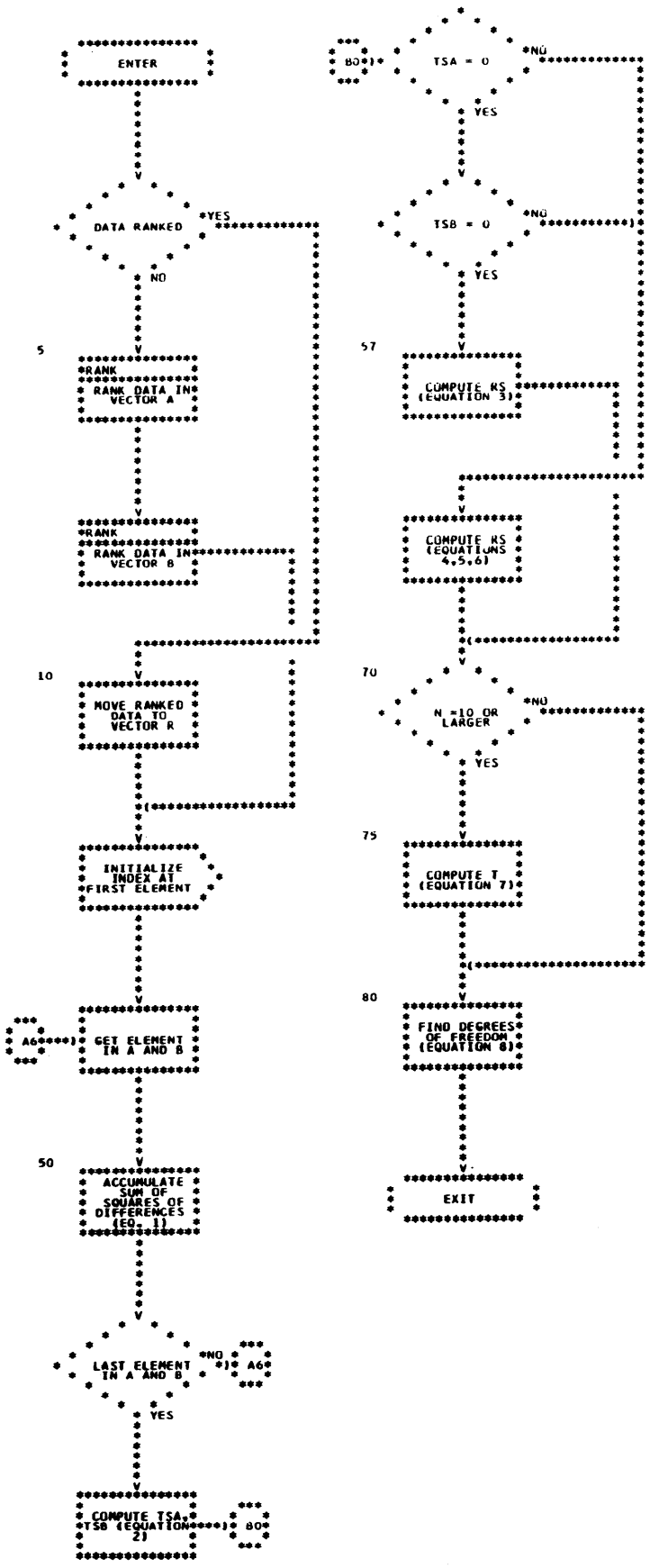

ENTER
DATA RANKED
YES
NO
5
RANK
RANK DATA IN VECTOR A
RANK
RANK DATA IN VECTOR B
10
MOVE RANKED DATA TO VECTOR R
40
SORT VECTOR R
COMPUTE S
SET S = 0
SET I = 1, J = N+1
GET RANK OF VARIABLE B
60
B0
SET L = I, K = N+L
B1
R(K) GREATER THAN R(J)
NO
YES
ADD 1 TO S
R(K) LESS THAN R(J)
NO
YES
SUBTRACT 1 FROM S
L = N
YES
NO
ADD 1 TO L
B1
I = N-1
YES
C0
NO
ADD 1 TO I
C0
TIE
COMPUTE TA (EQUATION 1)
TIE
COMPUTE TB (EQUATION 1)
TA AND TB = 0
YES
NO
70
COMPUTE TAU (EQUATION 3)
67
COMPUTE TAU (EQUATION 2)
80
N LESS THAN I0
NO
YES
90
EXIT
85
COMPUTE SD, STANDARD DEVIATION (EQUATION 4)
COMPUTE Z = SD/TAU
EXIT

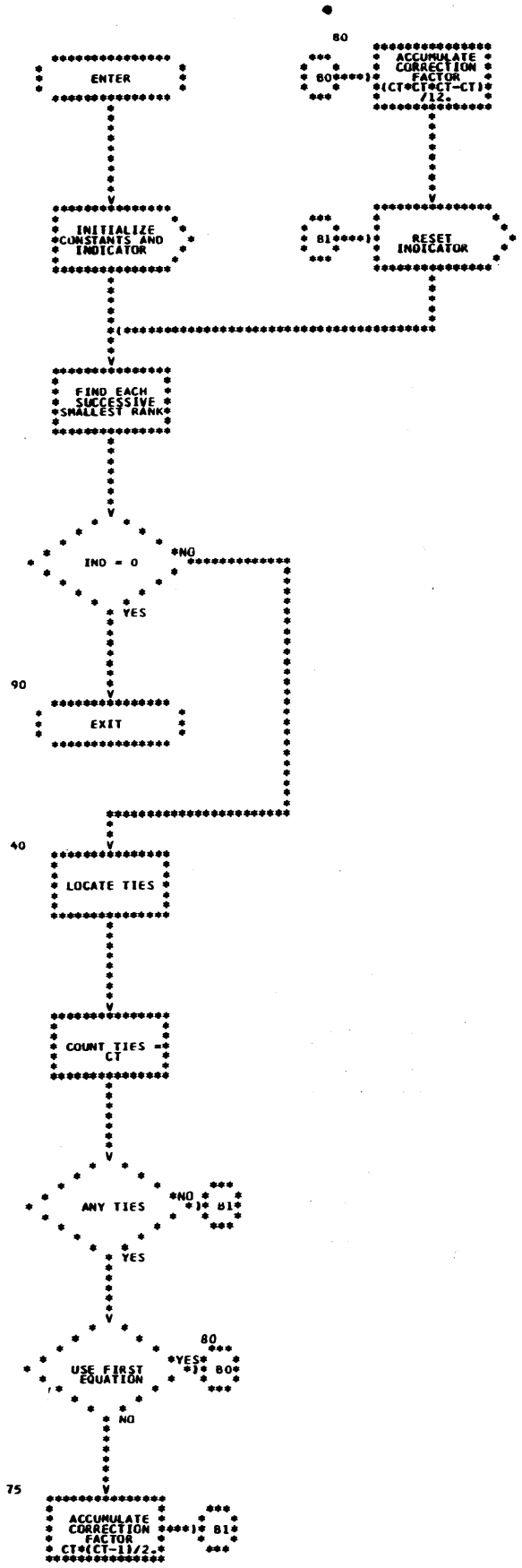

ENTER
SET IE=0, K=NUMBER OF OBSERVATIONS, I=1
FIND DIF, DIFFERENCE BETWEEN I-TH PAIR OF DATA
IS DIF=0
YES
NO
IS DIF POSITIVE
YES
NO
10
STORE 1 IN L(I)
40
B0
20
STORE 2 IN L(I)
SUBTRACT 1 FROM K
40
B0
30
STORE 3 IN L(I)
40
B0
40
STORE ABSOLUTE VALUE OF DIF IN D(I)
IS I=LAST PAIR
YES
NO
ADD 1 TO I
A2
IS K=0
NO
YES
IE=0, T=0.0, I=1,OETS, P=0
C9
COMPUTE BIG=2(THE LARGEST D(I))
SET I=1
IS L(I)=2
NO
YES
60
STORE BIG IN D(I)
RANK
CALL RANK SUBROUTINE
C0
C0
GET SUMP, SUM OF RANKS OF POSITIVE DIFFERENCES
GET SUMM, SUM OF RANKS OF NEGATIVE DIFFERENCES
IS SUMP GREATER THAN SUMM
YES
NO
85
STORE SUMP IN T
90
STORE SUMM IN T
95
COMPUTE MEAN SEE EQ. (1)
COMPUTE STANDARD DEVIATION SEE EQ. (2)
COMPUTE VALUE OF Z SEE EQ. (3)
NOTR
COMPUTE P1 PROB. OF Z
EXIT
C99

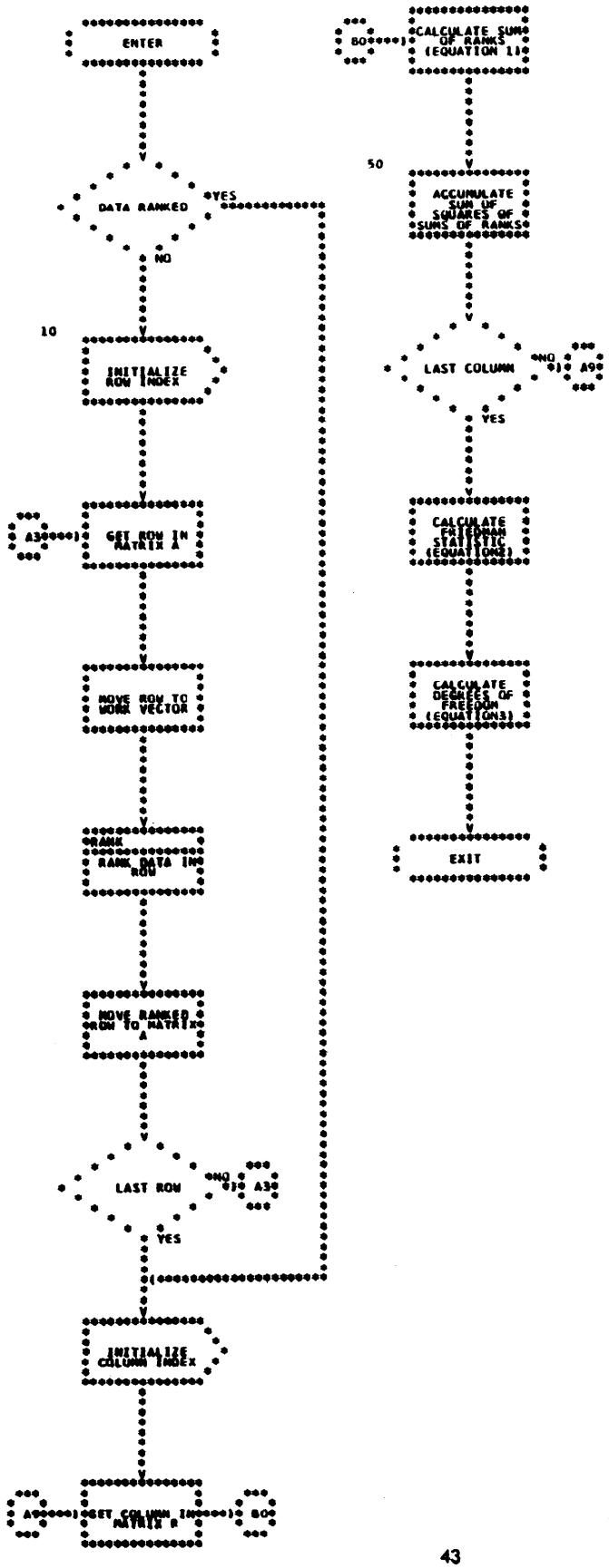

ENTER
CLEAR TOTALS
SET I = 1
A3
GET A ROW
COMPUTE TOTAL OF ROW (EQUATION 1)
20
ACCUMULATE SQUARES OF ROW TOTALS
ACCUMULATE GRAND TOTAL
I = N
YES
NO
INCREASE I BY 1
A3
SET J = 1
80
GET A COLUMN
COMPUTE TOTAL OF COLUMN (EQUATION2)
40
ACCUMULATE SQUARES OF COLUMN TOTALS
J = M
YES
NO
INCREASE J BY 1
80
COMPUTE COCHRAN Q STATISTIC (EQUATION 3)
FIND DEGREES OF FREEDOM (EQUATION 4)
EXIT

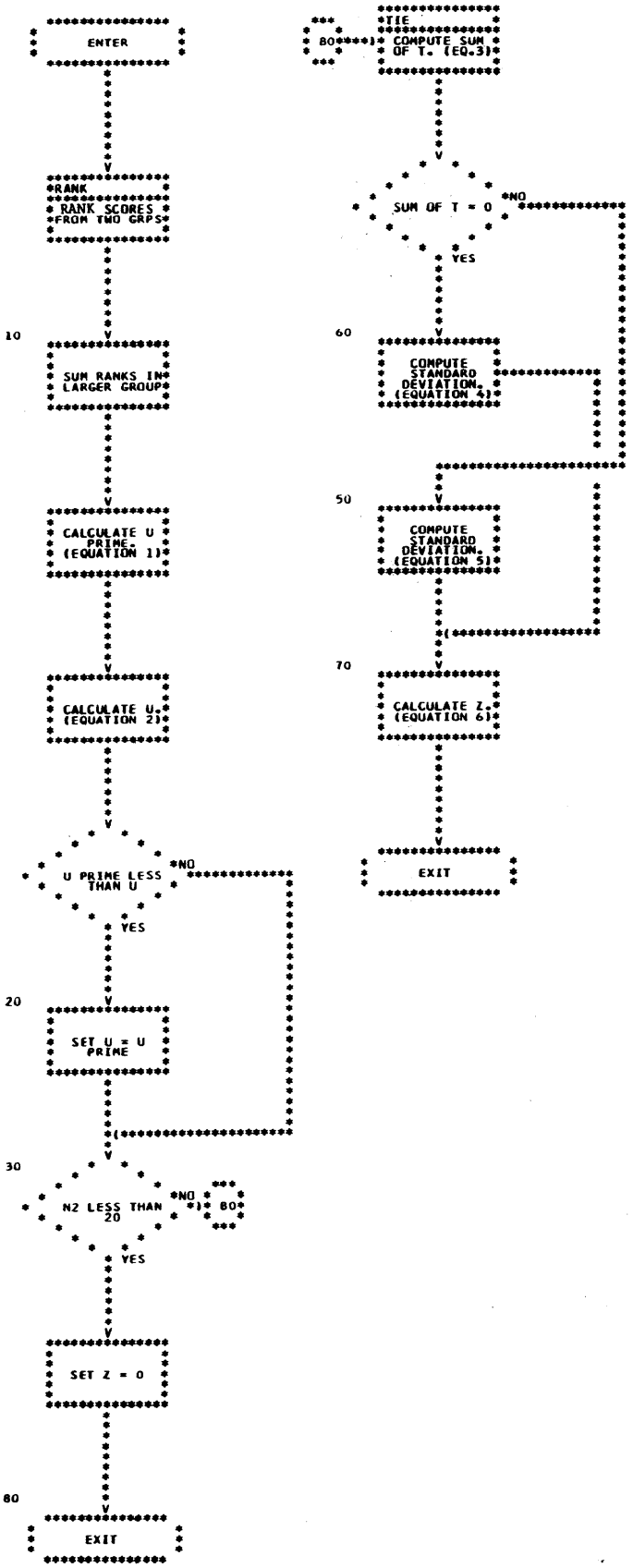

ENTER
INITIALIZE CONSTANTS AND INDICATOR
A2
FIND EACH SUCCESSIVE SMALLEST ELEMENT
COUNT NUMBER OF SMALLER DATA POINTS
COUNT NUMBER OF EQUAL DATA POINTS
IS THERE A TIE
YES
NO
60
STORE RANK
70
CALCULATE AVERAGE RANK AND STORE
LAST ELEMENT
NO
A2
YES
EXIT

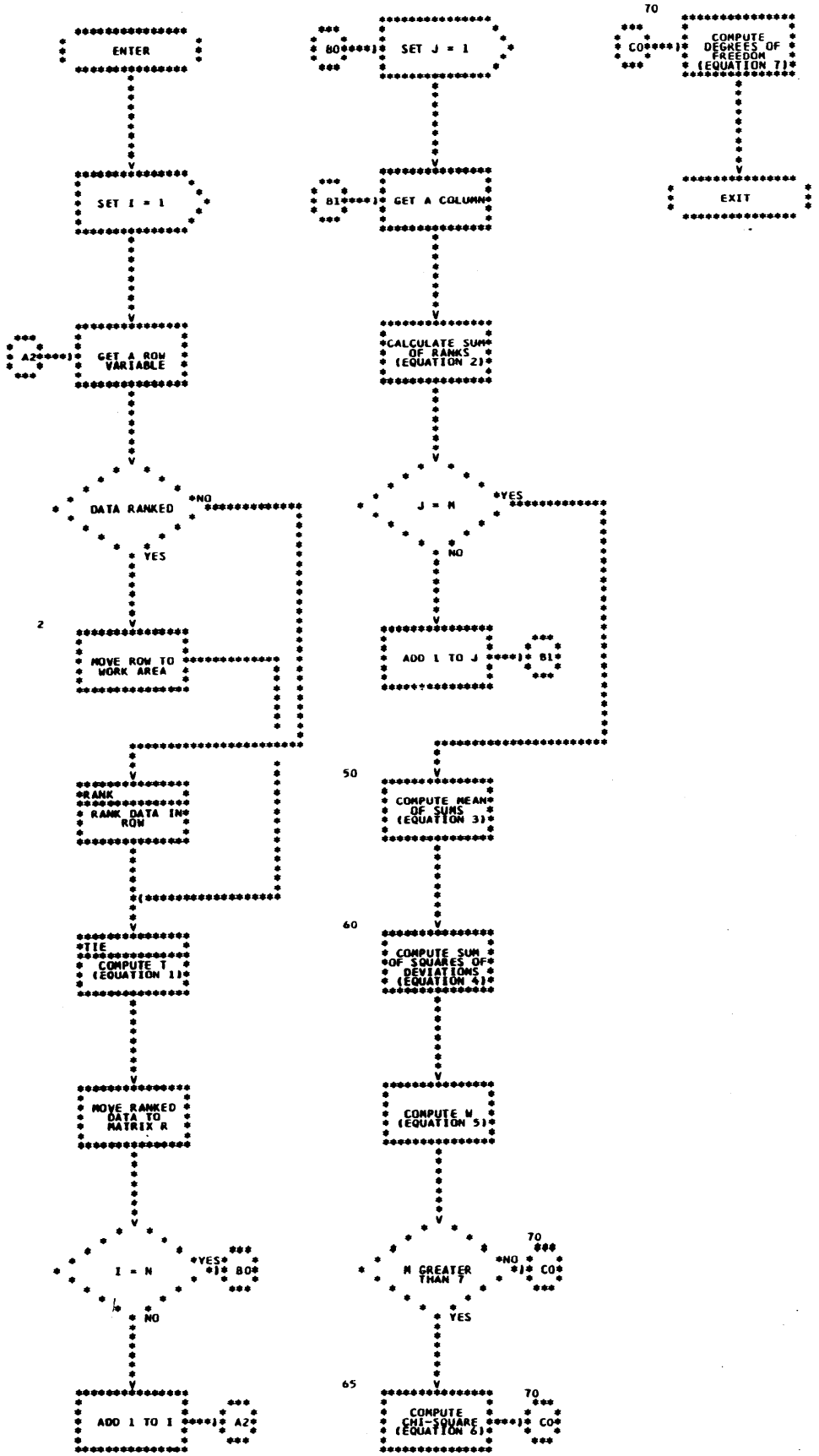

ENTER
INITIALIZE K=0, MPLUS=0, MMINS=0, I=1, IER=0
FIND D, DIFFERENCE BETWEEN I-TH PAIR OF DATA
A2
IS D=0
YES
NO
ADD 1 TO K
IS D POSITIVE
NO
YES
ADD 1 TO MPLUS
ADD 1 TO MMINS
IS I=LAST PAIR
YES
80
NO
ADD 1 TO I
A2
80
IS K=0
NO
YES
IER=1, P=1.0, M=0
IS K LESS THAN OR = 25
NO
YES
60
COMPUTE P, PROBABILITY BY BINOMIAL DIST. SEE EQ.(2)
88
77
COMPUTE MEAN SEE EQ. (3)
COMPUTE STANDARD DEVIATION SEE EQ. (4)
COMPUTE VALUE OF Z SEE EQ. (5)
NDTR
COMPUTE P, PROB. OF Z
88
EXIT


ENTER
DATA RANKED
YES
NO
5
RANK
RANK DATA IN VECTOR A
RANK
RANK DATA IN VECTOR B
10
MOVE RANKED DATA TO VECTOR R
INITIALIZE INDEX AT FIRST ELEMENT
A6
GET ELEMENT IN A AND B
50
ACCUMULATE SUM OF SQUARES OF DIFFERENCES (EQ. 1)
LAST ELEMENT IN A AND B
NO
YES
A6
COMPUTE TSA, TSB (EQUATION 2)
80
TSA = 0
NO
YES
TSB = 0
NO
YES
57
COMPUTE RS (EQUATION 3)
COMPUTE RS (EQUATIONS 4,5,6)
70
N =10 OR LARGER
NO
YES
75
COMPUTE T (EQUATION 7)
80
FIND DEGREES OF FREEDOM (EQUATION 8)
EXIT

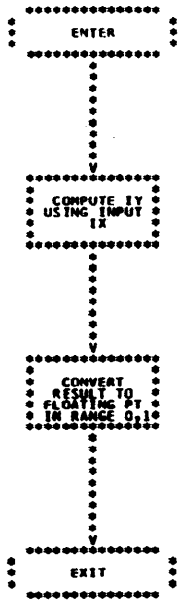

ENTER
INITIALIZE CONSTANTS AND INDICATOR
FIND EACH SUCCESSIVE SMALLEST RANK
IND = 0
NO
YES
90
EXIT
40
LOCATE TIES
COUNT TIES = CT
ANY TIES
NO
81
YES
80
USE FIRST EQUATION
YES
80
NO
75
ACCUMULATE CORRECTION FACTOR CT*(CT-1)/2.
81
RESET INDICATOR
80
ACCUMULATE CORRECTION FACTOR (CT*CT*CT-CT) /12.


ENTER
DATA RANKED
YES
NO
10
INITIALIZE ROW INDEX
A3
GET ROW IN MATRIX A
MOVE ROW TO WORK VECTOR
RANK
RANK DATA IN ROW
MOVE RANKED ROW TO MATRIX A
LAST ROW
NO
A3
YES
INITIALIZE COLUMN INDEX
A
GET COLUMN IN MATRIX B
B0
B0
CALCULATE SUM OF RANKS (EQUATION 1)
50
ACCUMULATE SUM OF SQUARES OF SUMS OF RANKS
LAST COLUMN
NO
A9
YES
CALCULATE FRIEDMAN STATISTIC (EQUATION2)
CALCULATE DEGREES OF FREEDOM (EQUATION3)
EXIT

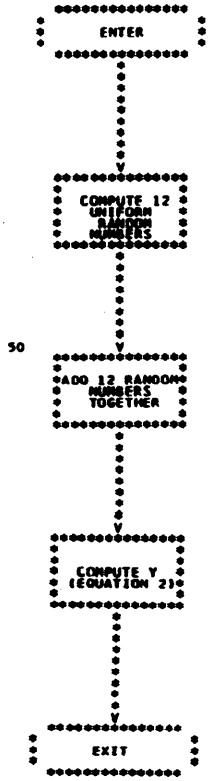

ENTER
RANK
RANK SCORES FROM TWO GRPS
10
SUM RANKS IN LARGER GROUP
CALCULATE U PRIME. (EQUATION 1)
CALCULATE U. (EQUATION 2)
U PRIME LESS THAN U
NO
YES
20
SET U = U PRIME
30
N2 LESS THAN 20
NO
80
YES
SET Z = 0
80
EXIT
TIE
80
COMPUTE SUM OF T. (EQ.3)
SUM OF T = 0
NO
YES
60
COMPUTE STANDARD DEVIATION. (EQUATION 4)
50
COMPUTE STANDARD DEVIATION. (EQUATION 5)
70
CALCULATE Z. (EQUATION 6)
EXIT


ENTER
SET I = 1
A2
GET A ROW VARIABLE
DATA RANKED
NO
YES
2
MOVE ROW TO WORK AREA
RANK
RANK DATA IN ROW
TIE
COMPUTE T (EQUATION 1)
MOVE RANKED DATA TO MATRIX R
I = N
YES
B0
NO
ADD 1 TO I
A2
B0
SET J = 1
B1
GET A COLUMN
CALCULATE SUM OF RANKS (EQUATION 2)
J = M
YES
NO
ADD 1 TO J
B1
50
COMPUTE MEAN OF SUMS (EQUATION 3)
60
COMPUTE SUM OF SQUARES OF DEVIATIONS (EQUATION 4)
COMPUTE W (EQUATION 5)
M GREATER THAN 7
70
NO
C0
YES
65
COMPUTE CHI-SQUARE (EQUATION 6)
70
C0
70
C0
COMPUTE DEGREES OF FREEDOM (EQUATION 7)
EXIT

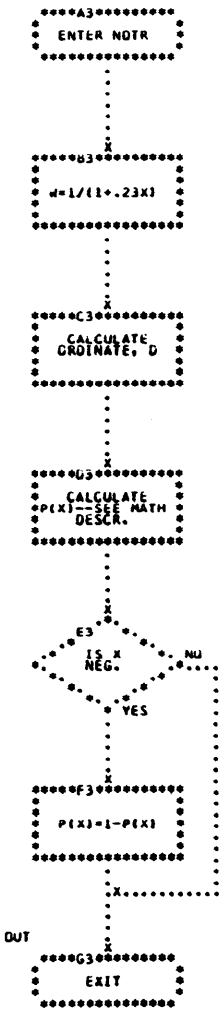
SUBROUTINE RANDU 1

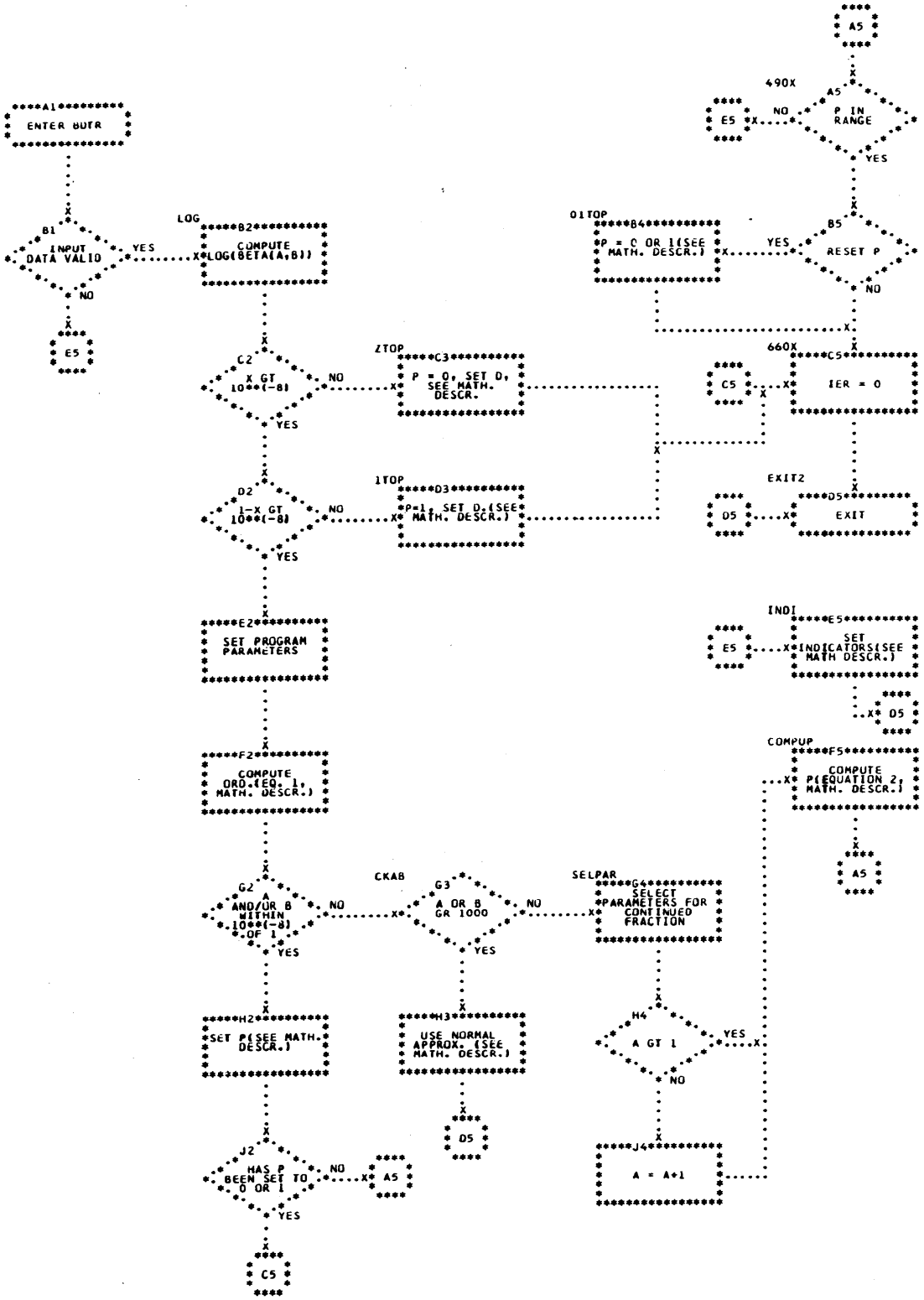


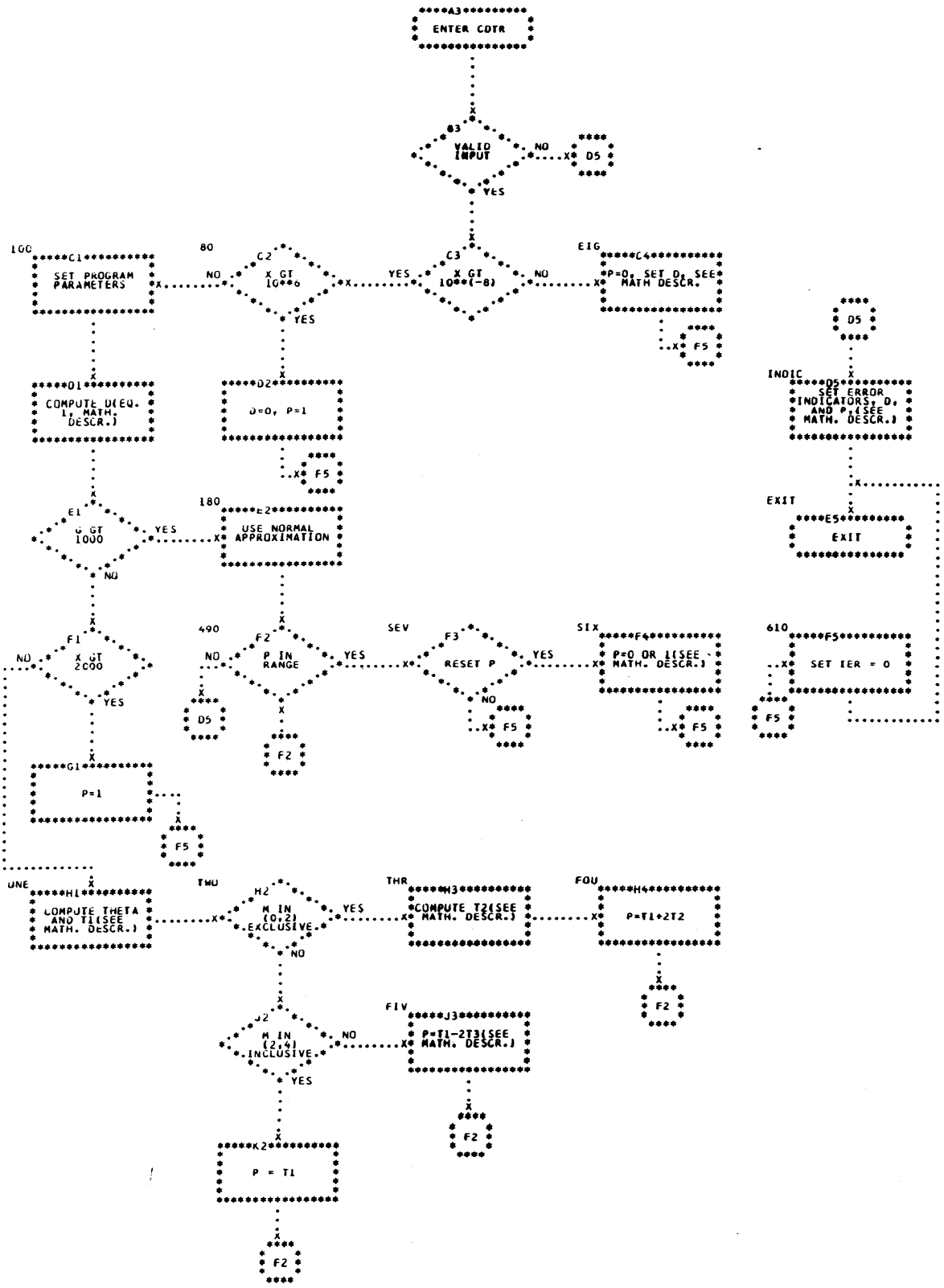
SUBROUTINE GAUSS 1



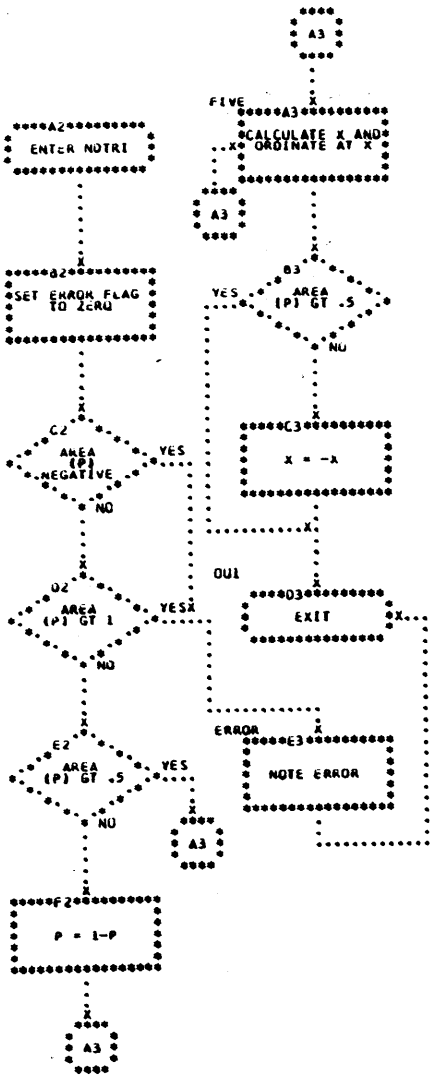
SUBROUTINE NOTR

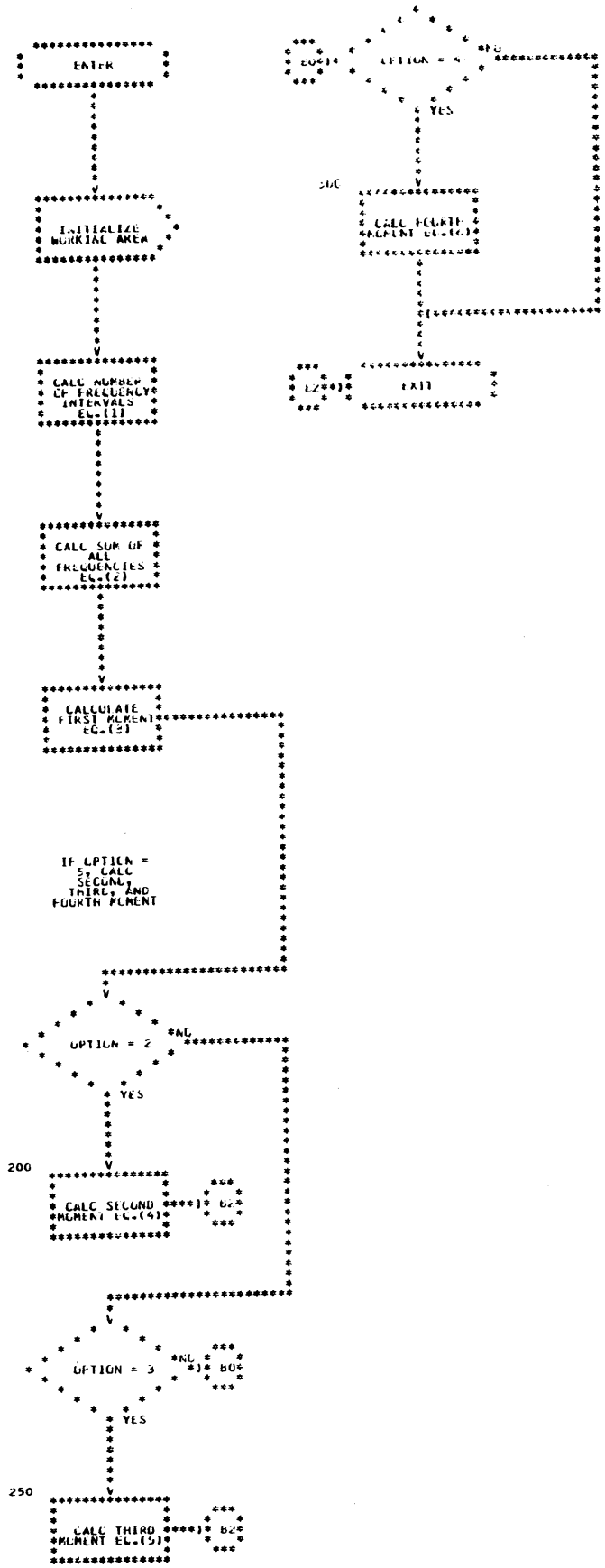


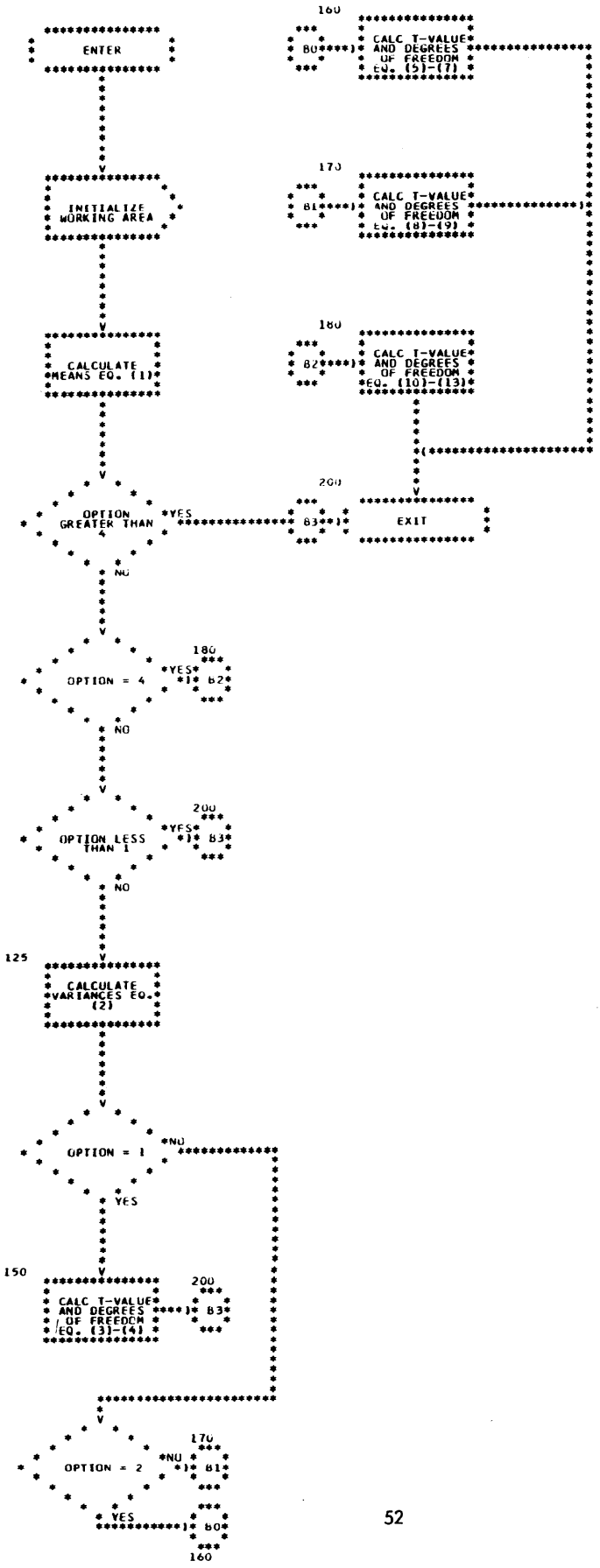

A1
ENTER BOTR
B1
INPUT DATA VALID
YES
NO
E5
LOG
B2
COMPUTE LOG(BETA(A,B))
C2
X GT 10**(-8)
NO
YES
ZTOP
C3
P = 0, SET D, SEE MATH. DESCR.
D2
1-X GT 10**(-8)
NO
YES
1TOP
D3
P=1, SET D, (SEE MATH. DESCR.)
E2
SET PROGRAM PARAMETERS
F2
COMPUTE ORD.(EQ. 1, MATH. DESCR.)
G2
A AND/OR B WITHIN 10**(-8) OF 1
NO
YES
CKAB
G3
A OR B GR 1000
NO
YES
H2
SET P(SEE MATH. DESCR.)
H3
USE NORMAL APPROX. (SEE MATH. DESCR.)
D5
J2
HAS P BEEN SET TO 0 OR 1
NO
YES
A5
C5
SELPAR
G4
SELECT PARAMETERS FOR CONTINUED FRACTION
H4
A GT 1
YES
NO
J4
A = A+1
A5
490X
E5
P IN RANGE
NO
YES
01TOP
B4
P = 0 OR 1(SEE MATH. DESCR.)
B5
RESET P
YES
NO
660X
C5
IER = 0
EXIT2
D5
EXIT
INDI
E5
SET INDICATORS(SEE MATH DESCR.)
D5
COMPUP
F5
COMPUTE P(EQUATION 2, MATH. DESCR.)
A5

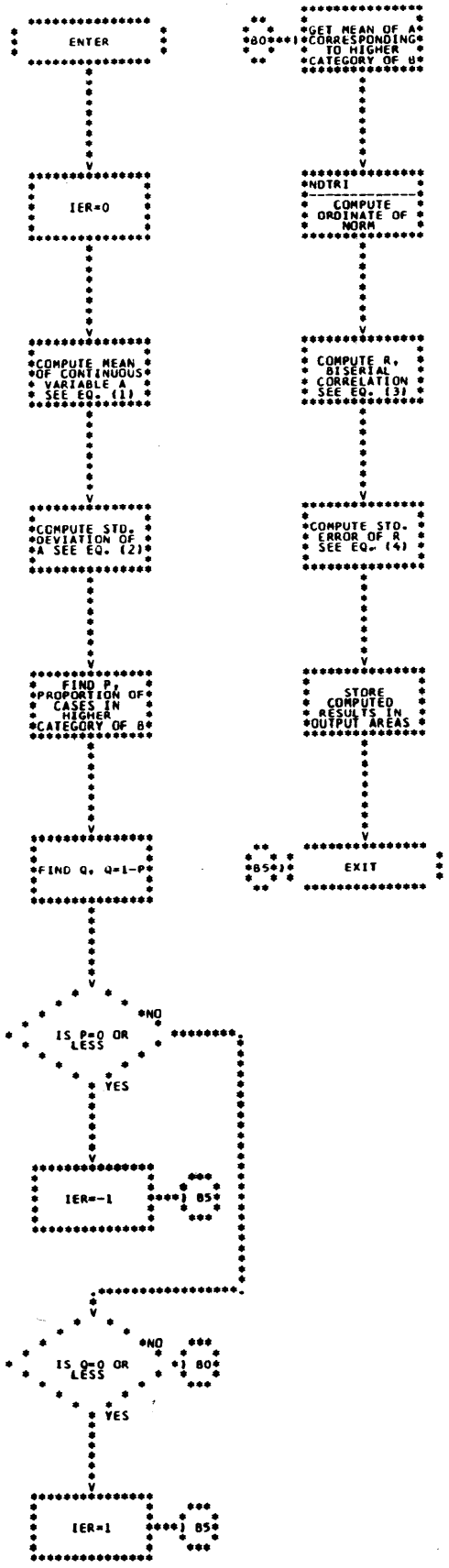

A3
ENTER CDTR
B3
VALID INPUT
NO
D5
YES
100
C1
SET PROGRAM PARAMETERS
80
C2
X GT 10**6
NO
YES
C3
X GT 10**(-8)
YES
NO
EIG
C4
P=0, SET D, SEE MATH DESCR.
F5
D1
COMPUTE D(EQ. 1, MATH. DESCR.)
D2
D=0, P=1
F5
E1
G GT 1000
YES
NO
180
E2
USE NORMAL APPROXIMATION
F1
X GT 2000
NO
YES
490
F2
P IN RANGE
NO
YES
D5
F2
SEV
F3
RESET P
YES
NO
F5
SIX
F6
P=0 OR 1(SEE MATH. DESCR.)
F5
D5
INDIC
D5
SET ERROR INDICATORS, D, AND P.(SEE MATH. DESCR.)
EXIT
E5
EXIT
610
F5
SET IER = 0
F5
G1
P=1
F5
ONE
H1
COMPUTE THETA AND T1(SEE MATH. DESCR.)
TWO
H2
M IN (0,2) EXCLUSIVE
YES
NO
THR
H3
COMPUTE T2(SEE MATH. DESCR.)
FOU
H4
P=T1+2T2
F2
J2
M IN (2,4) INCLUSIVE
NO
YES
FIV
J3
P=T1-2T3(SEE MATH. DESCR.)
F2
K2
P = T1
F2

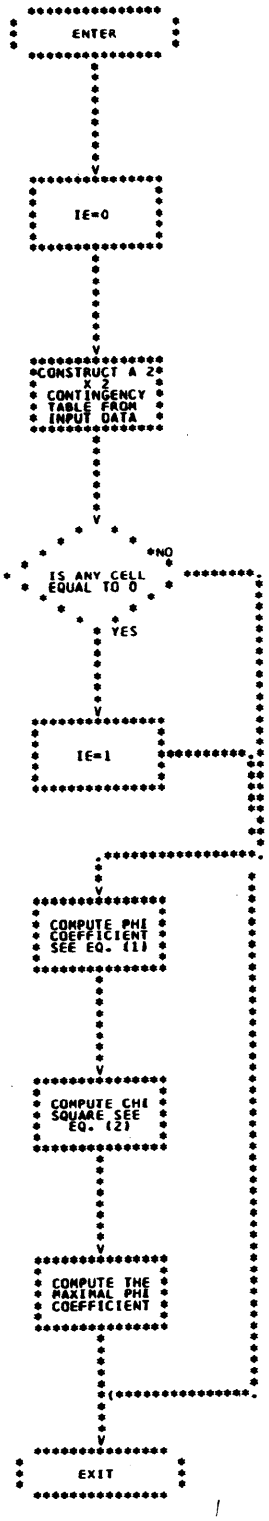
SUBROUTINE NDTRI

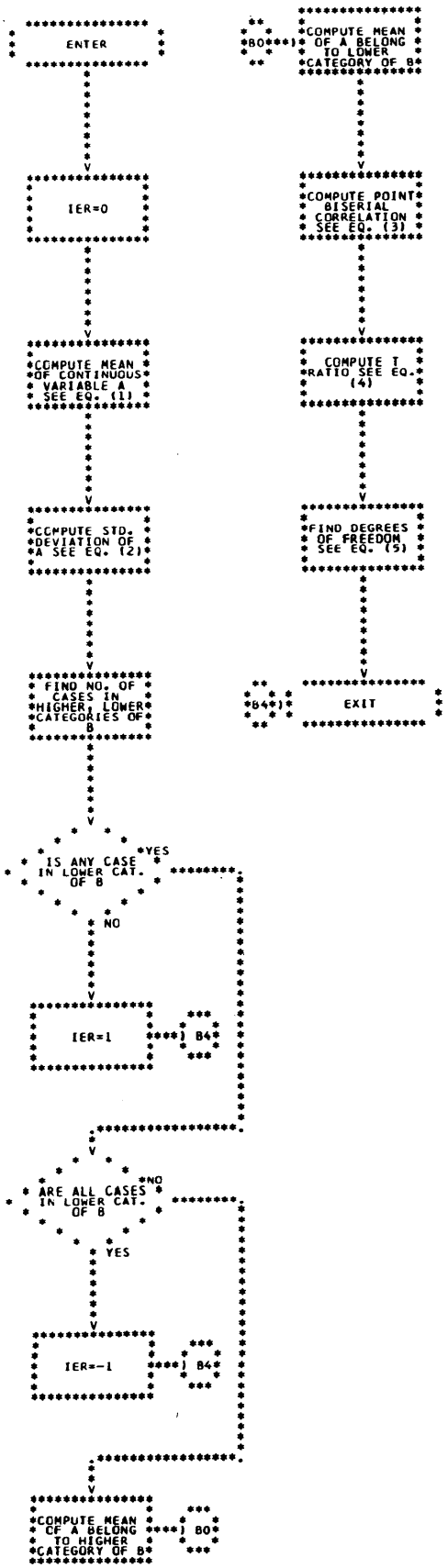


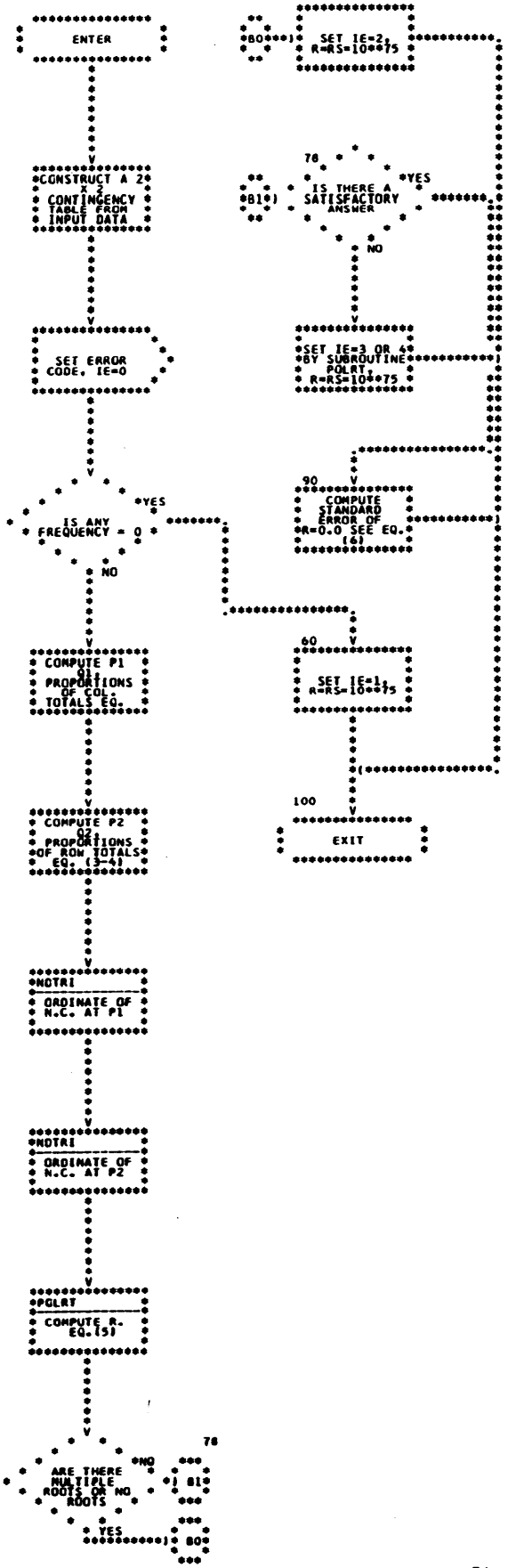

ENTER
INITIALIZE WORKING AREA
CALC NUMBER OF FREQUENCY INTERVALS EQ.(1)
CALC SUM OF ALL FREQUENCIES EQ.(2)
CALCULATE FIRST MOMENT EQ.(3)
IF OPTION = 5, CALC SECOND, THIRD, AND FOURTH MOMENT
OPTION = 2
NO
YES
200
CALC SECOND MOMENT EQ.(4)
OPTION = 3
NO
YES
250
CALC THIRD MOMENT EQ.(5)
OPTION = 4
NO
YES
300
CALC FOURTH MOMENT EQ.(6)
EXIT


ENTER
INITIALIZE WORKING AREA
CALCULATE MEANS EQ. (1)
OPTION GREATER THAN 4
YES
NO
OPTION = 4
YES
NO
OPTION LESS THAN 1
YES
NO
125
CALCULATE VARIANCES EQ. (2)
OPTION = 1
NO
YES
150
CALC T-VALUE AND DEGREES OF FREEDOM EQ. (3)-(4)
OPTION = 2
NO
YES
160
CALC T-VALUE AND DEGREES OF FREEDOM EQ. (5)-(7)
170
CALC T-VALUE AND DEGREES OF FREEDOM EQ. (8)-(9)
180
CALC T-VALUE AND DEGREES OF FREEDOM EQ. (10)-(13)
200
EXIT

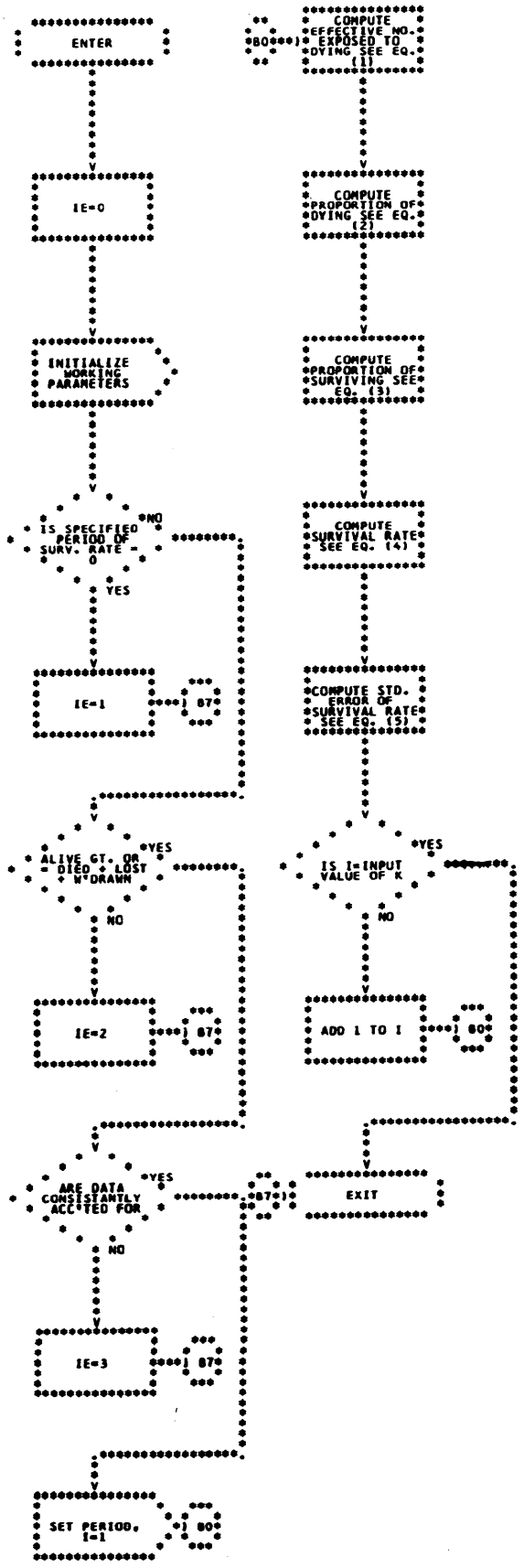

ENTER
IER=0
COMPUTE MEAN OF CONTINUOUS VARIABLE A SEE EQ. (1)
COMPUTE STD. DEVIATION OF A SEE EQ. (2)
FIND P, PROPORTION OF CASES IN HIGHER CATEGORY OF B
FIND Q, Q=1-P
IS P=0 OR LESS
NO
YES
IER=-1
85
IS Q=0 OR LESS
NO
80
YES
IER=1
85
80
GET MEAN OF A CORRESPONDING TO HIGHER CATEGORY OF B
NDTRI
COMPUTE ORDINATE OF NORM
COMPUTE R, BISERIAL CORRELATION SEE EQ. (3)
COMPUTE STD. ERROR OF R SEE EQ. (4)
STORE COMPUTED RESULTS IN OUTPUT AREAS
85
EXIT

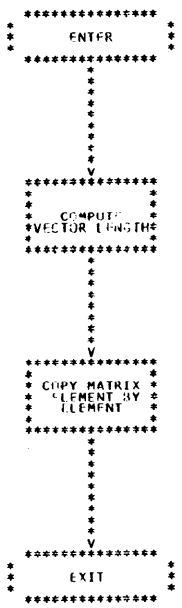

ENTER
IE=0
CONSTRUCT A 2 X 2 CONTINGENCY TABLE FROM INPUT DATA
IS ANY CELL EQUAL TO 0
NO
YES
IE=1
COMPUTE PHI COEFFICIENT SEE EQ. (1)
COMPUTE CHI SQUARE SEE EQ. (2)
COMPUTE THE MAXIMAL PHI COEFFICIENT
EXIT

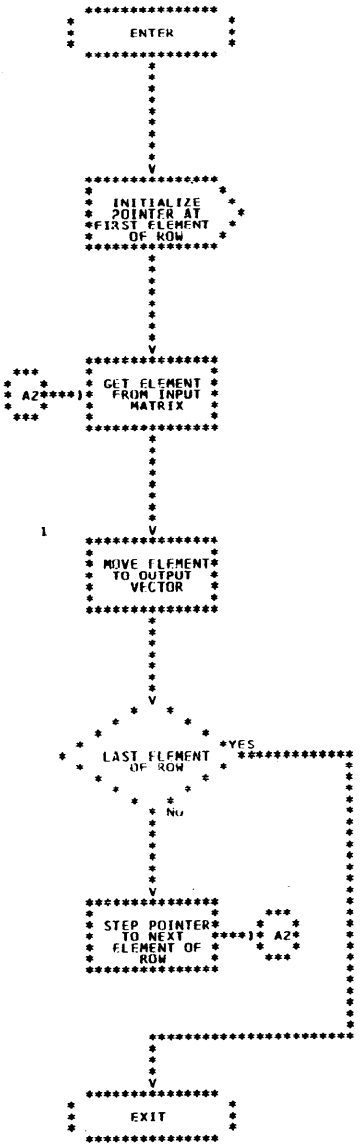

ENTER
IER=0
COMPUTE MEAN OF CONTINUOUS VARIABLE A SEE EQ. (1)
COMPUTE STD. DEVIATION OF A SEE EQ. (2)
FIND NO. OF CASES IN HIGHER, LOWER CATEGORIES OF B
IS ANY CASE IN LOWER CAT. OF B
YES
NO
IER=1
B4
ARE ALL CASES IN LOWER CAT. OF B
NO
YES
IER=-1
B4
COMPUTE MEAN OF A BELONG TO HIGHER CATEGORY OF B
B0
B0
COMPUTE MEAN OF A BELONG TO LOWER CATEGORY OF B
COMPUTE POINT BISERIAL CORRELATION SEE EQ. (3)
COMPUTE T RATIO SEE EQ. (4)
FIND DEGREES OF FREEDOM SEE EQ. (5)
B4
EXIT

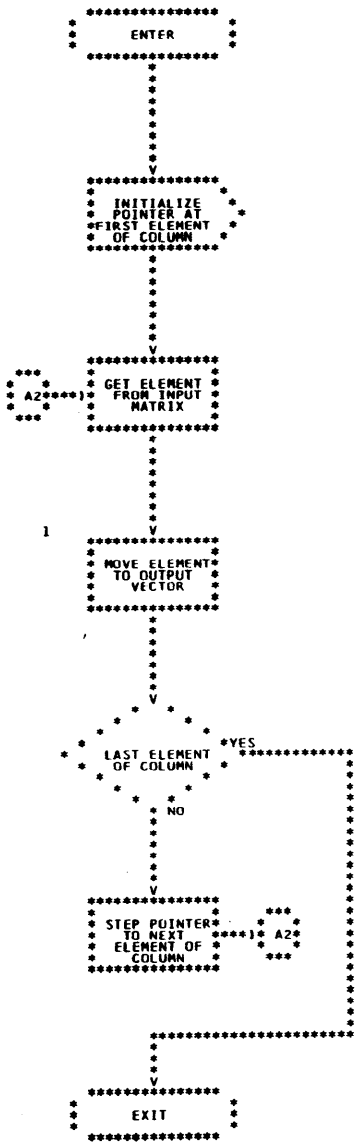

ENTER
CONSTRUCT A 2 X 2 CONTINGENCY TABLE FROM INPUT DATA
SET ERROR CODE, IE=0
IS ANY FREQUENCY = 0
YES
NO
COMPUTE P1 Q1, PROPORTIONS OF COL. TOTALS EQ. (3-3)
COMPUTE P2 Q2, PROPORTIONS OF ROW TOTALS EQ. (3-4)
NOTRI
ORDINATE OF N.C. AT P1
NOTRI
ORDINATE OF N.C. AT P2
PCLRT
COMPUTE R. EQ.(5)
ARE THERE MULTIPLE ROOTS OR NO ROOTS
NO
YES
78
81
80
SET IE=2, R=RS=10**75
78
IS THERE A SATISFACTORY ANSWER
YES
NO
SET IE=3 OR 4 BY SUBROUTINE POLRT, R=RS=10**75
90
COMPUTE STANDARD ERROR OF R=0.0 SEE EQ. (6)
60
SET IE=1, R=RS=10**75
100
EXIT

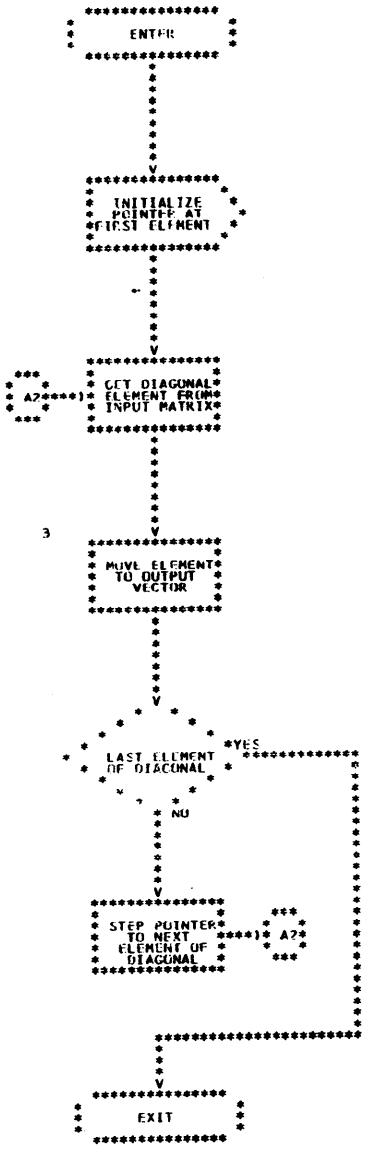
SUBROUTINE SRATE

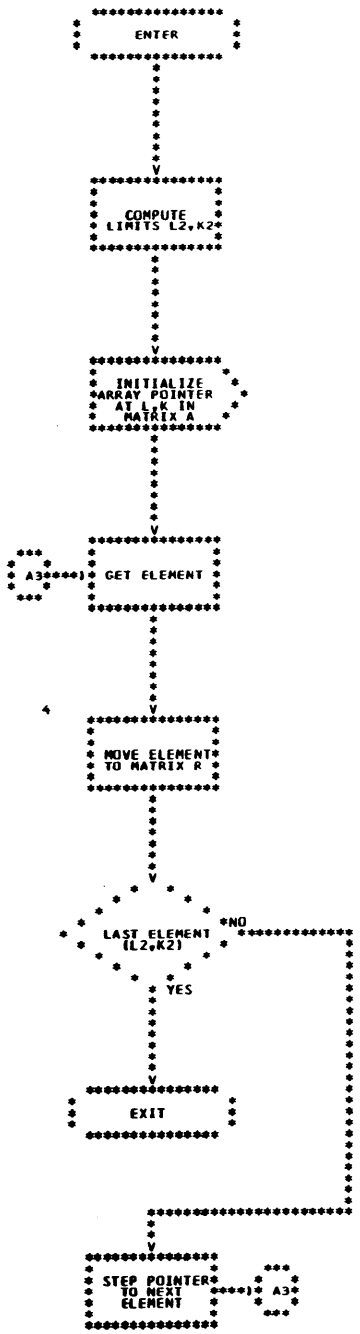


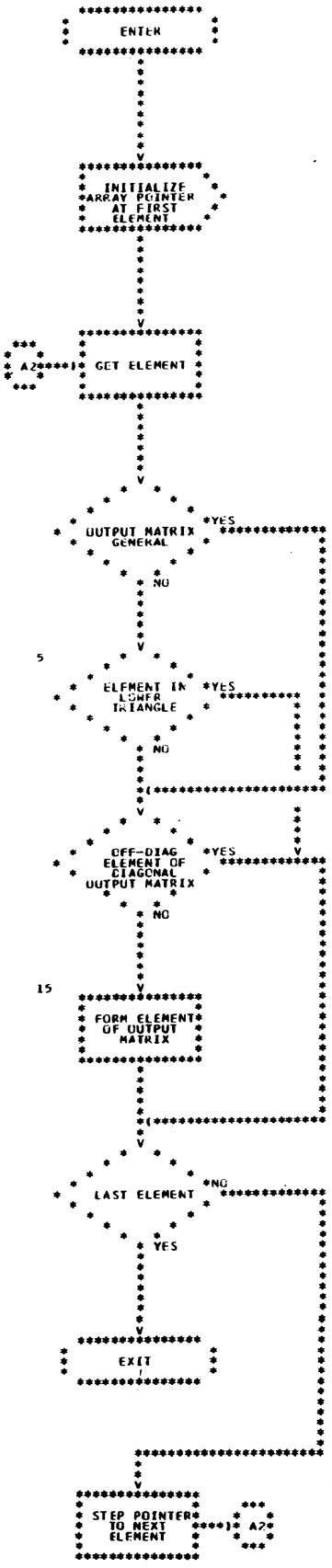

ENTER
COMPUTE
VECTOR LENGTHS
COPY MATRIX
ELEMENT BY
ELEMENT
EXIT

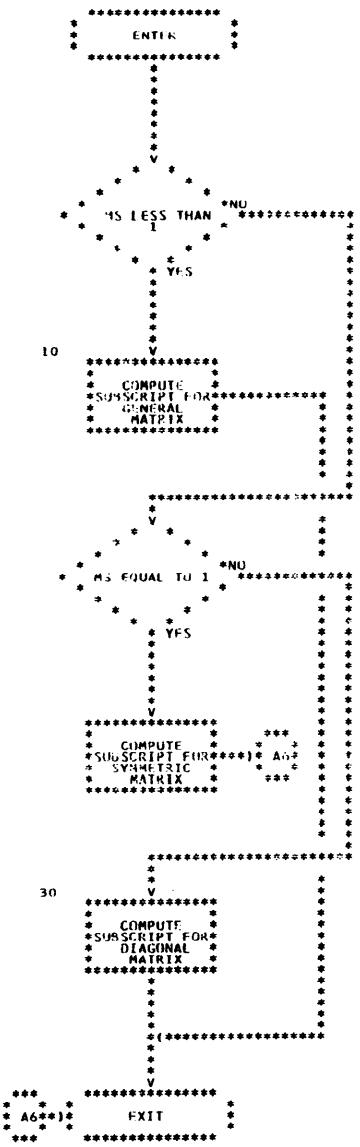

ENTER
INITIALIZE POINTER AT FIRST ELEMENT OF ROW
GET ELEMENT FROM INPUT MATRIX
A2
1
MOVE ELEMENT TO OUTPUT VECTOR
LAST ELEMENT OF ROW
YES
NO
STEP POINTER TO NEXT ELEMENT OF ROW
A2
EXIT

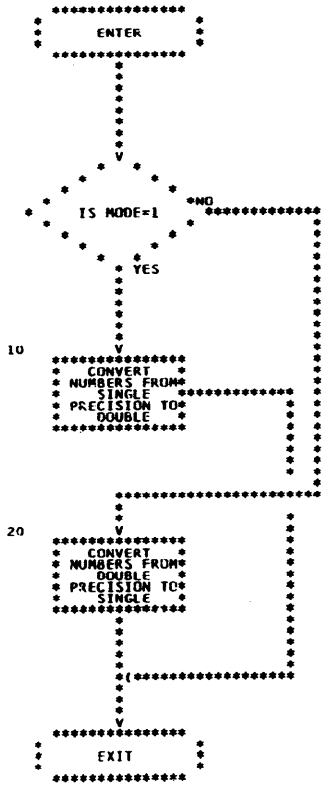

ENTER
INITIALIZE POINTER AT FIRST ELEMENT OF COLUMN
GET ELEMENT FROM INPUT MATRIX
A2
1
MOVE ELEMENT TO OUTPUT VECTOR
LAST ELEMENT OF COLUMN
YES
NO
STEP POINTER TO NEXT ELEMENT OF COLUMN
A2
EXIT

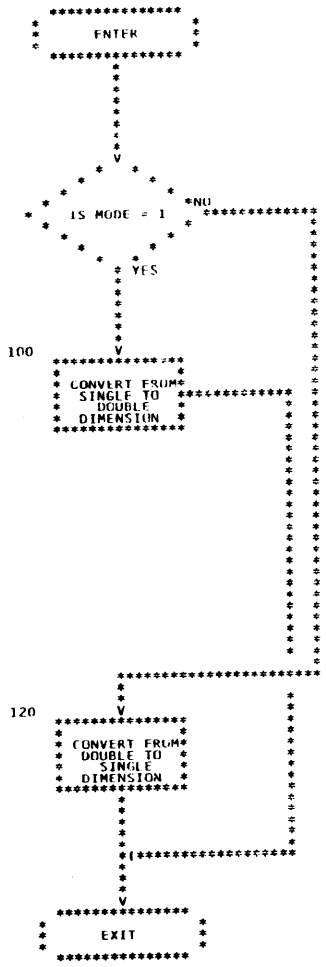

ENTER
INITIALIZE POINTER AT FIRST ELEMENT
A2
GET DIAGONAL ELEMENT FROM INPUT MATRIX
3
MOVE ELEMENT TO OUTPUT VECTOR
LAST ELEMENT OF DIAGONAL
YES
NO
STEP POINTER TO NEXT ELEMENT OF DIAGONAL
A2
EXIT

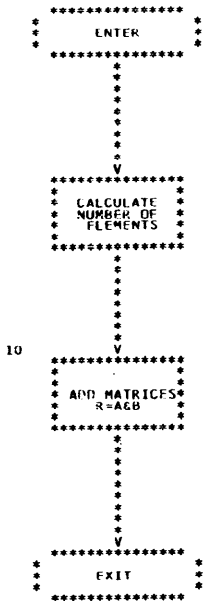

ENTER
COMPUTE LIMITS L2,K2
INITIALIZE ARRAY POINTER AT L,K IN MATRIX A
A3
GET ELEMENT
MOVE ELEMENT TO MATRIX R
LAST ELEMENT (L2,K2)
NO
YES
EXIT
STEP POINTER TO NEXT ELEMENT
A3

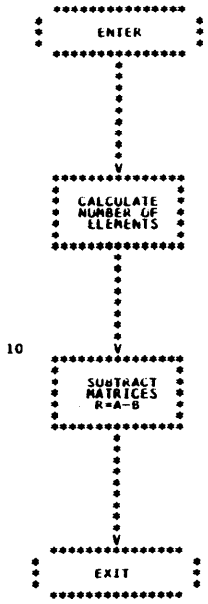

ENTER
INITIALIZE ARRAY POINTER AT FIRST ELEMENT
A2
GET ELEMENT
OUTPUT MATRIX GENERAL
YES
NO
5
ELEMENT IN LOWER TRIANGLE
YES
NO
OFF-DIAG ELEMENT OF DIAGONAL OUTPUT MATRIX
YES
NO
15
FORM ELEMENT OF OUTPUT MATRIX
LAST ELEMENT
NO
YES
EXIT
STEP POINTER TO NEXT ELEMENT
A2

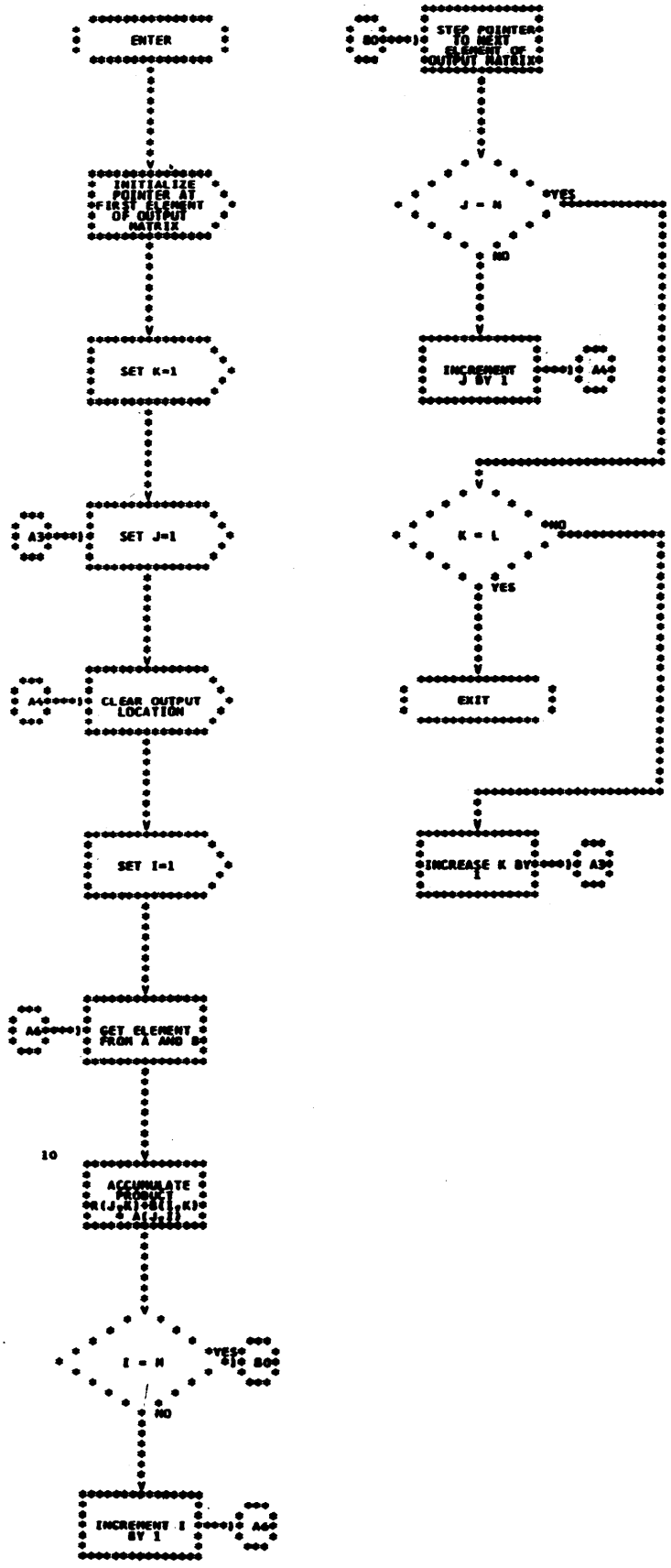

ENTER
MS LESS THAN 1
NO
YES
10
COMPUTE SUBSCRIPT FOR GENERAL MATRIX
MS EQUAL TO 1
NO
YES
COMPUTE SUBSCRIPT FOR SYMETRIC MATRIX
A6
30
COMPUTE SUBSCRIPT FOR DIAGONAL MATRIX
A6
EXIT

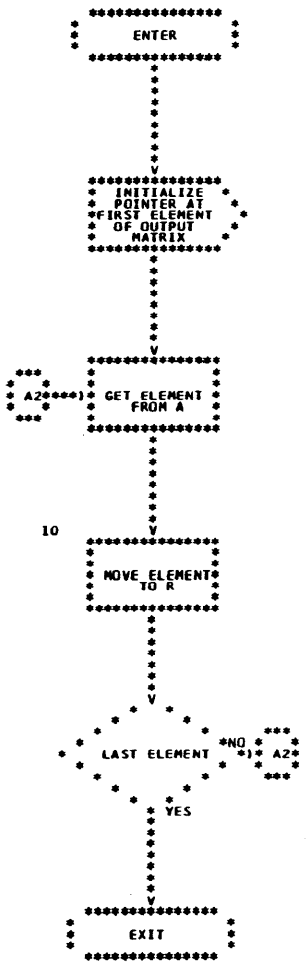

ENTER
IS MODE=1
NO
YES
10
CONVERT NUMBERS FROM SINGLE PRECISION TO DOUBLE
20
CONVERT NUMBERS FROM DOUBLE PRECISION TO SINGLE
EXIT

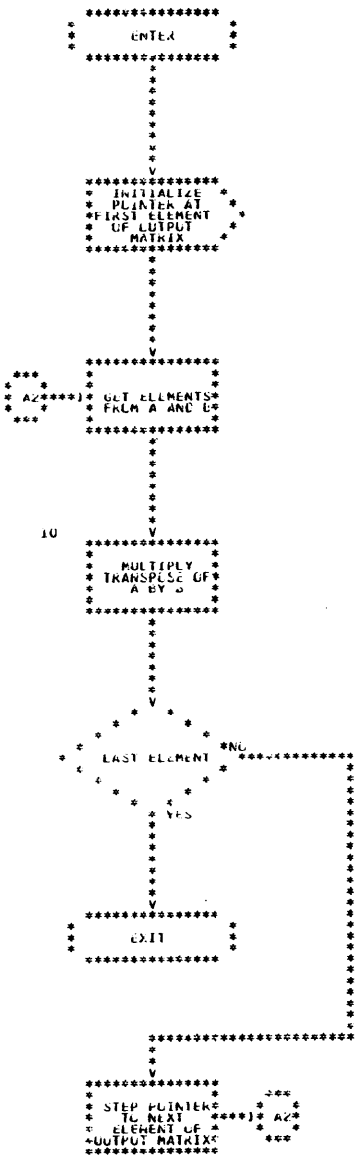

ENTER
IS MODE = 1
NO
YES
100
CONVERT FROM SINGLE TO DOUBLE DIMENSION
120
CONVERT FROM DOUBLE TO SINGLE DIMENSION
EXIT

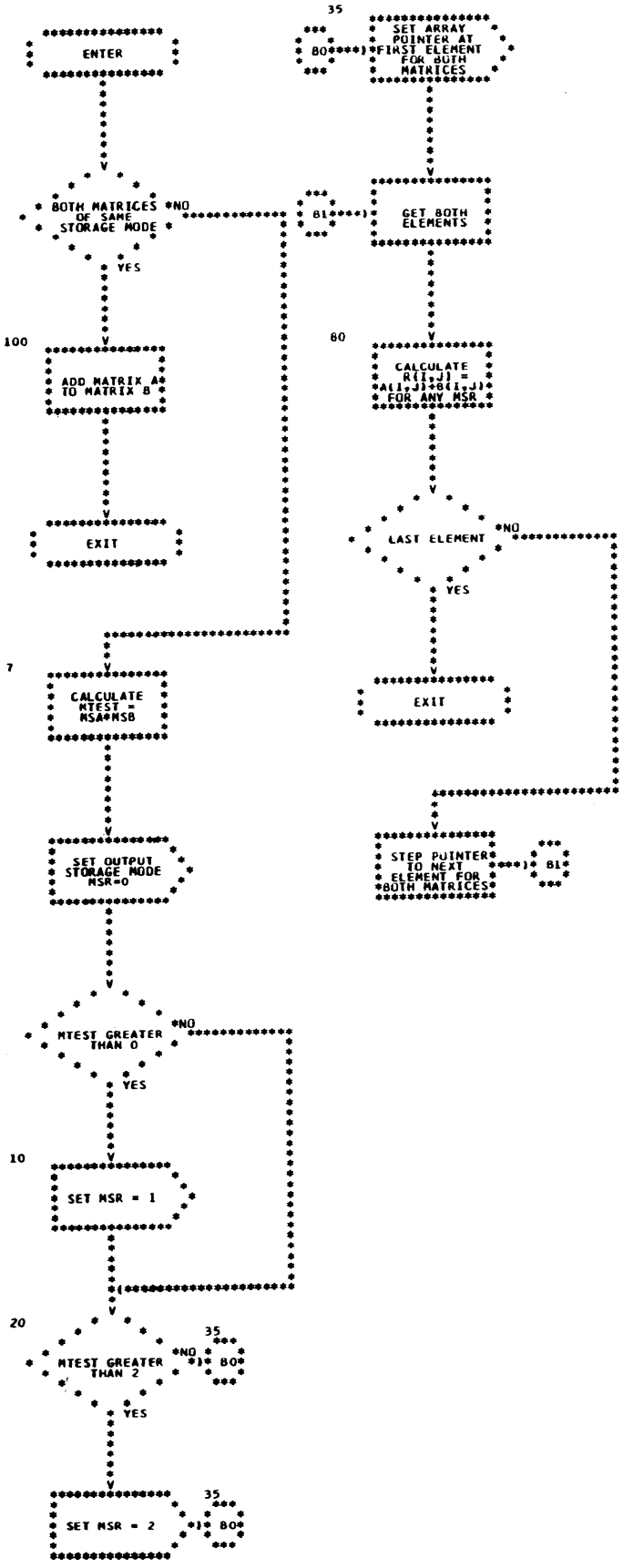

ENTER
CALCULATE NUMBER OF ELEMENTS
10
ADD MATRICES R=AGB
EXIT

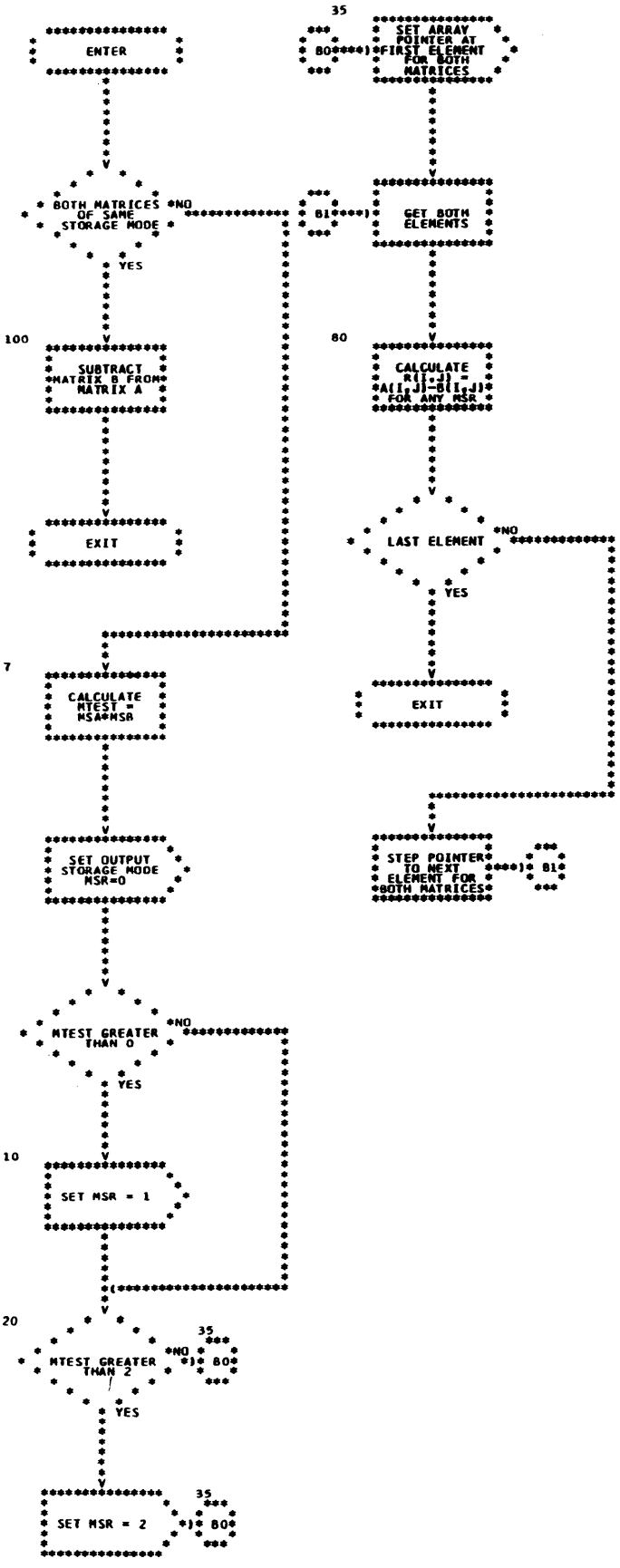

ENTER
CALCULATE NUMBER OF ELEMENTS
10
SUBTRACT MATRICES R=A-B
EXIT

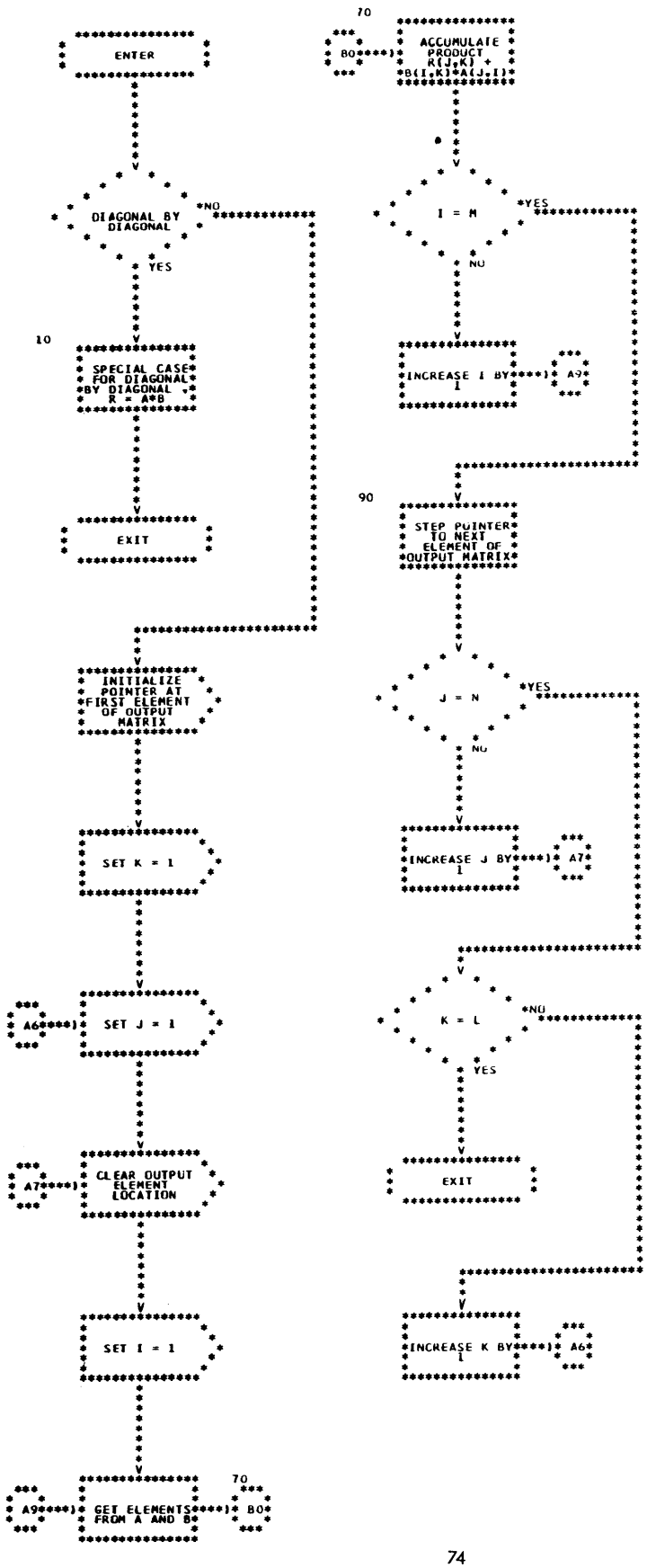

ENTER
INITIALIZE POINTER AT FIRST ELEMENT OF OUTPUT MATRIX
SET K=1
A3
SET J=1
A4
CLEAR OUTPUT LOCATION
SET I=1
A4
GET ELEMENT FROM A AND B
10
ACCUMULATE PRODUCT
I = N
YES
B0
NO
INCREMENT I BY 1
A4
B0
STEP POINTER TO NEXT ELEMENT OF OUTPUT MATRIX
J = N
YES
NO
INCREMENT J BY 1
A4
K = L
NO
YES
EXIT
INCREASE K BY 1
A3

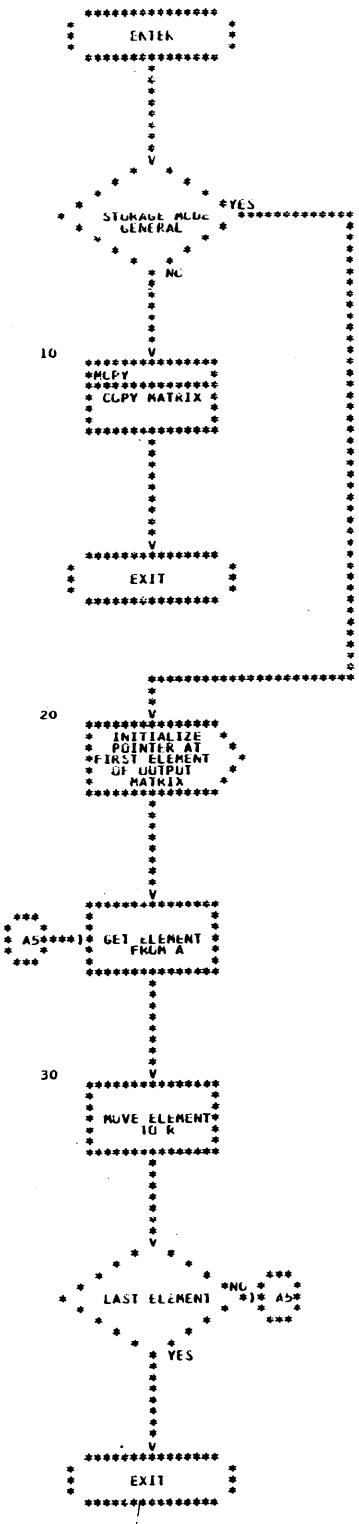

ENTER
INITIALIZE POINTER AT FIRST ELEMENT OF OUTPUT MATRIX
A2
GET ELEMENT FROM A
10
MOVE ELEMENT TO R
LAST ELEMENT
NO
A2
YES
EXIT

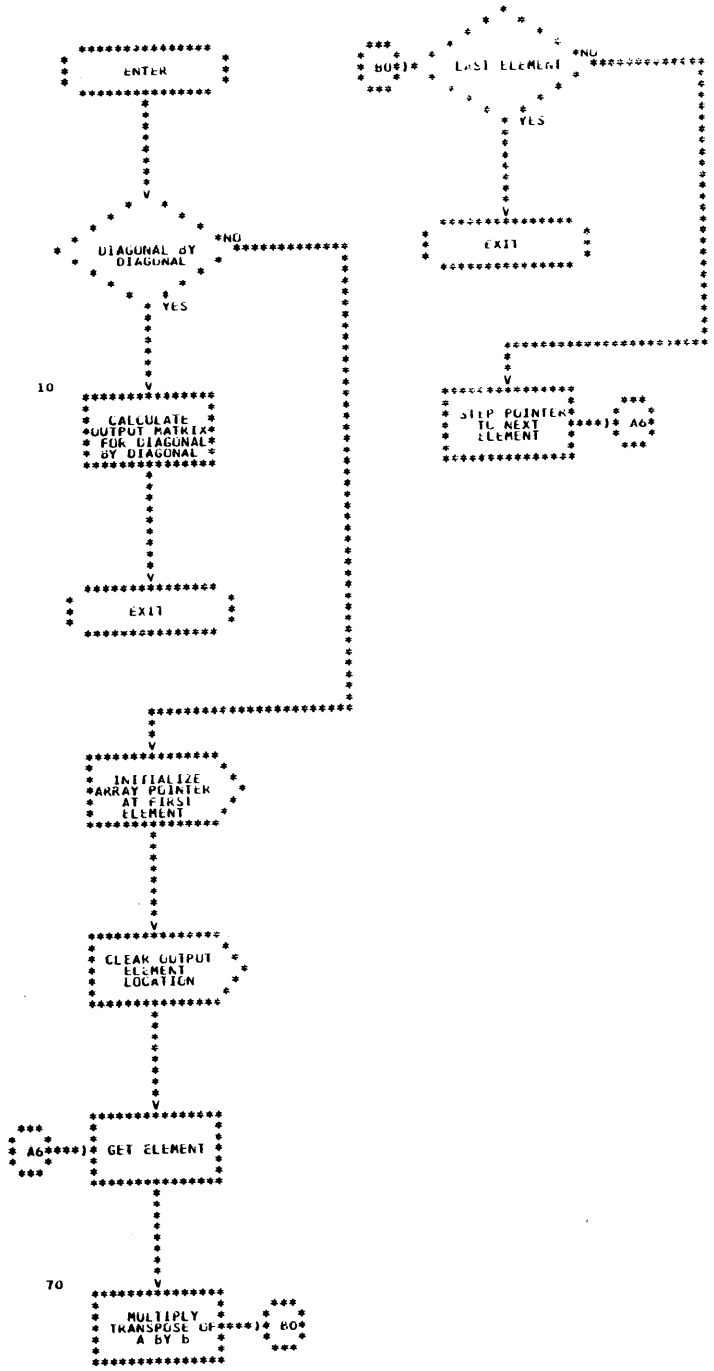

ENTER
INITIALIZE POINTER AT FIRST ELEMENT OF OUTPUT MATRIX
A2
GET ELEMENTS FROM A AND B
10
MULTIPLY TRANSPOSE OF A BY B
NO
LAST ELEMENT
YES
EXIT
STEP POINTER TO NEXT ELEMENT OF OUTPUT MATRIX
A2

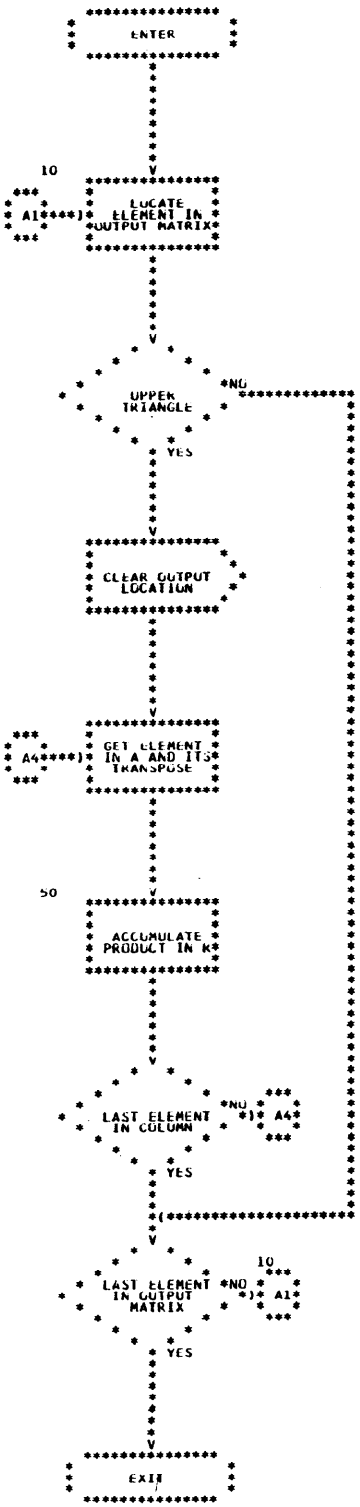

ENTER
BOTH MATRICES OF SAME STORAGE MODE
*NO
YES
100
ADD MATRIX A TO MATRIX B
EXIT
7
CALCULATE MTEST = MSA*MSB
SET OUTPUT STORAGE MODE MSR=0
MTEST GREATER THAN 0
*NO
YES
10
SET MSR = 1
20
MTEST GREATER THAN 2
*NO
35
80
YES
SET MSR = 2
35
80
81
SET ARRAY POINTER AT FIRST ELEMENT FOR BOTH MATRICES
GET BOTH ELEMENTS
80
CALCULATE R(I,J) = A(I,J)+B(I,J) FOR ANY MSR
LAST ELEMENT
*NO
YES
EXIT
STEP POINTER TO NEXT ELEMENT FOR BOTH MATRICES
81

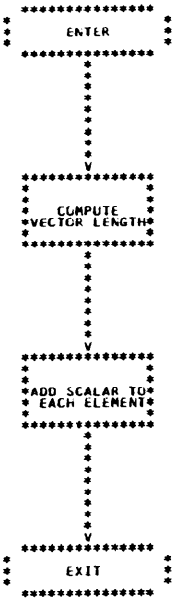

ENTER
BOTH MATRICES OF SAME STORAGE MODE
NO
YES
100
SUBTRACT MATRIX B FROM MATRIX A
EXIT
7
CALCULATE MTEST = MSA*MSA
SET OUTPUT STORAGE MODE MSR=0
MTEST GREATER THAN 0
NO
YES
10
SET MSR = 1
20
MTEST GREATER THAN 2
NO
35
80
YES
SET MSR = 2
35
80
81
GET BOTH ELEMENTS
80
CALCULATE R[I,J] = A[I,J]-B[I,J] FOR ANY MSR
LAST ELEMENT
NO
YES
EXIT
STEP POINTER TO NEXT ELEMENT FOR BOTH MATRICES
81
35
80
SET ARRAY POINTER AT FIRST ELEMENT FOR BOTH MATRICES

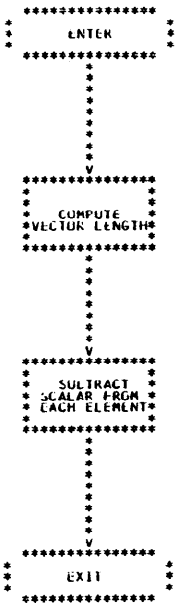

ENTER
DIAGONAL BY DIAGONAL
NO
YES
10
SPECIAL CASE FOR DIAGONAL BY DIAGONAL R = A*B
EXIT
INITIALIZE POINTER AT FIRST ELEMENT OF OUTPUT MATRIX
SET K = 1
A6
SET J = 1
A7
CLEAR OUTPUT ELEMENT LOCATION
SET I = 1
A9
GET ELEMENTS FROM A AND B
70
B0
ACCUMULATE PRODUCT R(J,K) + B(I,K)*A(J,I)
I = M
YES
NO
INCREASE I BY 1
A9
90
STEP POINTER TO NEXT ELEMENT OF OUTPUT MATRIX
J = N
YES
NO
INCREASE J BY 1
A7
K = L
NO
YES
EXIT
INCREASE K BY 1
A6

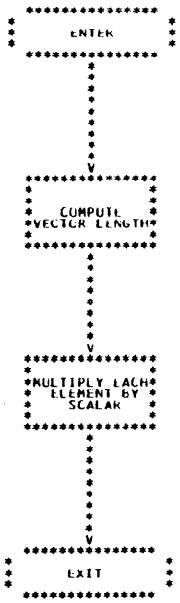

ENTER
STORAGE ALLOC GENERAL
YES
NO
10
MCPY
COPY MATRIX
EXIT
20
INITIALIZE POINTER AT FIRST ELEMENT OF OUTPUT MATRIX
GET ELEMENT FROM A
A5
30
MOVE ELEMENT TO R
LAST ELEMENT
NO
A5
YES
EXIT

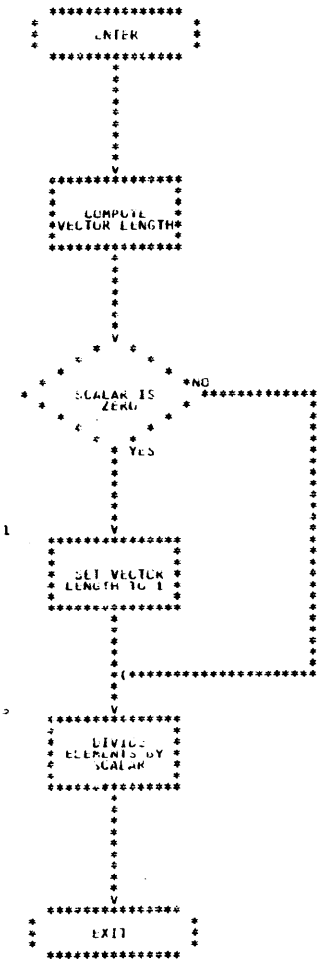

ENTER
DIAGONAL BY DIAGONAL
NO
YES
10
CALCULATE OUTPUT MATRIX FOR DIAGONAL BY DIAGONAL
EXIT
INITIALIZE ARRAY POINTER AT FIRST ELEMENT
CLEAR OUTPUT ELEMENT LOCATION
A6
GET ELEMENT
70
MULTIPLY TRANSPOSE OF A BY B
B0
B0
LAST ELEMENT
NO
YES
EXIT
STEP POINTER TO NEXT ELEMENT
A6

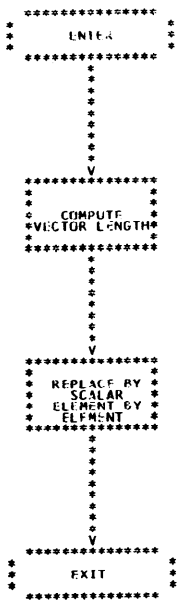

ENTER
10
A1
LOCATE ELEMENT IN OUTPUT MATRIX
UPPER TRIANGLE
NO
YES
CLEAR OUTPUT LOCATION
A4
GET ELEMENT IN A AND ITS TRANSPOSE
50
ACCUMULATE PRODUCT IN K
LAST ELEMENT IN COLUMN
NO
A4
YES
10
LAST ELEMENT IN OUTPUT MATRIX
NO
A1
YES
EXIT

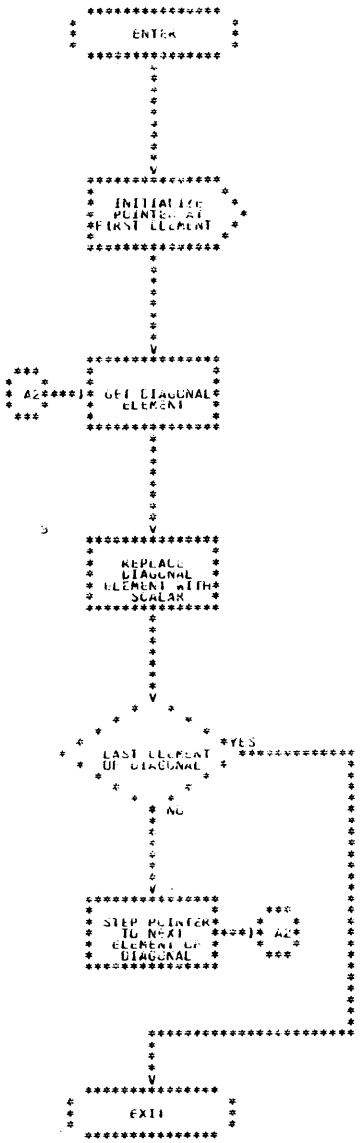

ENTER
COMPUTE VECTOR LENGTH
ADD SCALAR TO EACH ELEMENT
EXIT

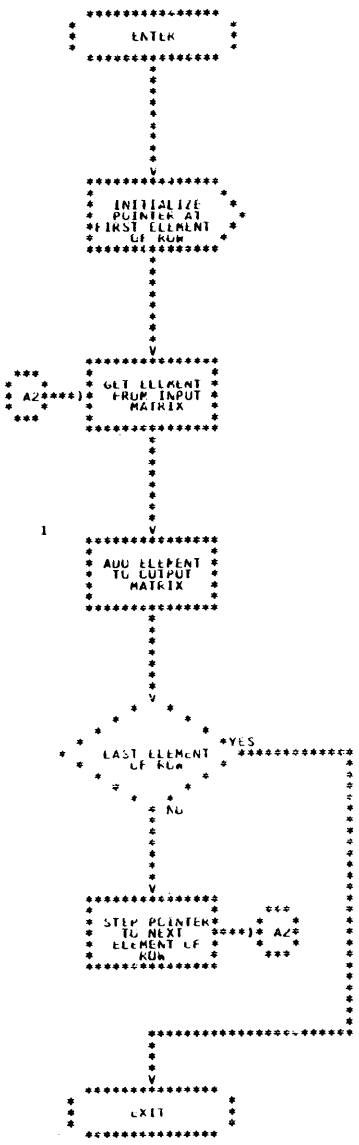

ENTER
COMPUTE VECTOR LENGTH
SUBTRACT SCALAR FROM EACH ELEMENT
EXIT

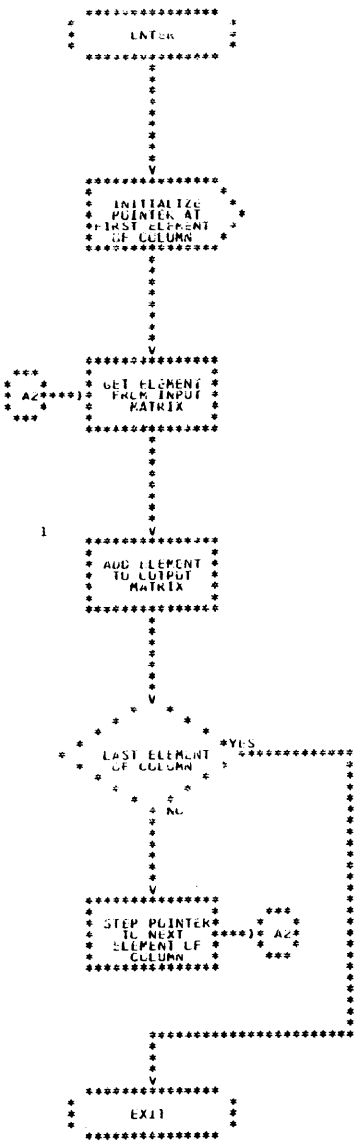

ENTER
COMPUTE VECTOR LENGTH
MULTIPLY EACH ELEMENT BY SCALAR
EXIT

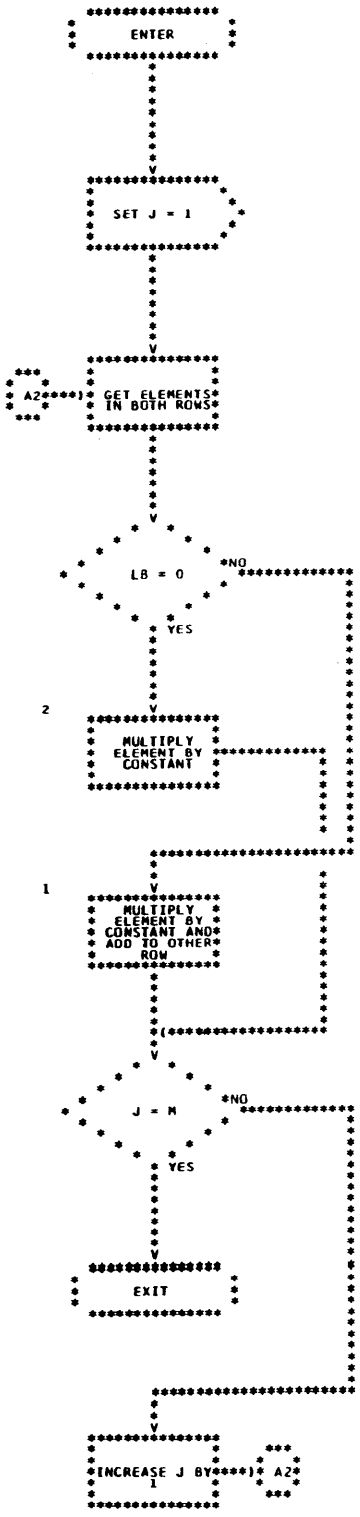

ENTER
COMPUTE VECTOR LENGTH
SCALAR IS ZERO
NO
YES
SET VECTOR LENGTH TO 1
EXIT

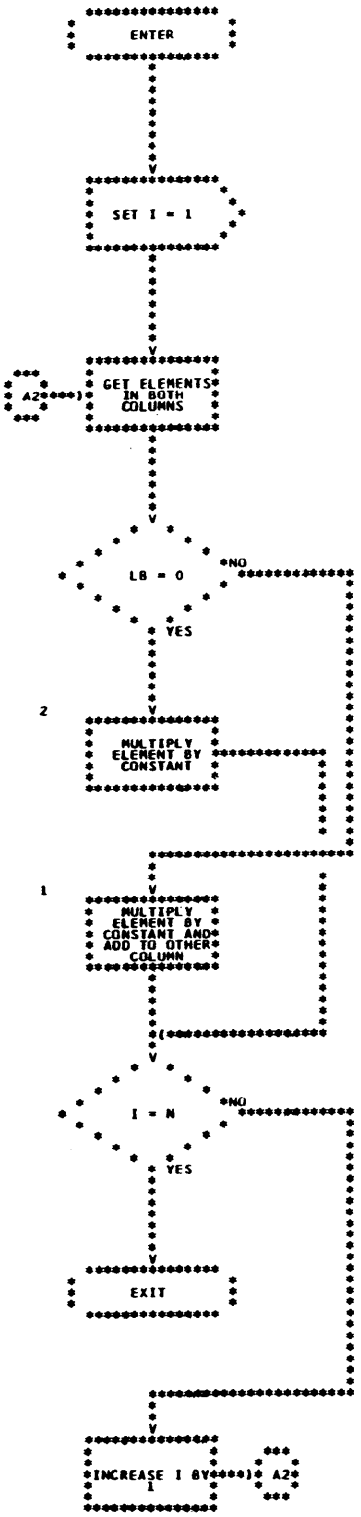

ENTER
COMPUTE VECTOR LENGTH
REPLACE BY SCALAR ELEMENT BY ELEMENT
EXIT

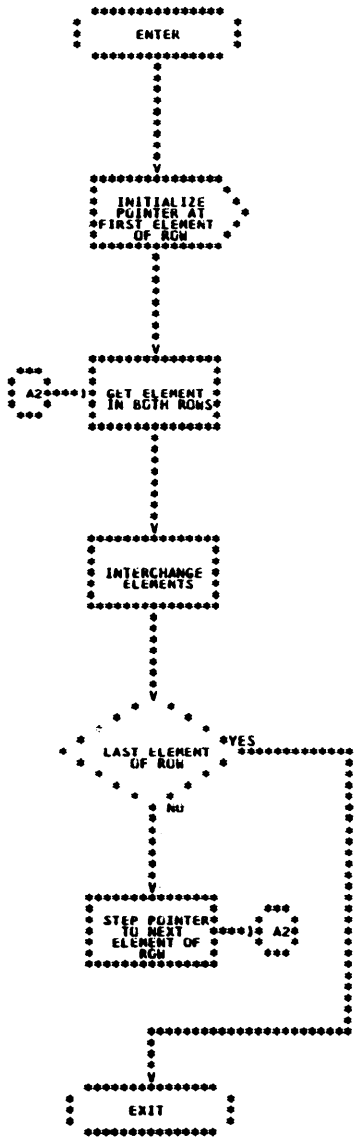

ENTER
INITIALIZE POINTER AT FIRST ELEMENT
A2
GET DIAGONAL ELEMENT
REPLACE DIAGONAL ELEMENT WITH SCALAR
LAST ELEMENT OF DIAGONAL
YES
A0
STEP POINTER TO NEXT ELEMENT OF DIAGONAL
A2
EXIT

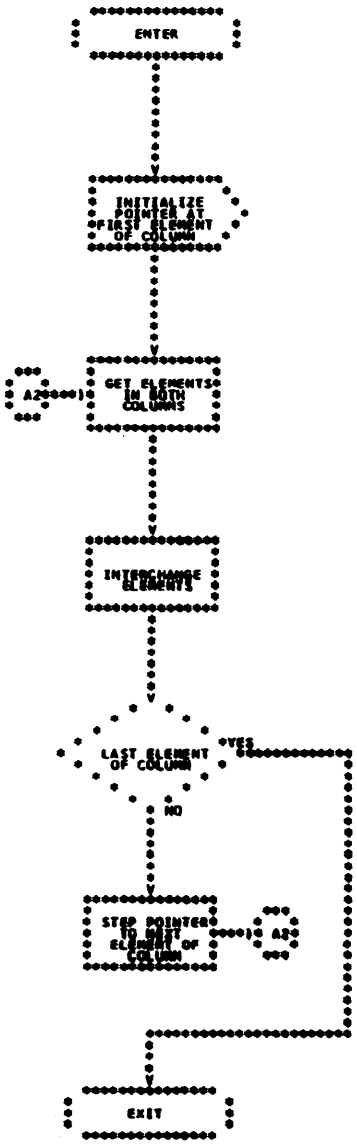

ENTER
INITIALIZE POINTER AT FIRST ELEMENT OF ROW
GET ELEMENT FROM INPUT MATRIX
A2
ADD ELEMENT TO OUTPUT MATRIX
LAST ELEMENT OF ROW
YES
NO
STEP POINTER TO NEXT ELEMENT OF ROW
A2
EXIT

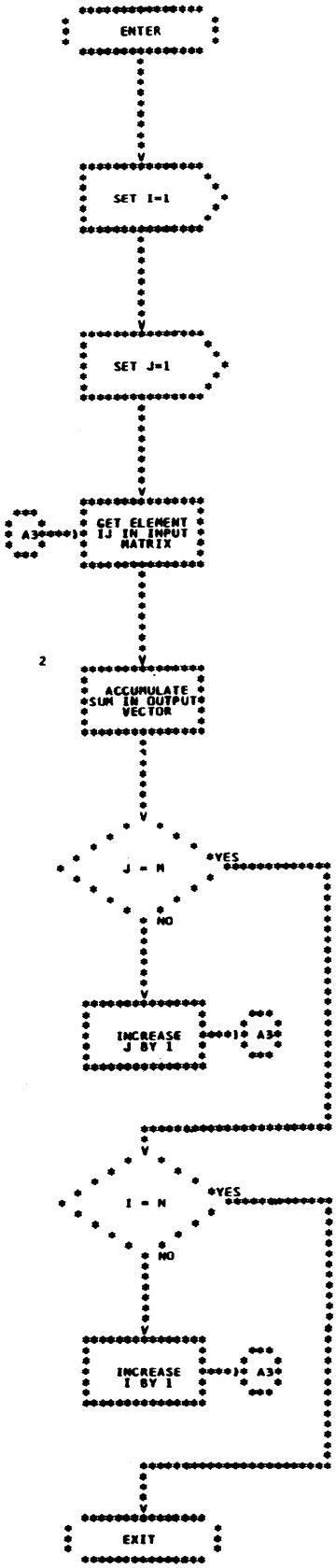

ENTER
INITIALIZE POINTER AT FIRST ELEMENT OF COLUMN
GET ELEMENT FROM INPUT MATRIX
A2
1
ADD ELEMENT TO OUTPUT MATRIX
LAST ELEMENT OF COLUMN
YES
NO
STEP POINTER TO NEXT ELEMENT OF COLUMN
A2
EXIT

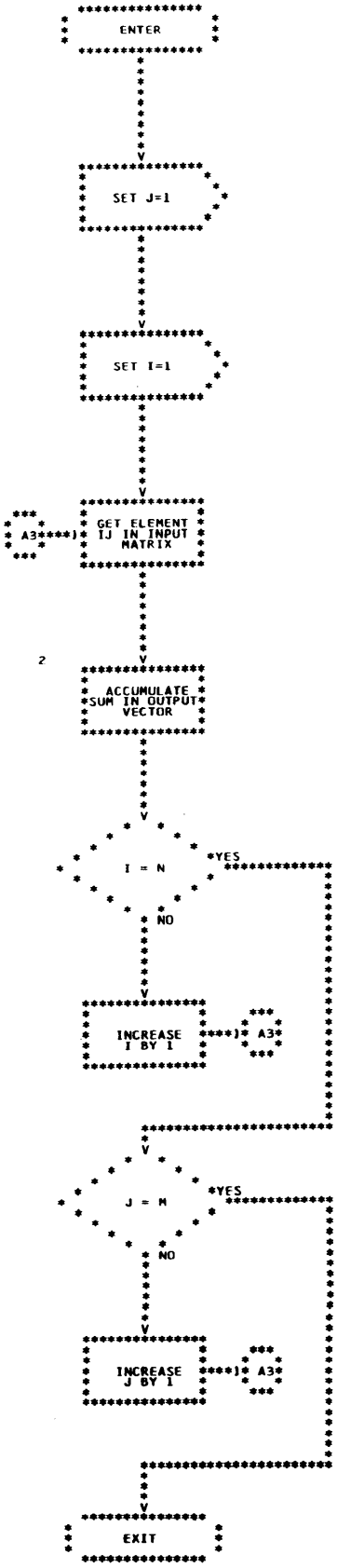

ENTER
SET J = 1
A2
GET ELEMENTS IN BOTH ROWS
LB = 0
NO
YES
2
MULTIPLY ELEMENT BY CONSTANT
1
MULTIPLY ELEMENT BY CONSTANT AND ADD TO OTHER ROW
J = M
NO
YES
EXIT
INCREASE J BY 1
A2

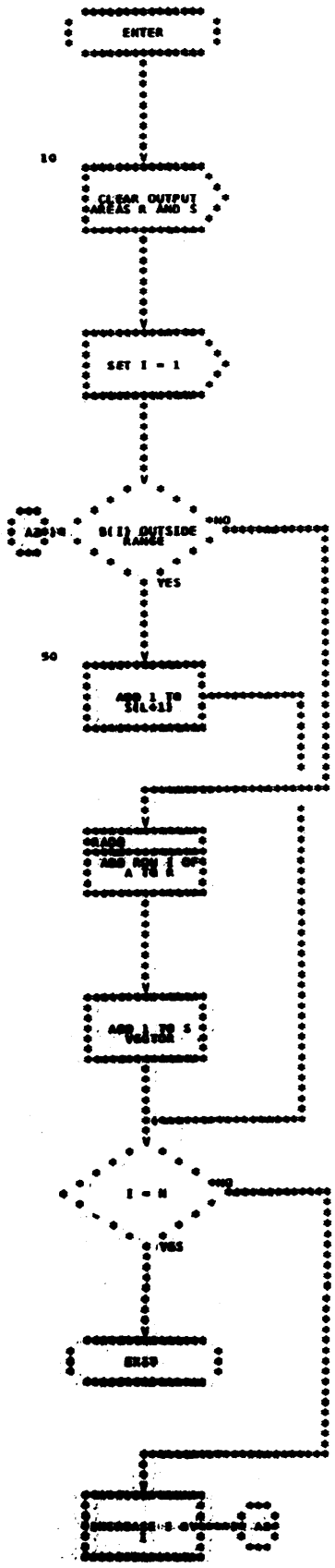

ENTER
SET I = 1
GET ELEMENTS IN BOTH COLUMNS
A2
LB = 0
NO
YES
2
MULTIPLY ELEMENT BY CONSTANT
1
MULTIPLY ELEMENT BY CONSTANT AND ADD TO OTHER COLUMN
I = N
NO
YES
EXIT
INCREASE I BY 1
A2


ENTER
INITIALIZE POINTER AT FIRST ELEMENT OF ROW
A2
GET ELEMENT IN BOTH ROWS
INTERCHANGE ELEMENTS
LAST ELEMENT OF ROW
YES
NO
STEP POINTER TO NEXT ELEMENT OF ROW
A2
EXIT

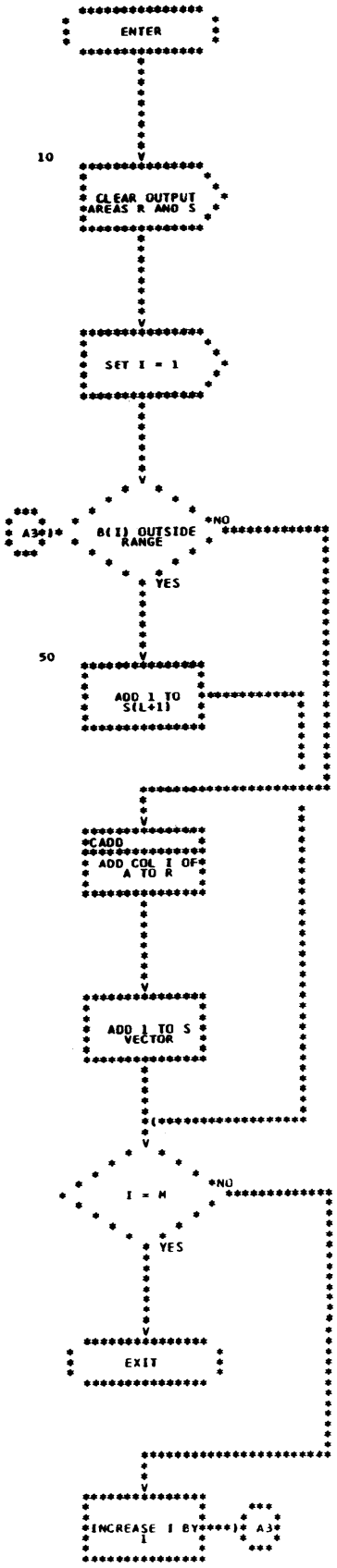

ENTER
INITIALIZE POINTER AT FIRST ELEMENT OF COLUMN
A2
GET ELEMENTS IN BOTH COLUMNS
INTERCHANGE ELEMENTS
LAST ELEMENT OF COLUMN
YES
NO
STEP POINTER TO NEXT ELEMENT OF COLUMN
A2
EXIT

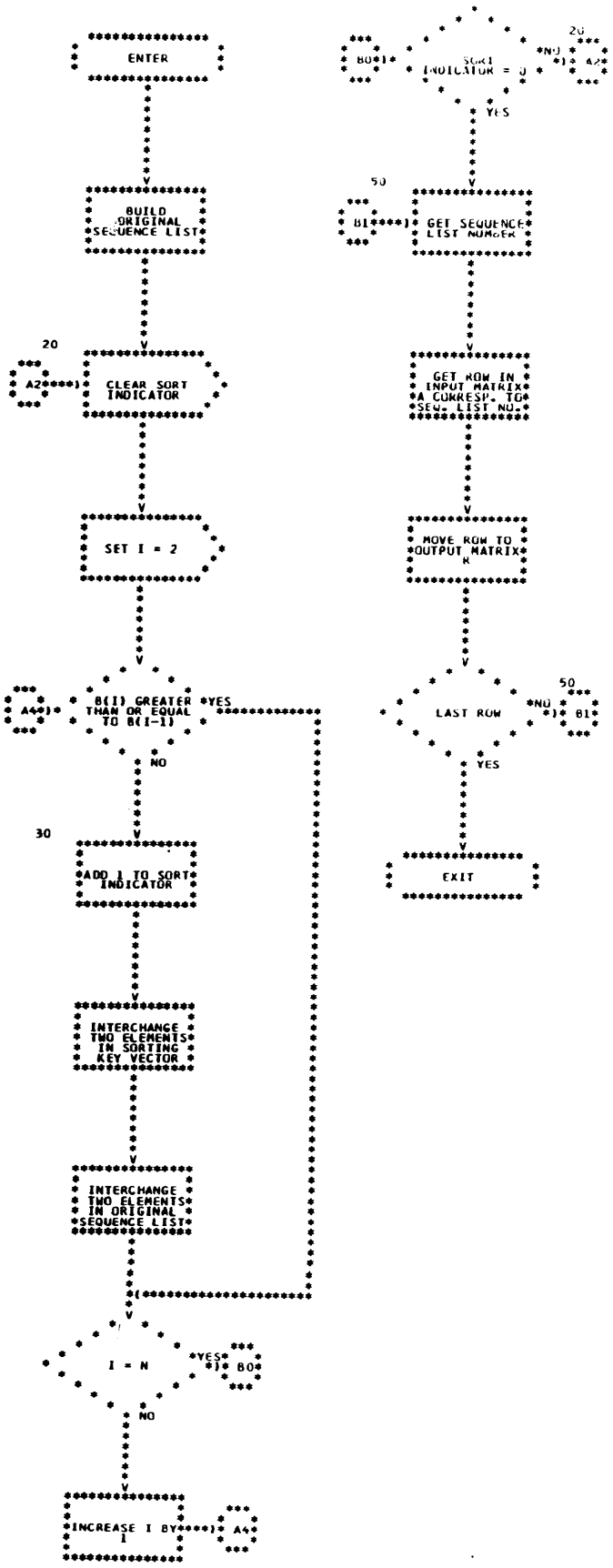

ENTER
SET I=1
SET J=1
A3
GET ELEMENT IJ IN INPUT MATRIX
2
ACCUMULATE SUM IN OUTPUT VECTOR
J = N
YES
NO
INCREASE J BY 1
A3
I = N
YES
NO
INCREASE I BY 1
A3
EXIT

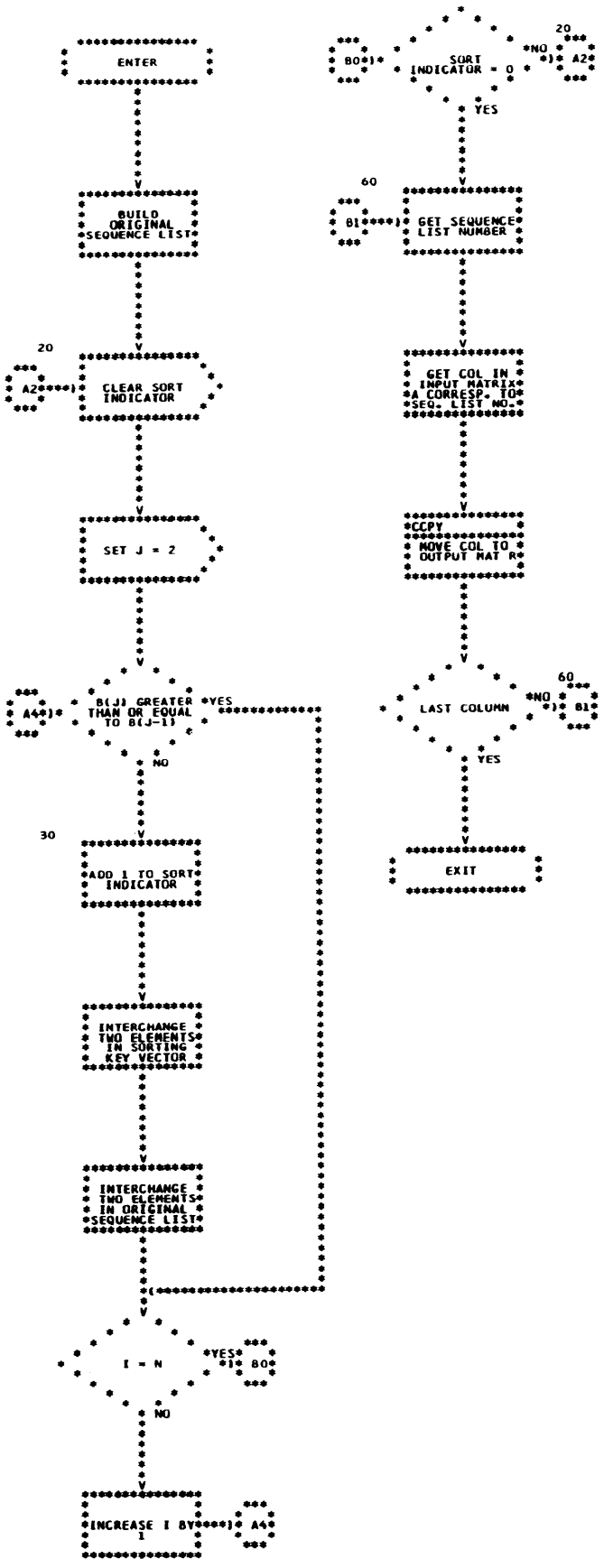

ENTER
SET J=1
SET I=1
A3
GET ELEMENT IJ IN INPUT MATRIX
2
ACCUMULATE SUM IN OUTPUT VECTOR
I = N
YES
NO
INCREASE I BY 1
A3
J = M
YES
NO
INCREASE J BY 1
A3
EXIT

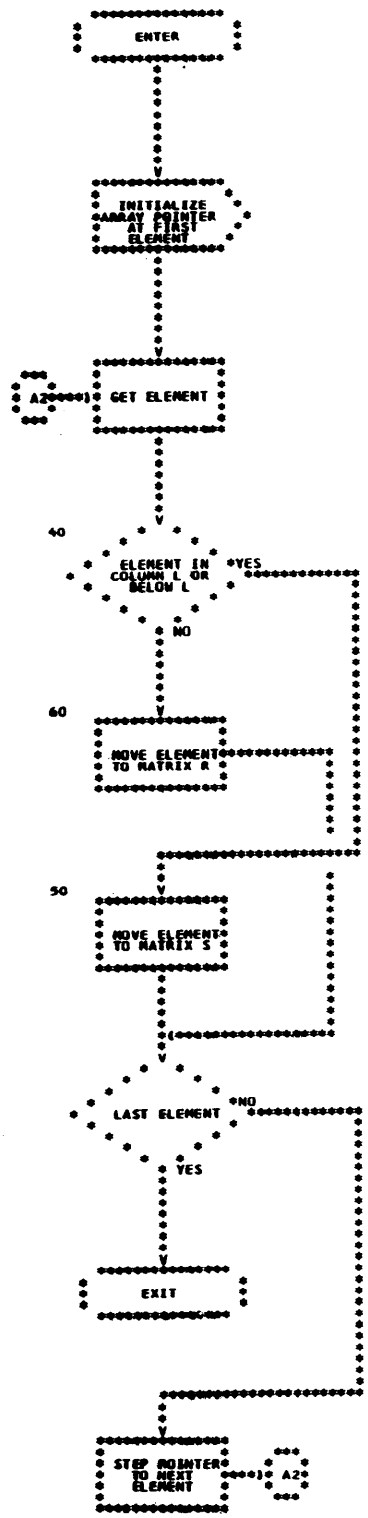

ENTER
10
CLEAR OUTPUT AREAS R AND S
SET I = 1
B(I) OUTSIDE RANGE
NO
YES
50
ADD 1 TO S(L+1)
ADD B(I) TO R(L+1)
ADD 1 TO S(L+1)
I = N
NO
YES
ERR59

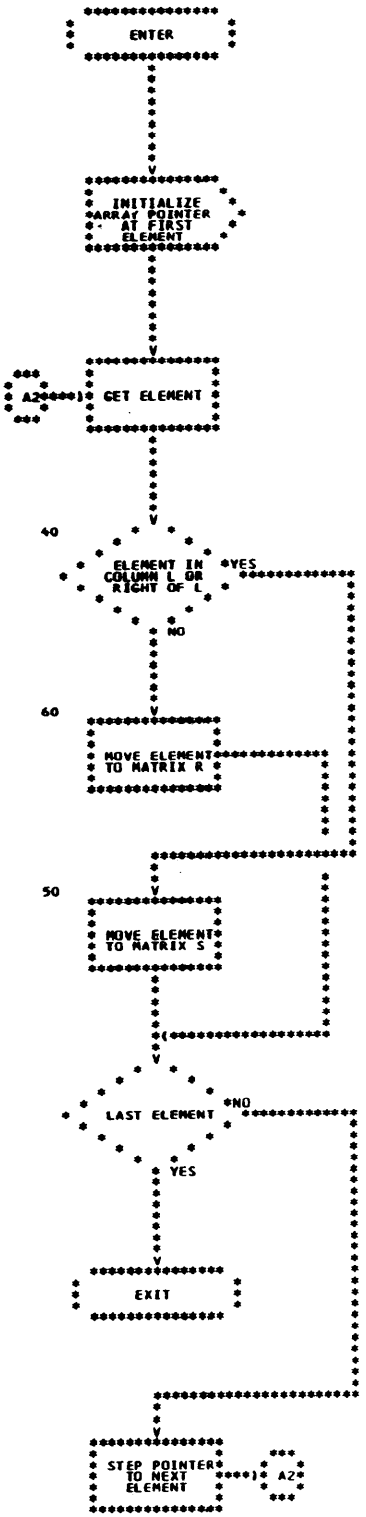
SUBROUTINE CTAB

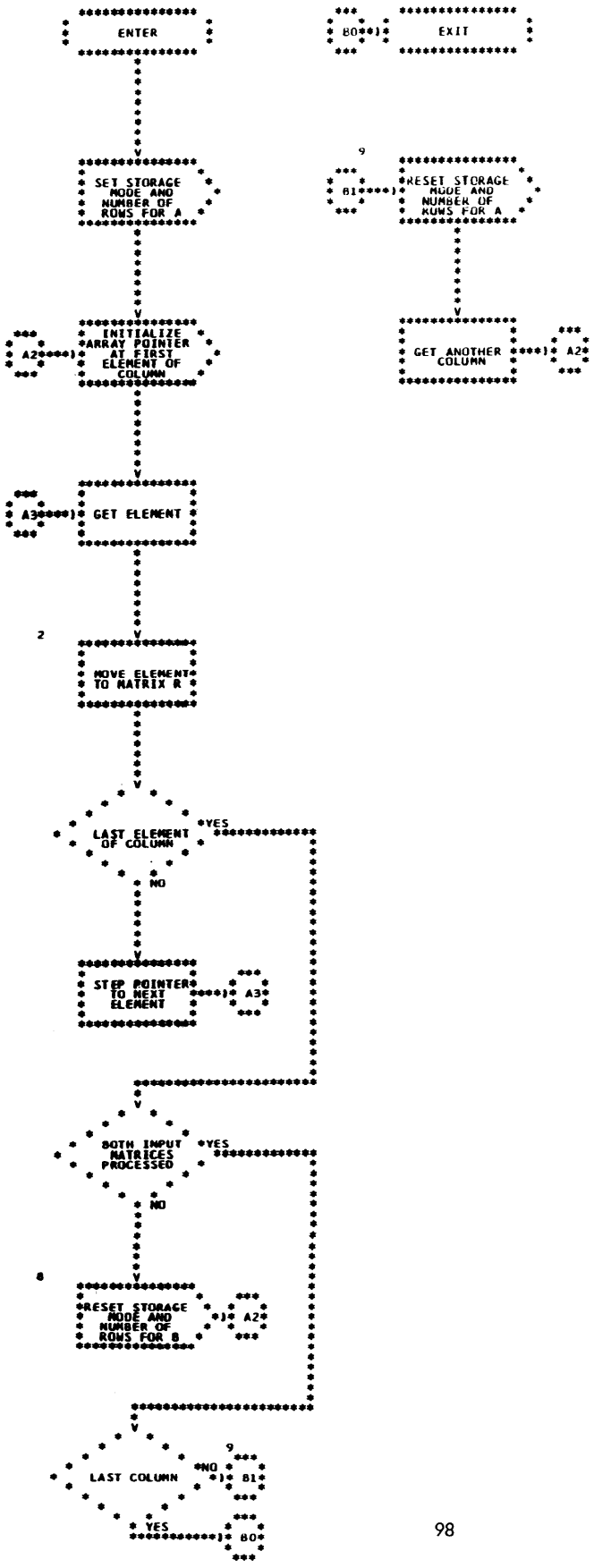


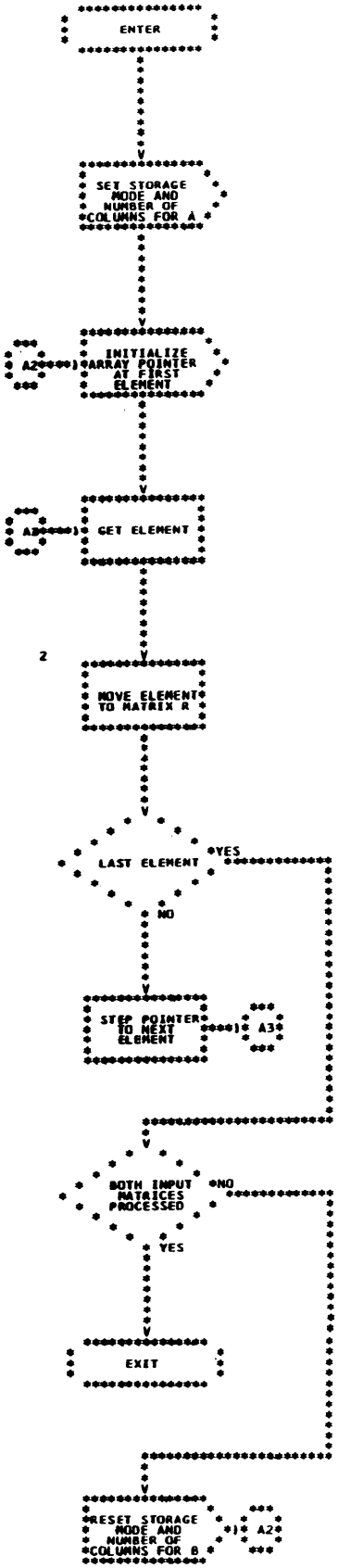

SUBROUTINE RSRT
ENTER
BUILD ORIGINAL SEQUENCE LIST
20
A2
CLEAR SORT INDICATOR
SET I = 2
A4
B(I) GREATER THAN OR EQUAL TO B(I-1)
YES
NO
30
ADD 1 TO SORT INDICATOR
INTERCHANGE TWO ELEMENTS IN SORTING KEY VECTOR
INTERCHANGE TWO ELEMENTS IN ORIGINAL SEQUENCE LIST
I = N
YES
B0
NO
INCREASE I BY 1
A4
B0
SORT INDICATOR = 0
20
NO
A2
YES
50
B1
GET SEQUENCE LIST NUMBER
GET ROW IN INPUT MATRIX A CORRESP. TO SEQ. LIST NO.
MOVE ROW TO OUTPUT MATRIX R
LAST ROW
50
NO
B1
YES
EXIT

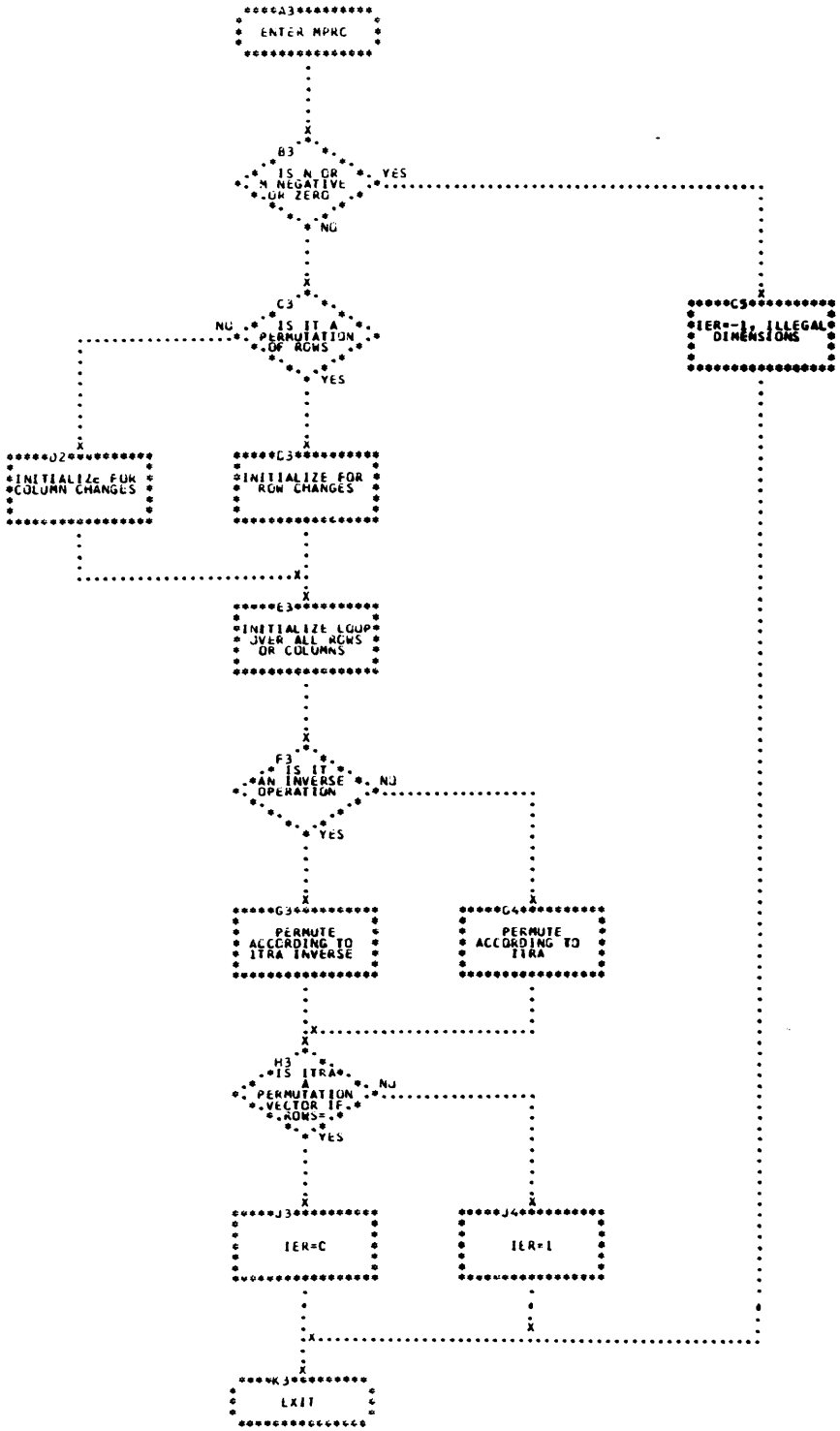

ENTER
BUILD ORIGINAL SEQUENCE LIST
20
A2
CLEAR SORT INDICATOR
SET J = 2
A4
B(J) GREATER THAN OR EQUAL TO B(J-1)
YES
NO
30
ADD 1 TO SORT INDICATOR
INTERCHANGE TWO ELEMENTS IN SORTING KEY VECTOR
INTERCHANGE TWO ELEMENTS IN ORIGINAL SEQUENCE LIST
I = N
YES
B0
NO
INCREASE I BY 1
A4
B0
SORT INDICATOR = 0
NO
20
A2
YES
60
B1
GET SEQUENCE LIST NUMBER
GET COL IN INPUT MATRIX A CORRESP. TO SEQ. LIST NO.
CCPY
MOVE COL TO OUTPUT MAT R
LAST COLUMN
NO
60
B1
YES
EXIT


ENTER
INITIALIZE ARRAY POINTER AT FIRST ELEMENT
A2
GET ELEMENT
40
ELEMENT IN COLUMN L OR BELOW L
YES
NO
60
MOVE ELEMENT TO MATRIX R
50
MOVE ELEMENT TO MATRIX S
LAST ELEMENT
NO
YES
EXIT
STEP POINTER TO NEXT ELEMENT
A2

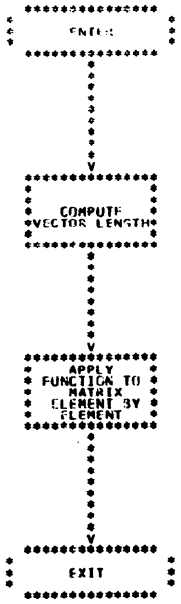

ENTER
INITIALIZE ARRAY POINTER AT FIRST ELEMENT
A2
GET ELEMENT
40
ELEMENT IN COLUMN L OR RIGHT OF L
YES
NO
60
MOVE ELEMENT TO MATRIX R
50
MOVE ELEMENT TO MATRIX S
LAST ELEMENT
NO
YES
EXIT
STEP POINTER TO NEXT ELEMENT
A2

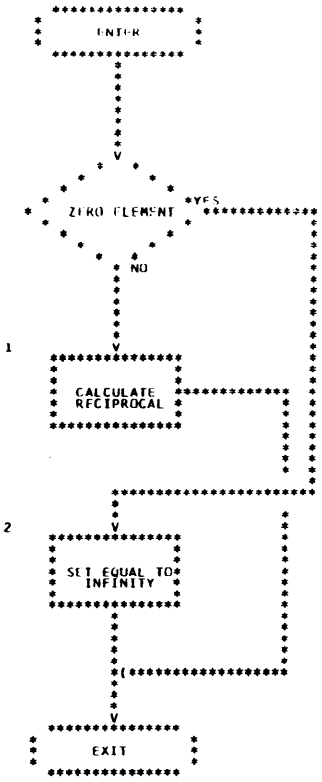

ENTER
B0
EXIT
9
SET STORAGE MODE AND NUMBER OF ROWS FOR A
B1
RESET STORAGE MODE AND NUMBER OF ROWS FOR A
INITIALIZE ARRAY POINTER AT FIRST ELEMENT OF COLUMN
A2
GET ANOTHER COLUMN
A2
A3
GET ELEMENT
2
MOVE ELEMENT TO MATRIX R
LAST ELEMENT OF COLUMN
YES
NO
STEP POINTER TO NEXT ELEMENT
A3
BOTH INPUT MATRICES PROCESSED
YES
NO
8
RESET STORAGE MODE AND NUMBER OF ROWS FOR B
A2
9
LAST COLUMN
NO
B1
YES
B0

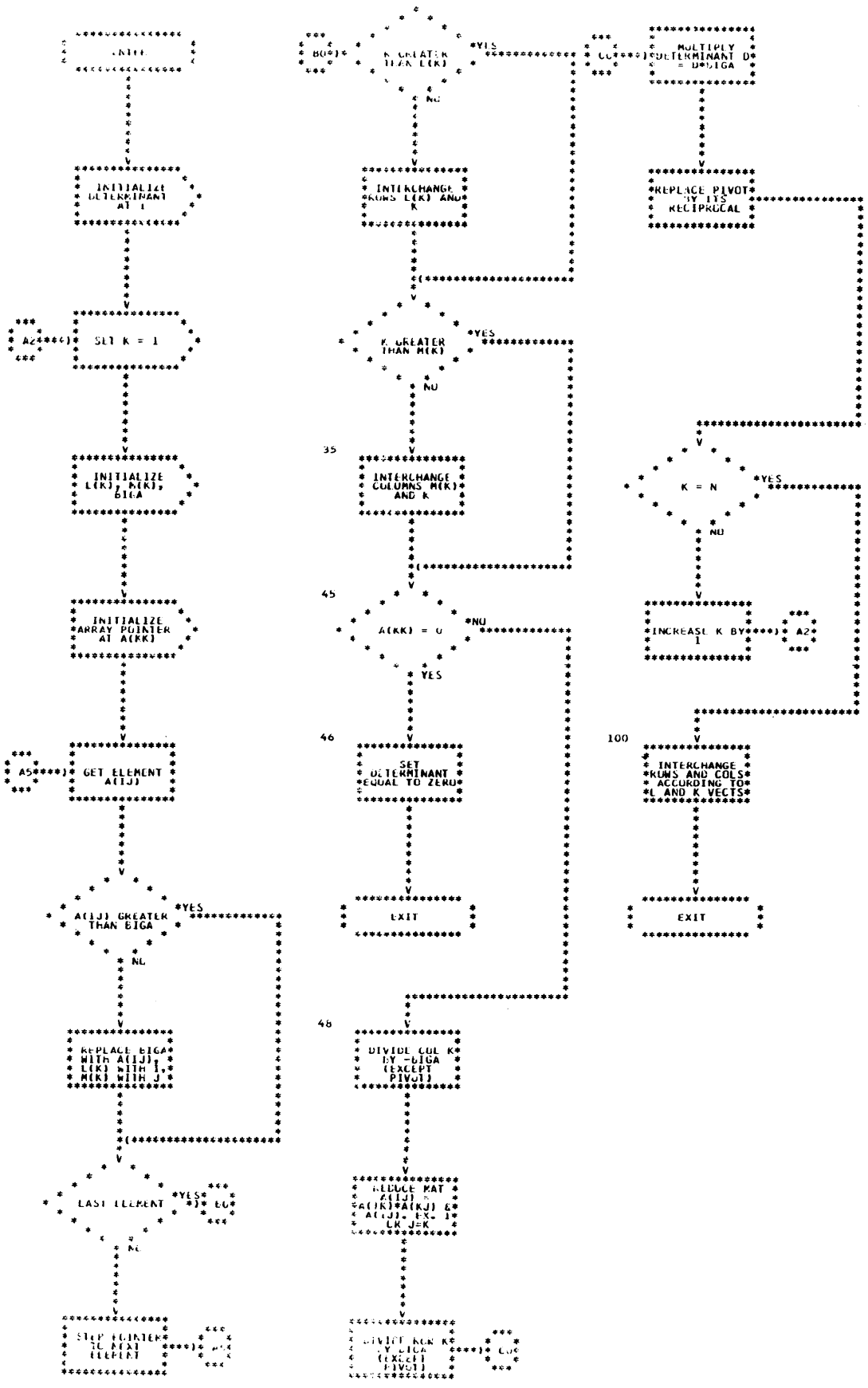

ENTER
SET STORAGE MODE AND NUMBER OF COLUMNS FOR A
INITIALIZE ARRAY POINTER AT FIRST ELEMENT
A2
GET ELEMENT
A3
2
MOVE ELEMENT TO MATRIX R
LAST ELEMENT
YES
NO
STEP POINTER TO NEXT ELEMENT
A3
BOTH INPUT MATRICES PROCESSED
NO
YES
EXIT
RESET STORAGE MODE AND NUMBER OF COLUMNS FOR B
A2

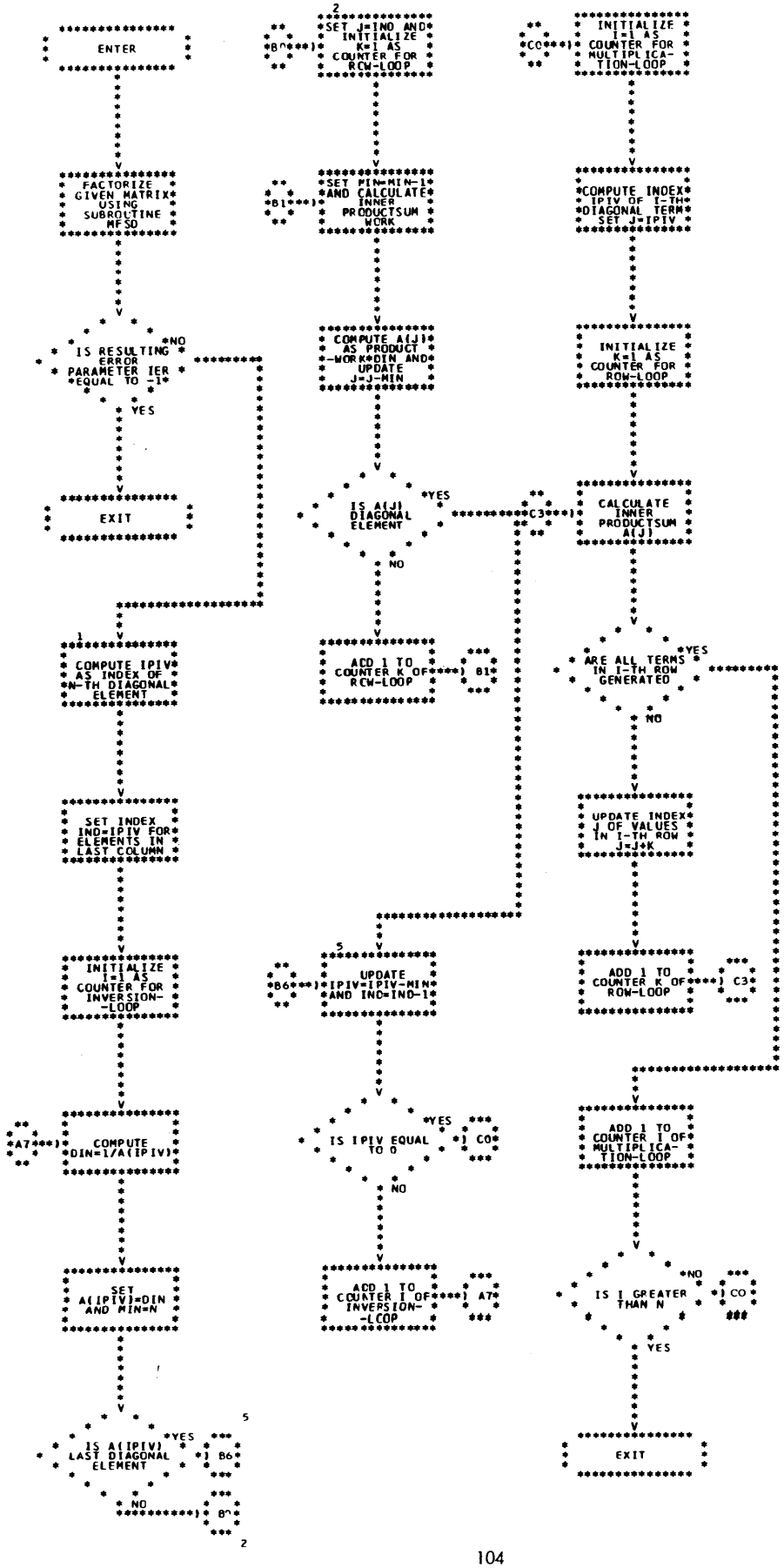


H3. ARE PERMUTED

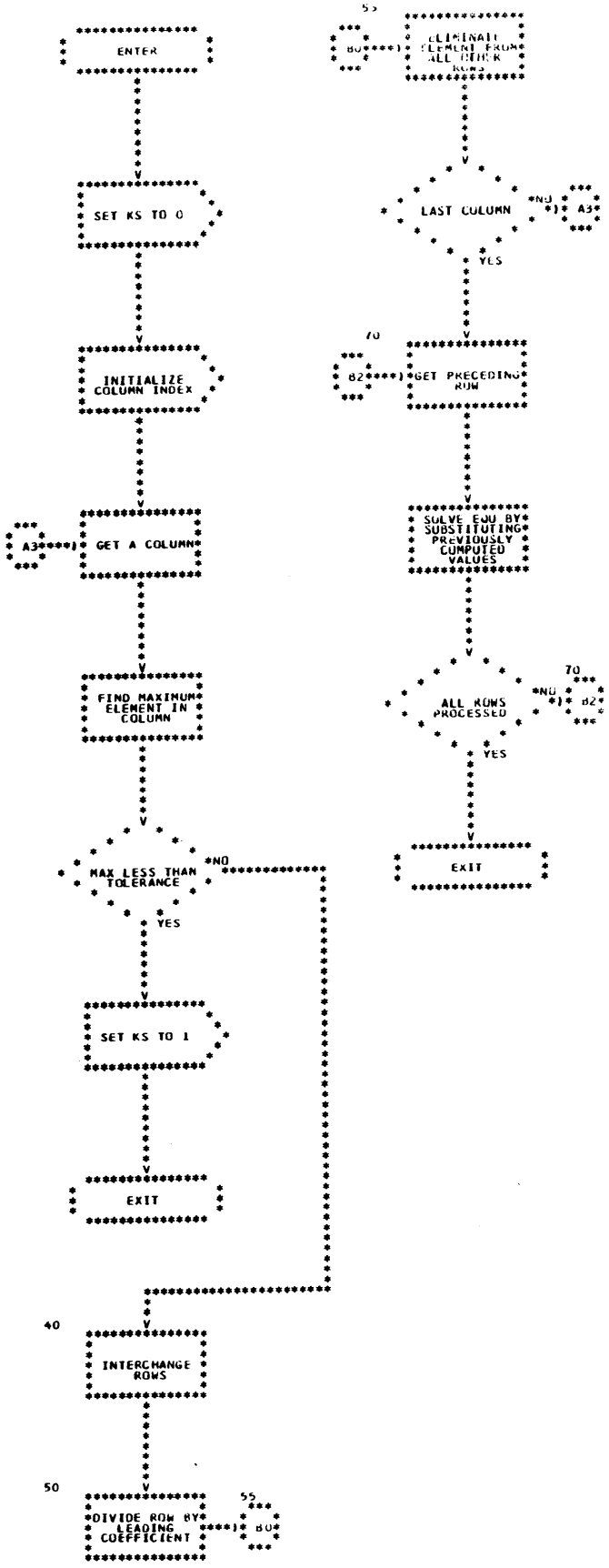

ENTER
COMPUTE VECTOR LENGTH
APPLY FUNCTION TO MATRIX ELEMENT BY ELEMENT
EXIT

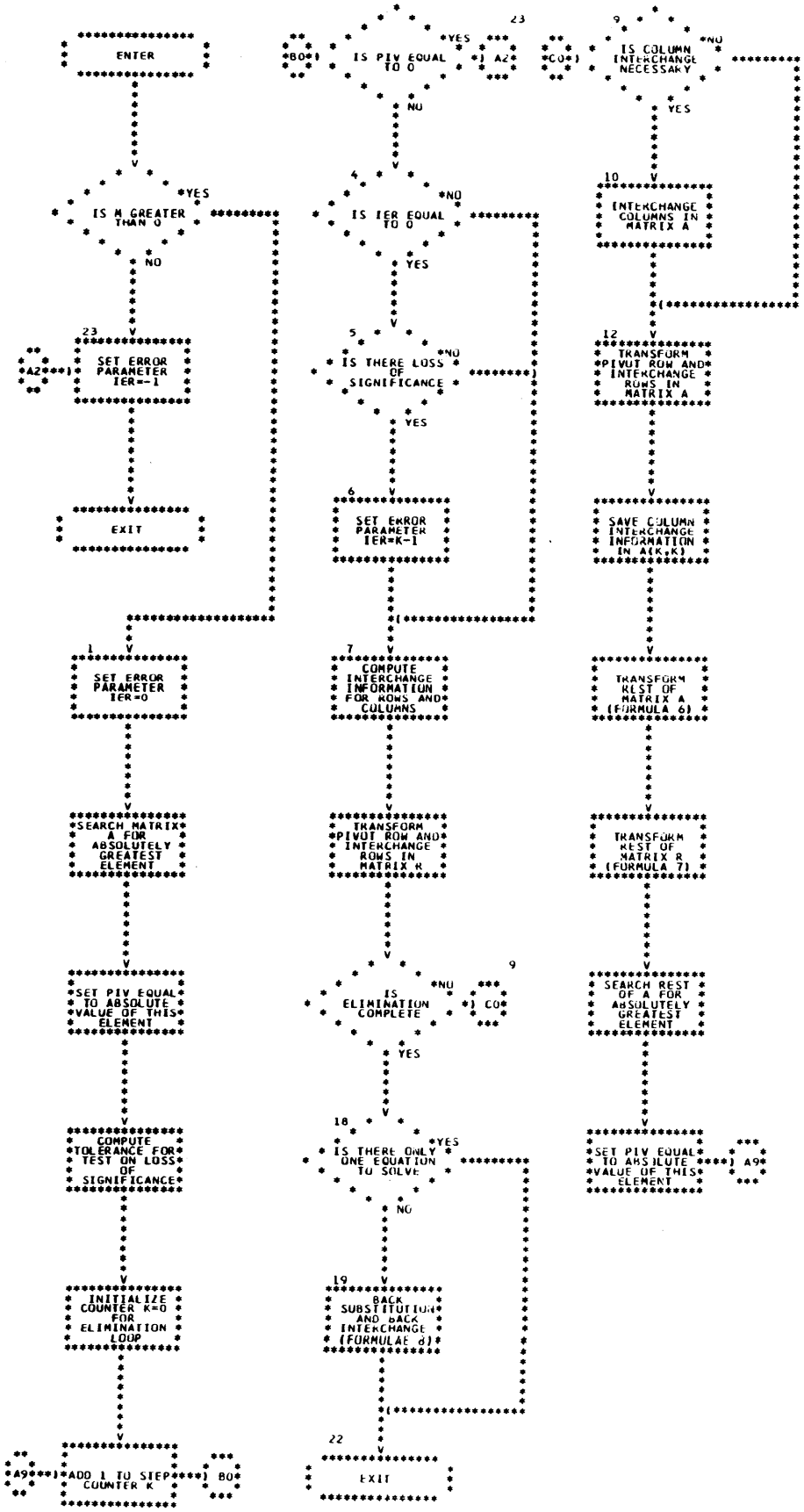

ENTER
ZERO ELEMENT
YES
NO
1
CALCULATE RECIPROCAL
2
SET EQUAL TO INFINITY
EXIT

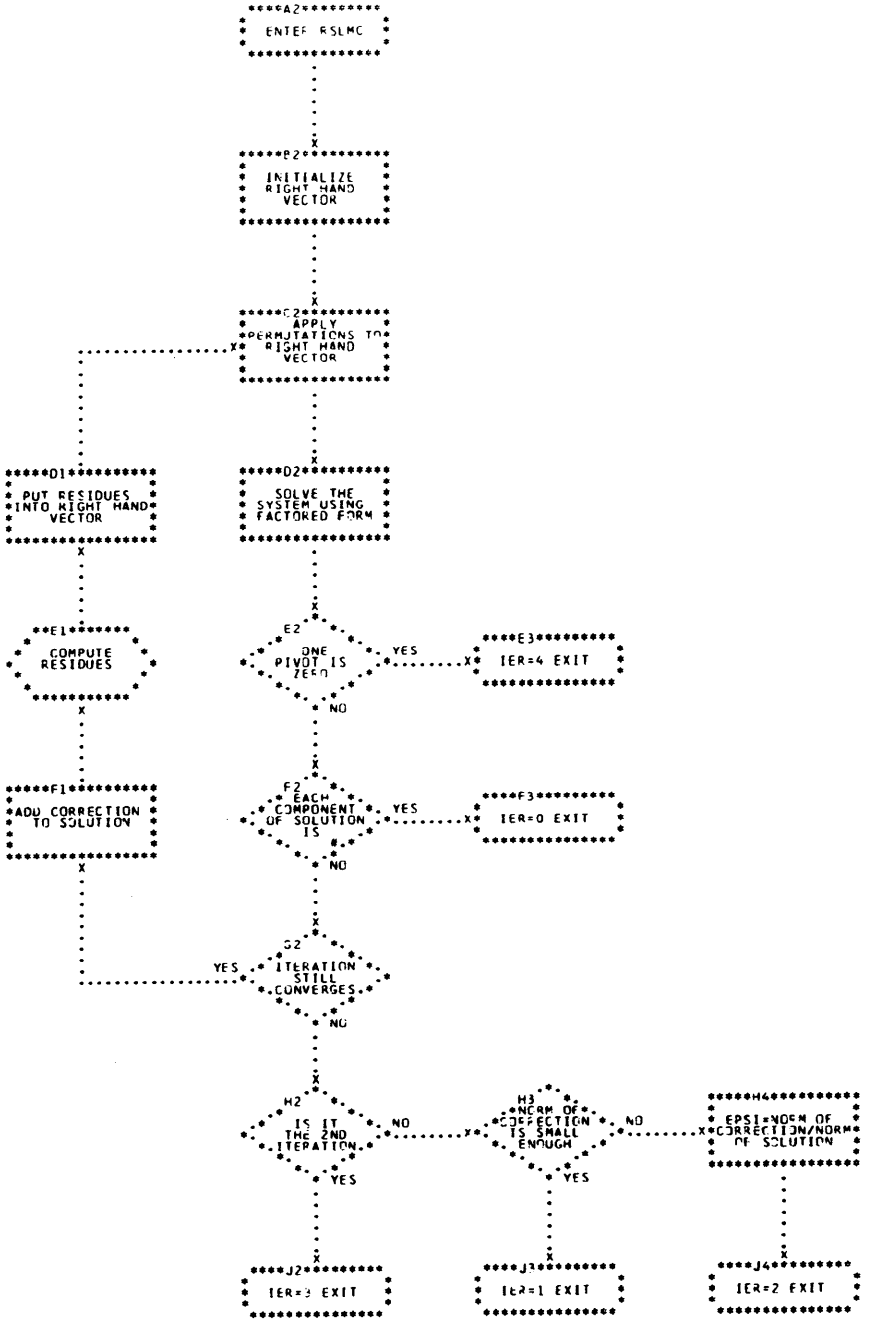

ENTER
INITIALIZE DETERMINANT AT 1
SET K = 1
INITIALIZE L(K), M(K), BIGA
INITIALIZE ARRAY POINTER AT A(KK)
GET ELEMENT A(IJ)
A(IJ) GREATER THAN BIGA
YES
NO
REPLACE BIGA WITH A(IJ), L(K) WITH I, M(K) WITH J
LAST ELEMENT
STEP POINTER TO NEXT ELEMENT
L(K) GREATER THAN K
INTERCHANGE ROWS L(K) AND K
K GREATER THAN M(K)
INTERCHANGE COLUMNS M(K) AND K
A(KK) = 0
SET DETERMINANT EQUAL TO ZERO
EXIT
DIVIDE COL K BY -BIGA (EXCEPT PIVOT)
REDUCE MAT A(IJ) = A(IJ) + A(IK)*A(KJ) EX. IK, KJ
DIVIDE ROW K BY BIGA (EXCEPT PIVOT)
MULTIPLY DETERMINANT BY BIGA
REPLACE PIVOT BY ITS RECIPROCAL
K = N
INCREASE K BY 1
INTERCHANGE ROWS AND COLS ACCORDING TO L AND K VECTS
EXIT
35
45
46
46
100


ENTER
FACTORIZE GIVEN MATRIX USING SUBROUTINE MFSD
IS RESULTING ERROR PARAMETER IER EQUAL TO -1
NO
YES
EXIT
COMPUTE IPIV AS INDEX OF N-TH DIAGONAL ELEMENT
SET INDEX IND=IPIV FOR ELEMENTS IN LAST COLUMN
INITIALIZE I=1 AS COUNTER FOR INVERSION-LOOP
COMPUTE DIN=1/A(IPIV)
SET A(IPIV)=DIN AND MIN=N
IS A(IPIV) LAST DIAGONAL ELEMENT
YES
NO
B6
B^
SET J=IND AND INITIALIZE K=1 AS COUNTER FOR ROW-LOOP
B^
SET PIV=MIN-1 AND CALCULATE INNER PRODUCTSUM WORK
B1
COMPUTE A(J) AS PRODUCT -WORK*DIN AND UPDATE J=J-MIN
IS A(J) DIAGONAL ELEMENT
YES
NO
ADD 1 TO COUNTER K OF ROW-LOOP
B1
UPDATE IPIV=IPIV-MIN AND IND=IND-1
B6
IS IPIV EQUAL TO 0
YES
NO
ADD 1 TO COUNTER I OF INVERSION--LOOP
A7
C0
INITIALIZE I=1 AS COUNTER FOR MULTIPLICA-TION-LOOP
C0
COMPUTE INDEX IPIV OF I-TH DIAGONAL TERM SET J=IPIV
INITIALIZE K=1 AS COUNTER FOR ROW-LOOP
CALCULATE INNER PRODUCTSUM A(J)
C3
ARE ALL TERMS IN I-TH ROW GENERATED
YES
NO
UPDATE INDEX J OF VALUES IN I-TH ROW J=J+K
ADD 1 TO COUNTER K OF ROW-LOOP
C3
ADD 1 TO COUNTER I OF MULTIPLICA-TION-LOOP
IS I GREATER THAN N
NO
YES
C0
EXIT

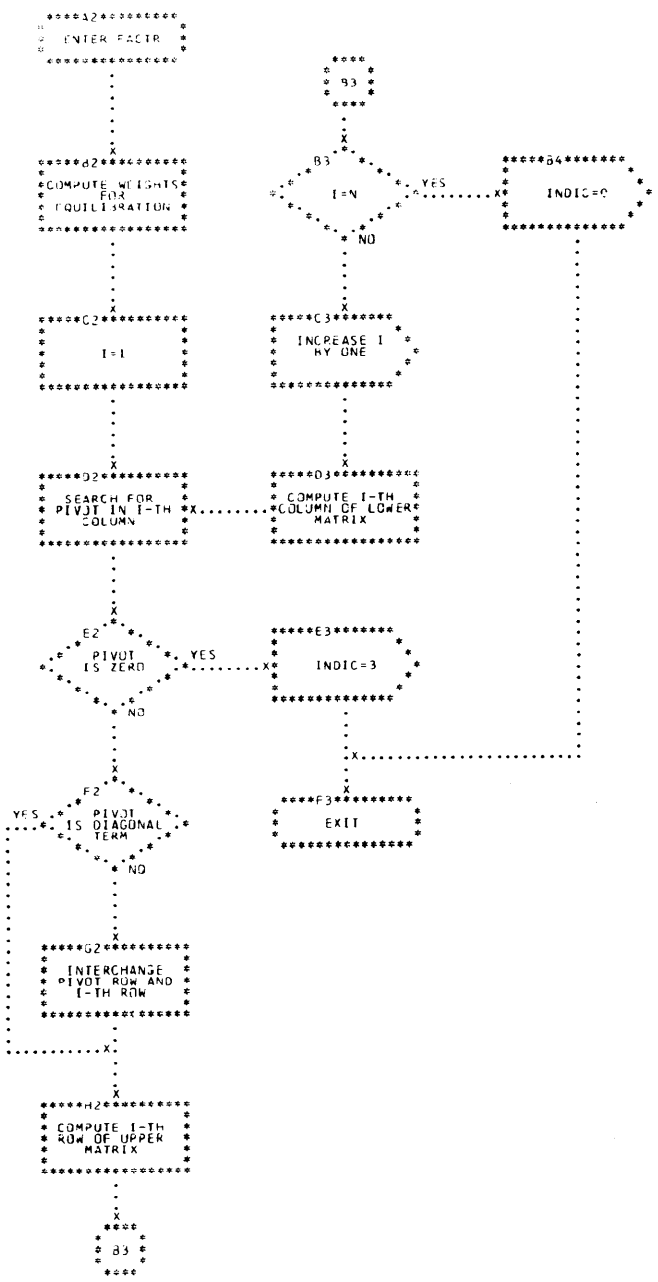
SUBROUTINE SIMQ

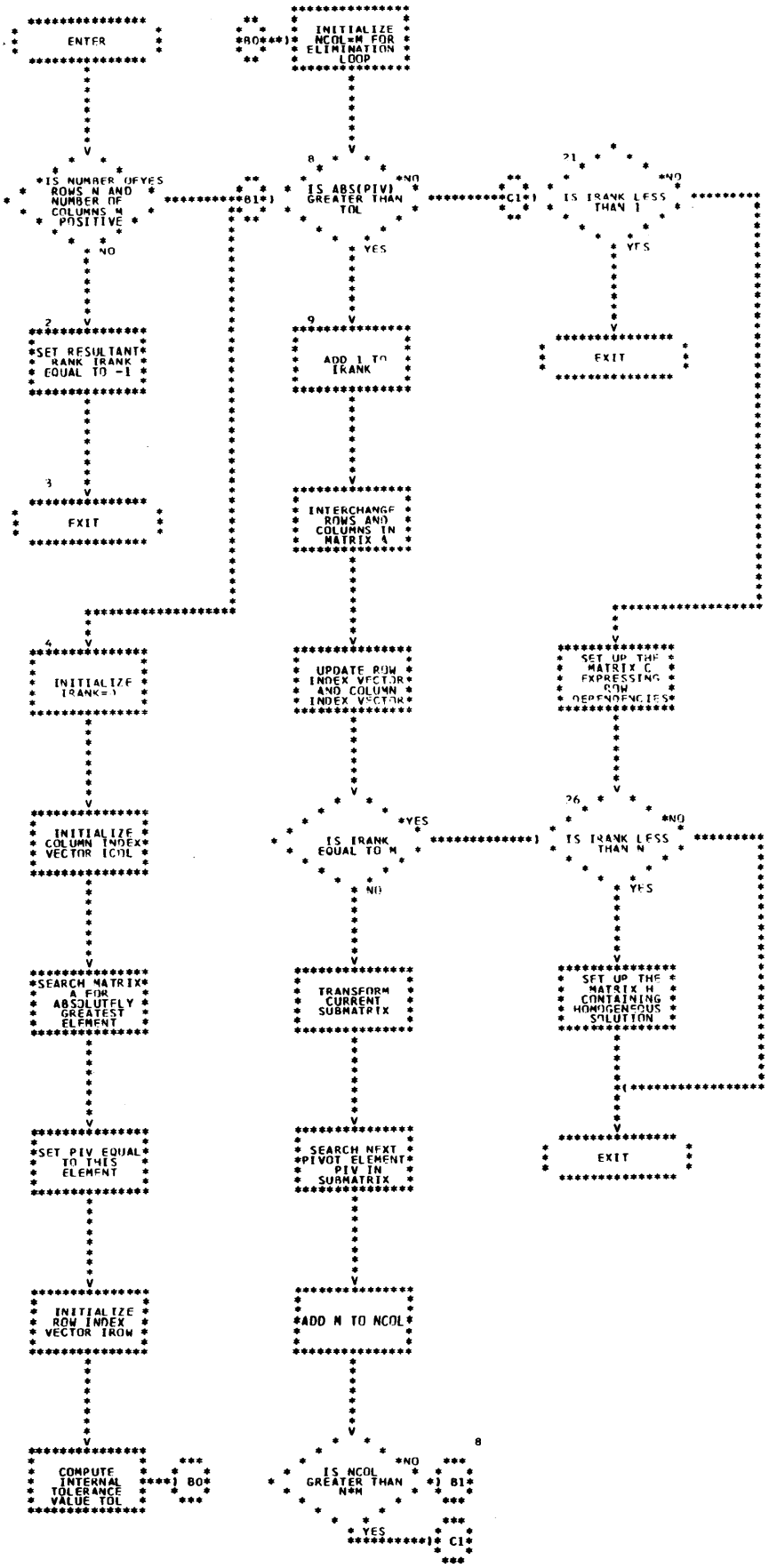


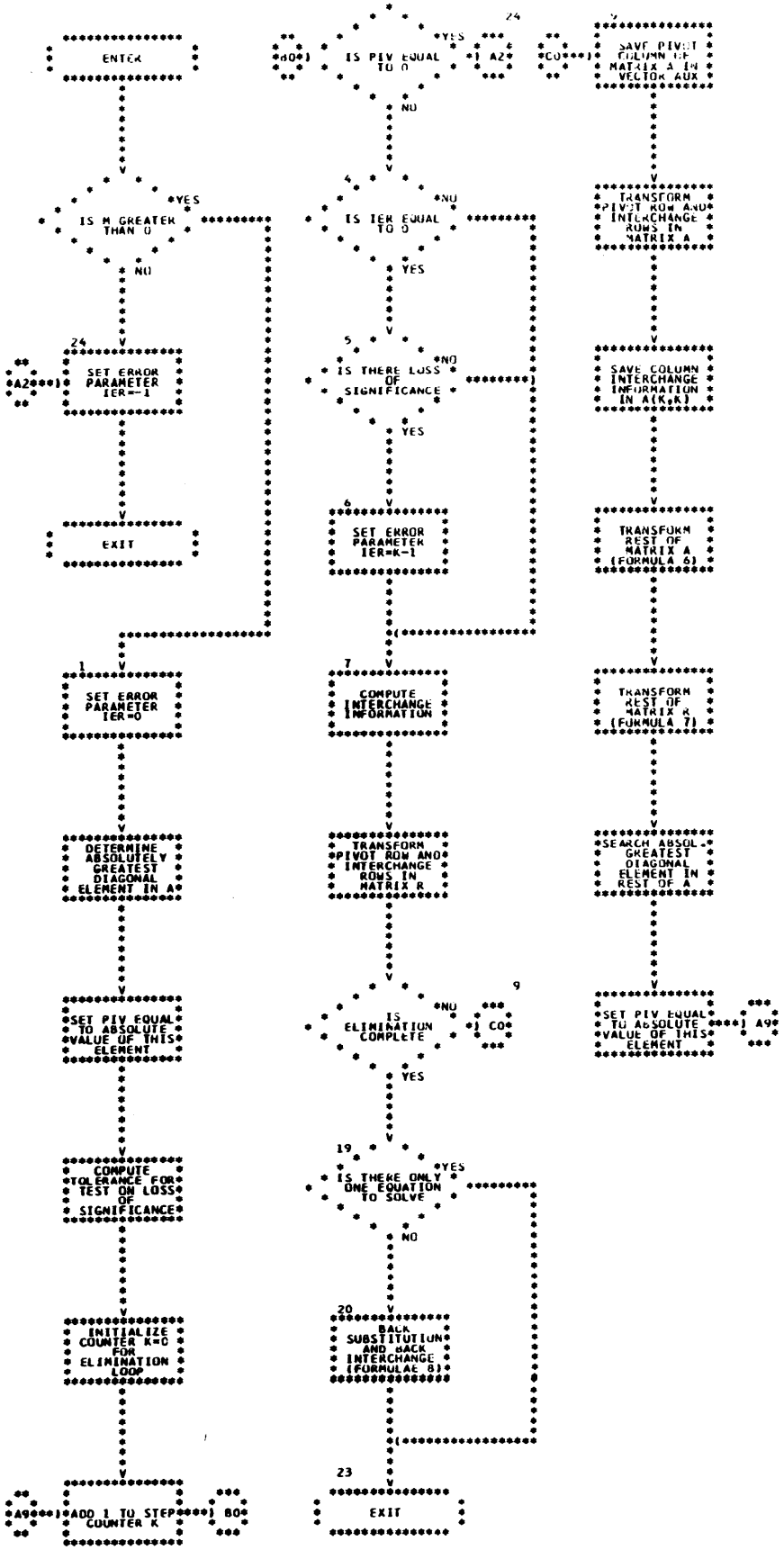

ENTER
IS M GREATER THAN 0
YES
NO
SET ERROR PARAMETER IER=-1
EXIT
SET ERROR PARAMETER IER=0
SEARCH MATRIX A FOR ABSOLUTELY GREATEST ELEMENT
SET PIV EQUAL TO ABSOLUTE VALUE OF THIS ELEMENT
COMPUTE TOLERANCE FOR TEST ON LOSS OF SIGNIFICANCE
INITIALIZE COUNTER K=0 FOR ELIMINATION LOOP
ADD 1 TO STEP COUNTER K
A9
B0
IS PIV EQUAL TO 0
A2
IS IER EQUAL TO 0
IS THERE LOSS OF SIGNIFICANCE
SET ERROR PARAMETER IER=K-1
COMPUTE INTERCHANGE INFORMATION FOR ROWS AND COLUMNS
TRANSFORM PIVOT ROW AND INTERCHANGE ROWS IN MATRIX R
IS ELIMINATION COMPLETE
C0
IS THERE ONLY ONE EQUATION TO SOLVE
BACK SUBSTITUTION AND BACK INTERCHANGE (FORMULAE 8)
EXIT
IS COLUMN INTERCHANGE NECESSARY
INTERCHANGE COLUMNS IN MATRIX A
TRANSFORM PIVOT ROW AND INTERCHANGE ROWS IN MATRIX A
SAVE COLUMN INTERCHANGE INFORMATION IN A(K,K)
TRANSFORM REST OF MATRIX A (FORMULA 6)
TRANSFORM REST OF MATRIX R (FORMULA 7)
SEARCH REST OF A FOR ABSOLUTELY GREATEST ELEMENT
SET PIV EQUAL TO ABSOLUTE VALUE OF THIS ELEMENT
A9
1
2
3
4
5
6
7
8
9
10
12
18
19
22
23

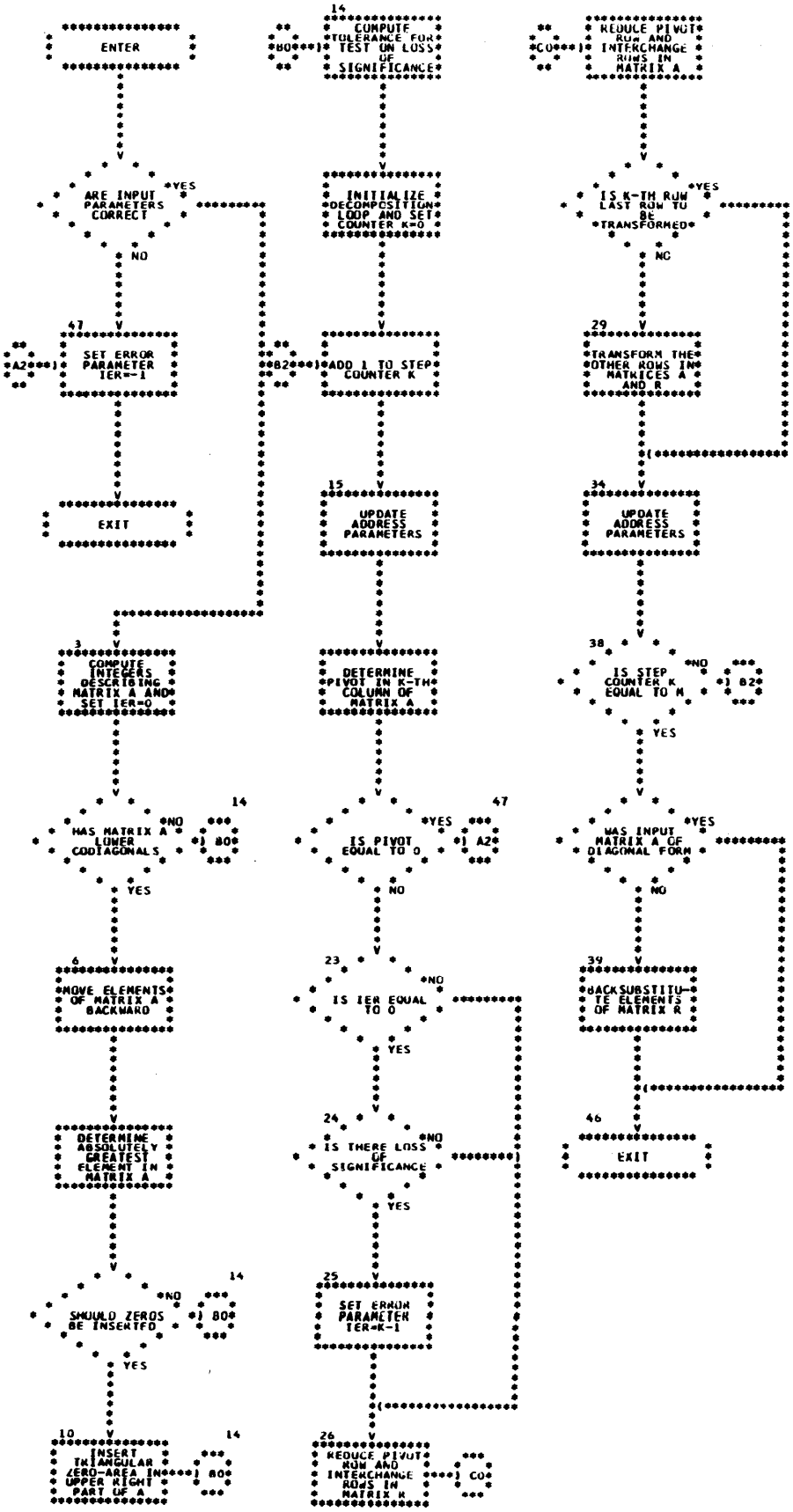


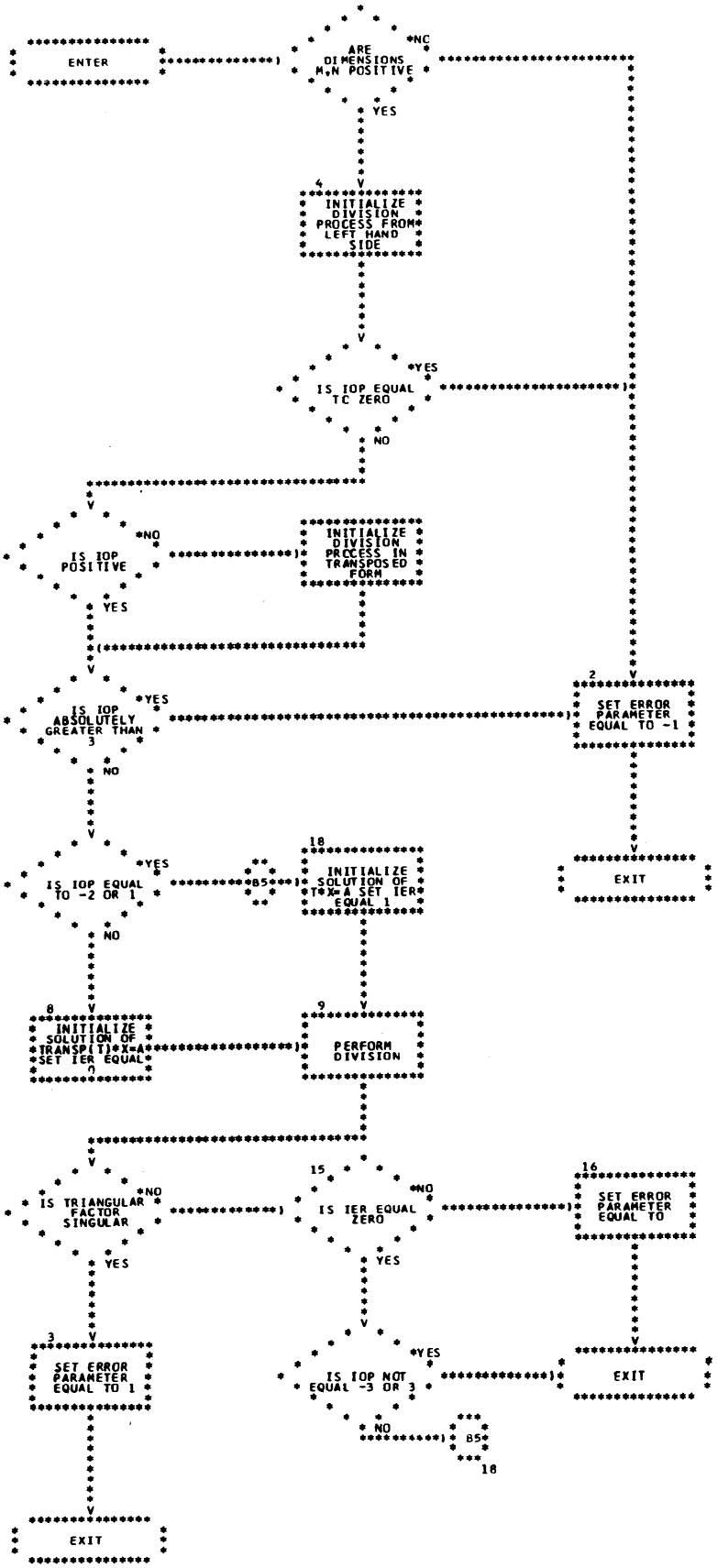
F2. ACCURATE ENOUGH

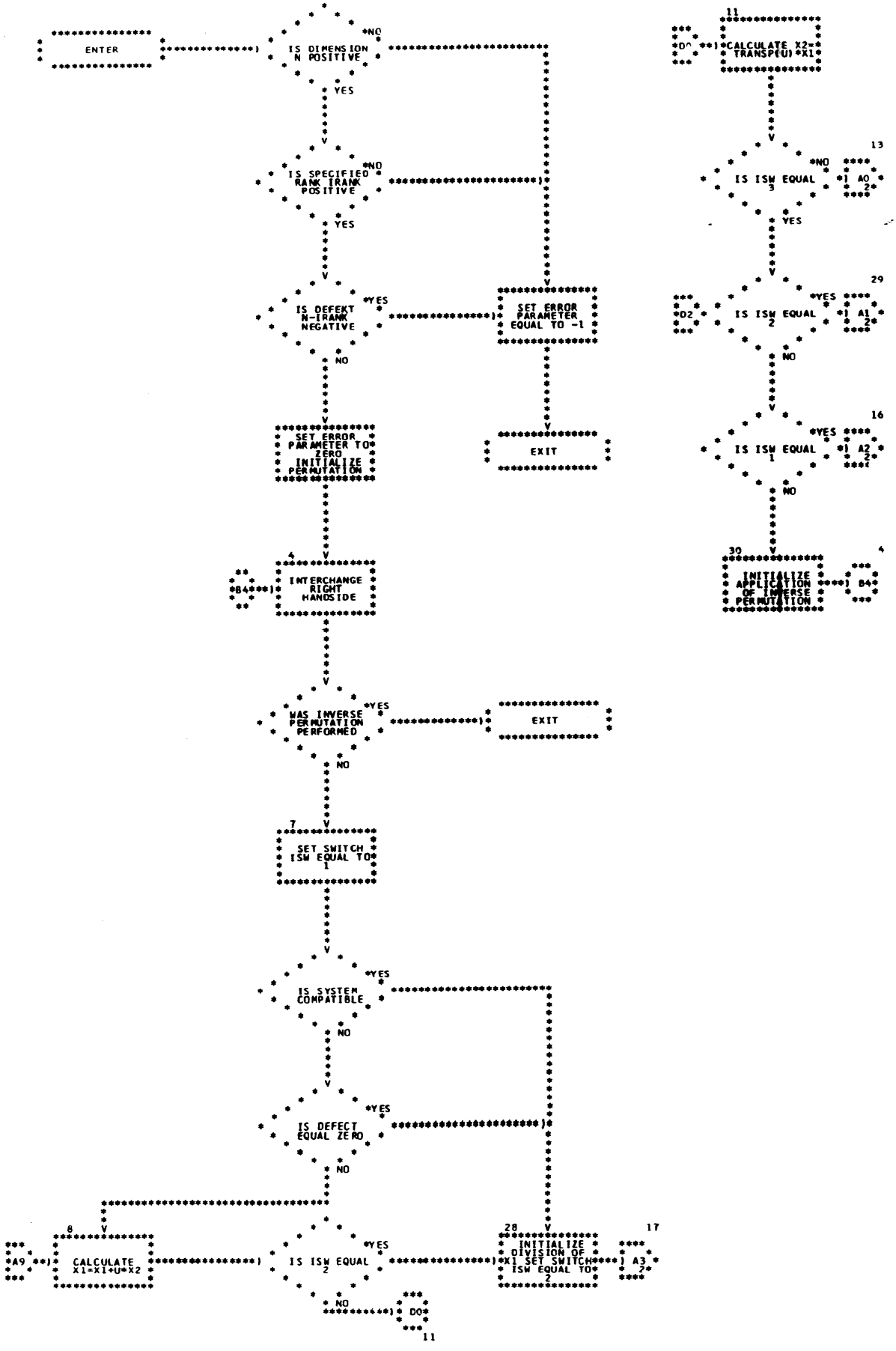

ENTER EQCTR
COMPUTE WEIGHTS FOR EQUILIBRATION
I=1
SEARCH FOR PIVOT IN I-TH COLUMN
B2
PIVOT IS ZERO
YES
NO
INTERCHANGE PIVOT ROW AND I-TH ROW
B2
PIVOT IS DIAGONAL TERM
YES
NO
COMPUTE I-TH ROW OF UPPER MATRIX
B3
B3
I=N
YES
NO
INCREASE I BY ONE
COMPUTE I-TH COLUMN OF LOWER MATRIX
INDIC=0
INDIC=3
EXIT

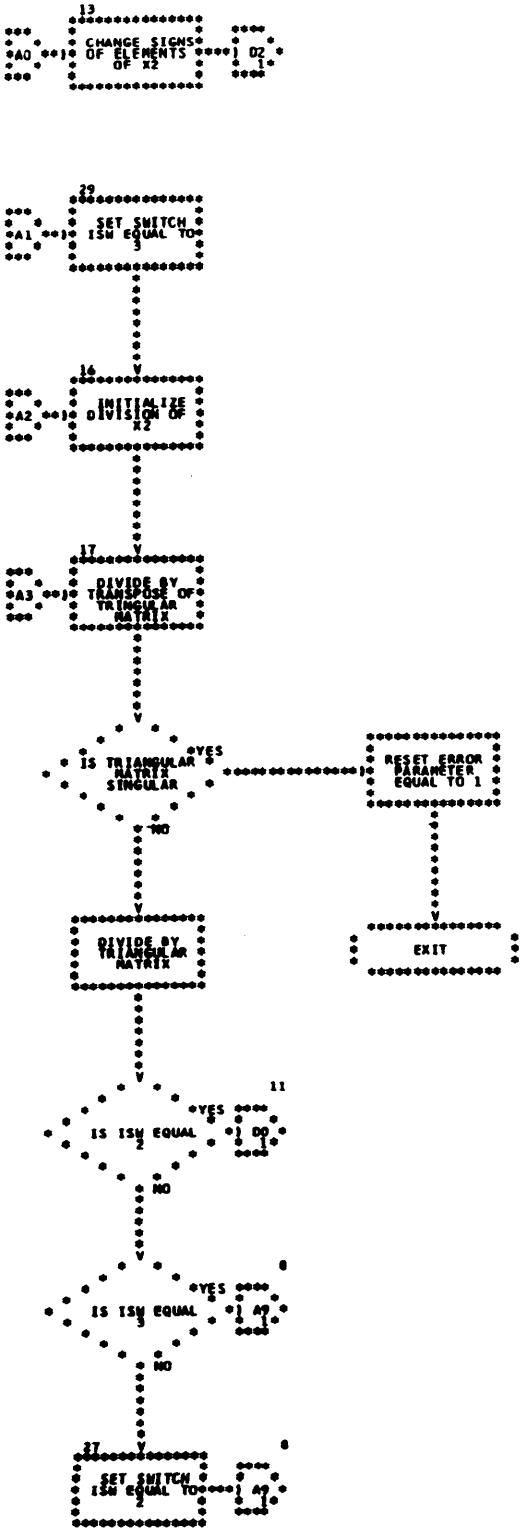

ENTER
IS NUMBER OF ROWS N AND NUMBER OF COLUMNS M POSITIVE
YES
NO
B0
SET RESULTANT RANK IRANK EQUAL TO -1
EXIT
INITIALIZE IRANK=1
INITIALIZE COLUMN INDEX VECTOR ICOL
SEARCH MATRIX A FOR ABSOLUTELY GREATEST ELEMENT
SET PIV EQUAL TO THIS ELEMENT
INITIALIZE ROW INDEX VECTOR IROW
COMPUTE INTERNAL TOLERANCE VALUE TOL
B1
INITIALIZE NCOL=M FOR ELIMINATION LOOP
IS ABS(PIV) GREATER THAN TOL
C1
ADD 1 TO IRANK
INTERCHANGE ROWS AND COLUMNS IN MATRIX A
UPDATE ROW INDEX VECTOR AND COLUMN INDEX VECTOR
IS IRANK EQUAL TO M
TRANSFORM CURRENT SUBMATRIX
SEARCH NEXT PIVOT ELEMENT PIV IN SUBMATRIX
ADD M TO NCOL
IS NCOL GREATER THAN N*M
IS IRANK LESS THAN 1
SET UP THE MATRIX C EXPRESSING ROW DEPENDENCIES
IS IRANK LESS THAN N
SET UP THE MATRIX H CONTAINING HOMOGENEOUS SOLUTION
EXIT

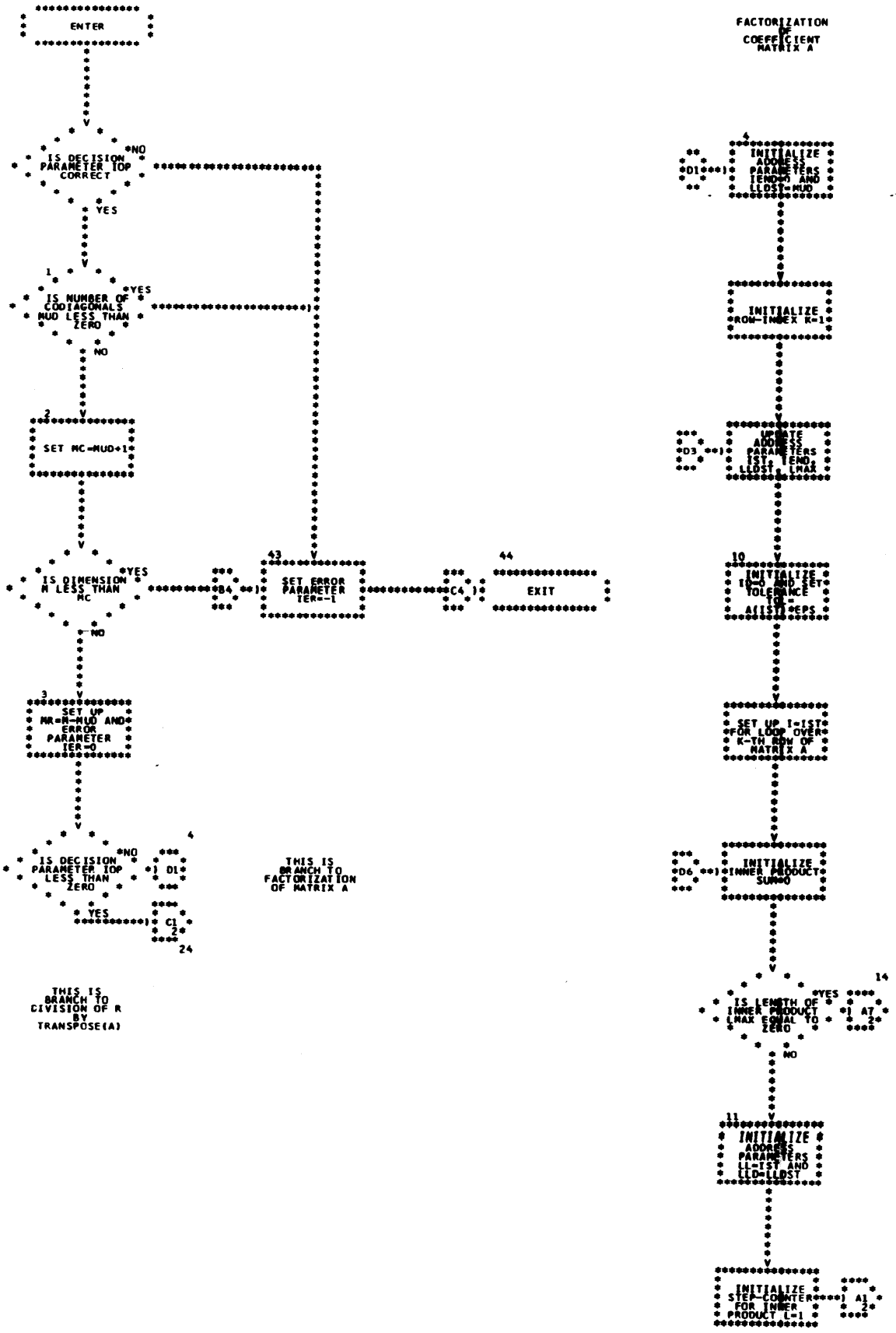

ENTER
IS M GREATER THAN 0
YES
NO
SET ERROR PARAMETER IER=-1
EXIT
SET ERROR PARAMETER IER=0
DETERMINE ABSOLUTELY GREATEST DIAGONAL ELEMENT IN A
SET PIV EQUAL TO ABSOLUTE VALUE OF THIS ELEMENT
COMPUTE TOLERANCE FOR TEST ON LOSS OF SIGNIFICANCE
INITIALIZE COUNTER K=0 FOR ELIMINATION LOOP
ADD 1 TO STEP COUNTER K
A9
B0
A2
IS PIV EQUAL TO 0
A0
C0
IS IER EQUAL TO 0
NO
YES
IS THERE LOSS OF SIGNIFICANCE
SET ERROR PARAMETER IER=K-1
COMPUTE INTERCHANGE INFORMATION
TRANSFORM PIVOT ROW AND INTERCHANGE ROWS IN MATRIX R
IS ELIMINATION COMPLETE
C0
IS THERE ONLY ONE EQUATION TO SOLVE
BACK SUBSTITUTION AND BACK INTERCHANGE (FORMULAE 8)
EXIT
SAVE PIVOT COLUMN OF MATRIX A IN VECTOR AUX
TRANSFORM PIVOT ROW AND INTERCHANGE ROWS IN MATRIX A
SAVE COLUMN INTERCHANGE INFORMATION IN A(K,K)
TRANSFORM REST OF MATRIX A (FORMULA 6)
TRANSFORM REST OF MATRIX R (FORMULA 7)
SEARCH ABSOLUTE GREATEST DIAGONAL ELEMENT IN REST OF A
SET PIV EQUAL TO ABSOLUTE VALUE OF THIS ELEMENT
A9

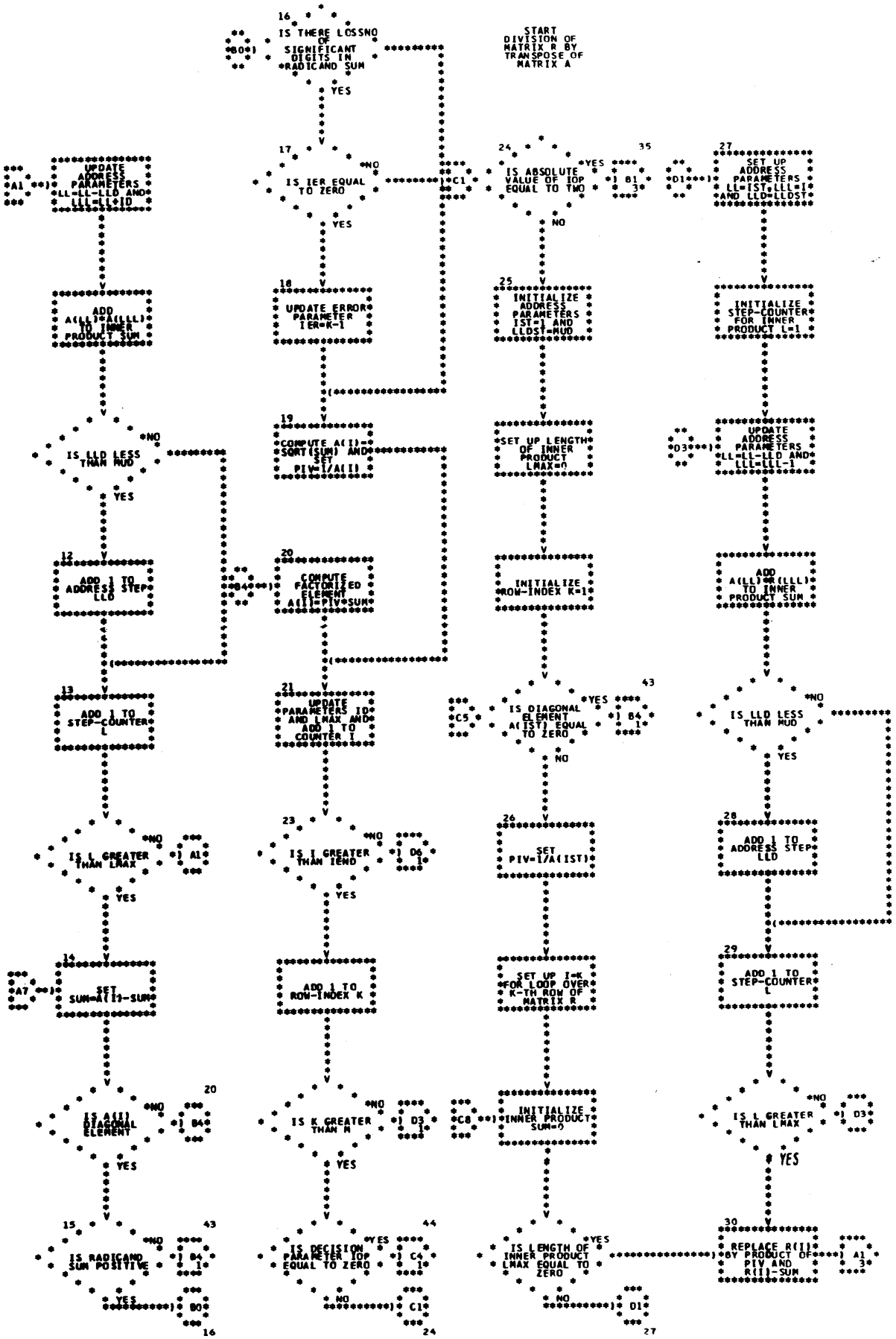

ENTER
ARE INPUT PARAMETERS CORRECT
YES
NO
SET ERROR PARAMETER IER=-1
EXIT
COMPUTE INTEGERS DESCRIBING MATRIX A AND SET IER=0
HAS MATRIX A LOWER CODIAGONALS
NO
YES
MOVE ELEMENTS OF MATRIX A BACKWARD
DETERMINE ABSOLUTELY GREATEST ELEMENT IN MATRIX A
SHOULD ZEROS BE INSERTED
INSERT TRIANGULAR ZERO-AREA IN UPPER RIGHT PART OF A
COMPUTE TOLERANCE FOR TEST ON LOSS OF SIGNIFICANCE
INITIALIZE DECOMPOSITION LOOP AND SET COUNTER K=0
ADD 1 TO STEP COUNTER K
UPDATE ADDRESS PARAMETERS
DETERMINE PIVOT IN K-TH COLUMN OF MATRIX A
IS PIVOT EQUAL TO 0
IS IER EQUAL TO 0
IS THERE LOSS OF SIGNIFICANCE
SET ERROR PARAMETER IER=K-1
REDUCE PIVOT ROW AND INTERCHANGE ROWS IN MATRIX R
IS K-TH RUW LAST RUW TO BE TRANSFORMED
TRANSFORM THE OTHER ROWS IN MATRICES A AND R
UPDATE ADDRESS PARAMETERS
IS STEP COUNTER K EQUAL TO N
WAS INPUT MATRIX A OF DIAGONAL FORM
BACKSUBSTITUTE ELEMENTS OF MATRIX R
EXIT
B0
B2
A2
C0

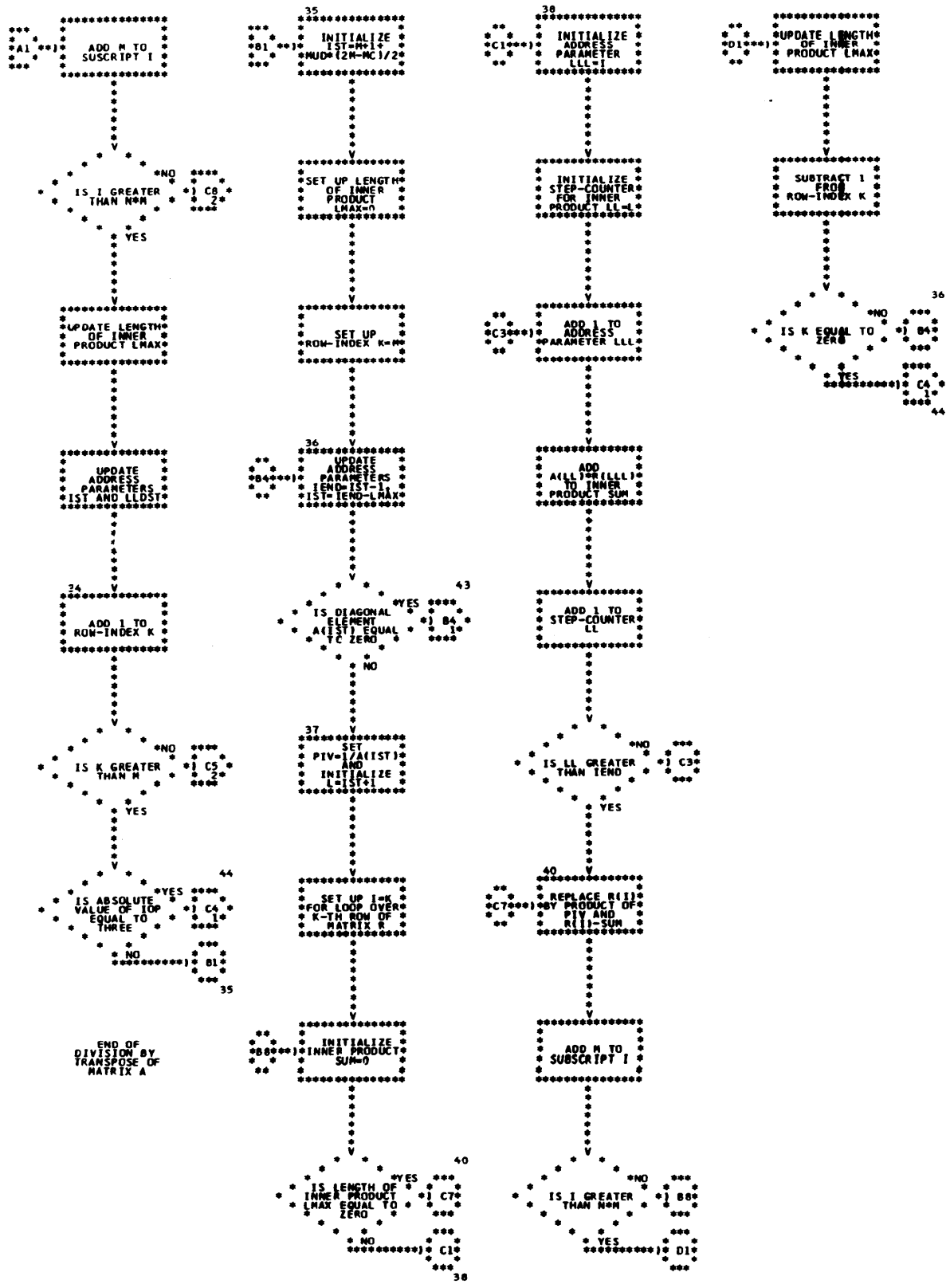

ENTER
ARE DIMENSIONS M,N POSITIVE
*NO
YES
4
INITIALIZE DIVISION PROCESS FROM LEFT HAND SIDE
IS IOP EQUAL TO ZERO
*YES
NO
IS IOP POSITIVE
*NO
INITIALIZE DIVISION PROCESS IN TRANSPOSED FORM
YES
2
SET ERROR PARAMETER EQUAL TO -1
IS IOP ABSOLUTELY GREATER THAN 3
*YES
NO
EXIT
IS IOP EQUAL TO -2 OR 1
*YES
B5
18
INITIALIZE SOLUTION OF T*X=A SET IER EQUAL 1
NO
8
INITIALIZE SOLUTION OF TRANSPIT*X=A SET IER EQUAL 0
9
PERFORM DIVISION
IS TRIANGULAR FACTOR SINGULAR
*NO
YES
15
IS IER EQUAL ZERO
*NO
16
SET ERROR PARAMETER EQUAL TO
YES
3
SET ERROR PARAMETER EQUAL TO 1
IS IOP NOT EQUAL -3 OR 3
*YES
EXIT
NO
B5
18
EXIT

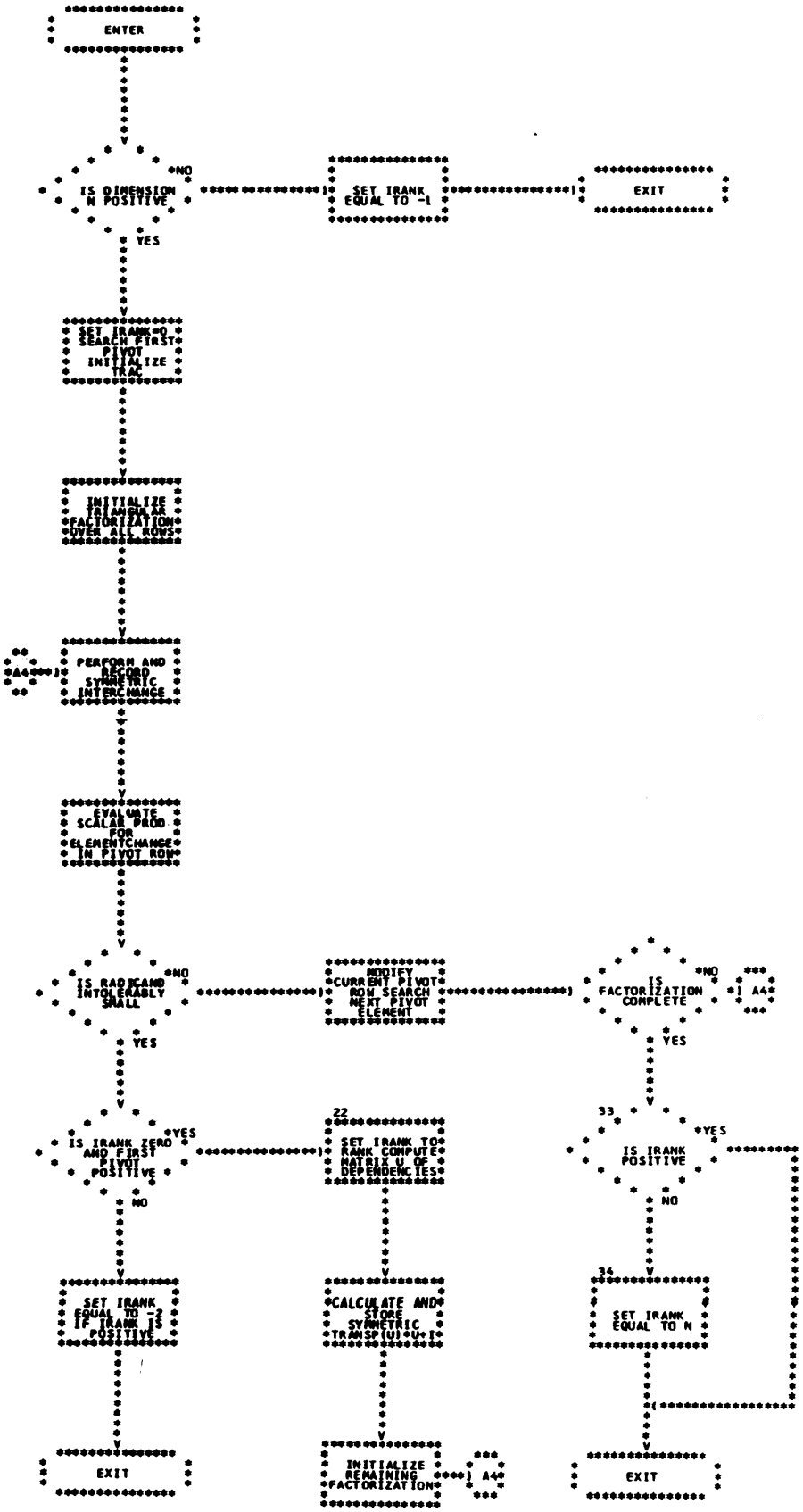

ENTER
IS DIMENSION N POSITIVE
NO
YES
IS SPECIFIED RANK IRANK POSITIVE
NO
YES
IS DEFEKT N-IRANK NEGATIVE
YES
SET ERROR PARAMETER EQUAL TO -1
NO
SET ERROR PARAMETER TO ZERO INITIALIZE PERMUTATION
EXIT
4
INTERCHANGE RIGHT HANDSIDE
B4
WAS INVERSE PERMUTATION PERFORMED
YES
EXIT
NO
7
SET SWITCH ISW EQUAL TO 1
IS SYSTEM COMPATIBLE
YES
NO
IS DEFECT EQUAL ZERO
YES
NO
8
CALCULATE X1=X1+U*X2
A9
IS ISW EQUAL 2
YES
NO
DO 11
28
INITIALIZE DIVISION OF I, SET SWITCH ISW EQUAL TO 2
17
A3 2
11
DO
CALCULATE X2=TRANSP(U)*X1
13
IS ISW EQUAL 3
NO
YES
A0 2
29
IS ISW EQUAL 2
YES
NO
D2
A1 2
16
IS ISW EQUAL 1
YES
NO
A2 2
30
INITIALIZE APPLICATION OF INVERSE PERMUTATION
4
B4

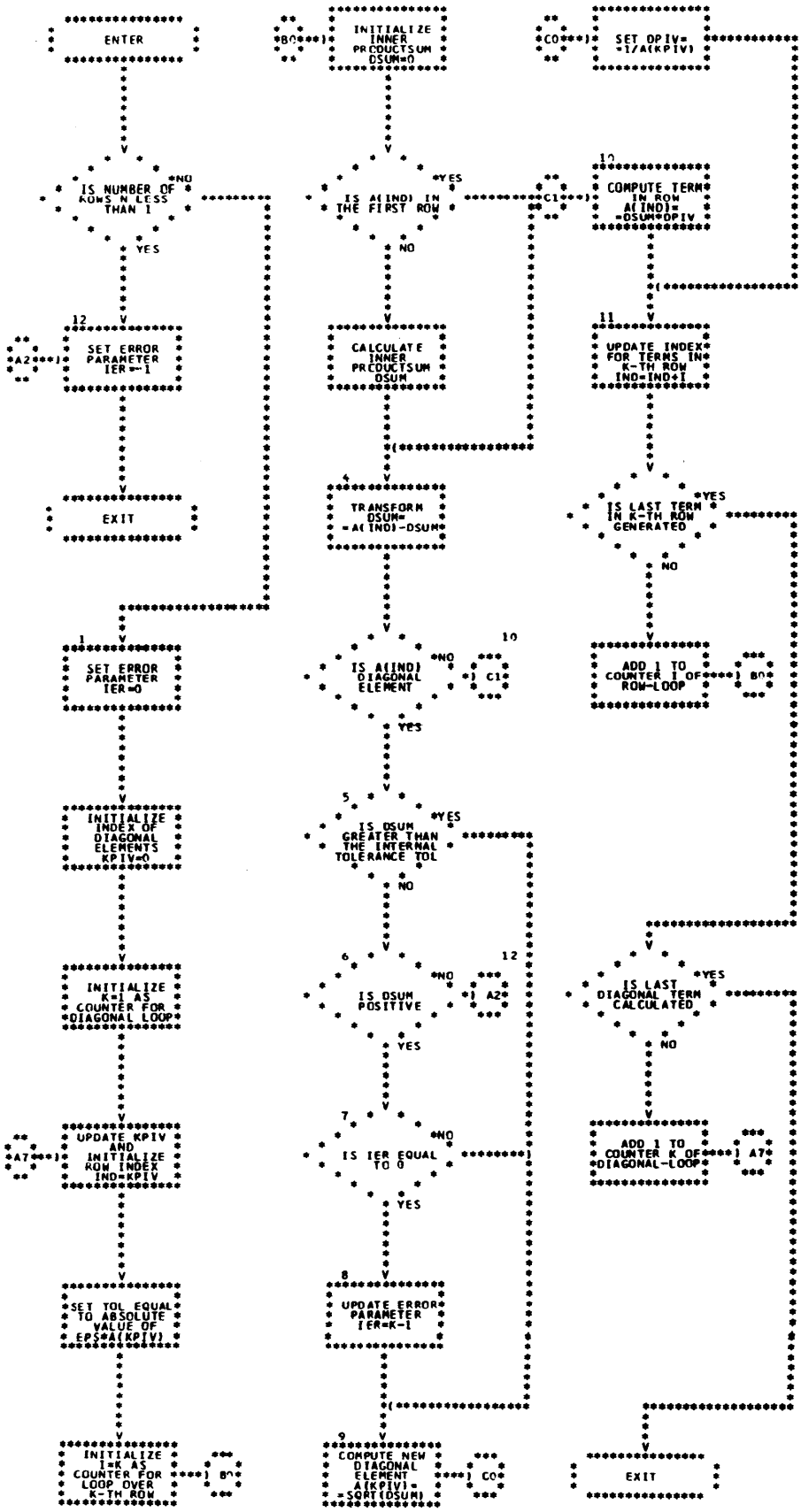

13
CHANGE SIGNS OF ELEMENTS OF X2
A0
D2
1
29
SET SWITCH ISW EQUAL TO 3
A1
16
INITIALIZE DIVISION OF X2
A2
17
DIVIDE BY TRANSPOSE OF TRIANGULAR MATRIX
A3
IS TRIANGULAR MATRIX SINGULAR
YES
NO
RESET ERROR PARAMETER EQUAL TO 1
EXIT
DIVIDE BY TRIANGULAR MATRIX
11
IS ISW EQUAL 2
YES
D0
1
NO
IS ISW EQUAL 3
YES
A9
NO
27
SET SWITCH ISW EQUAL TO 2
A9
1

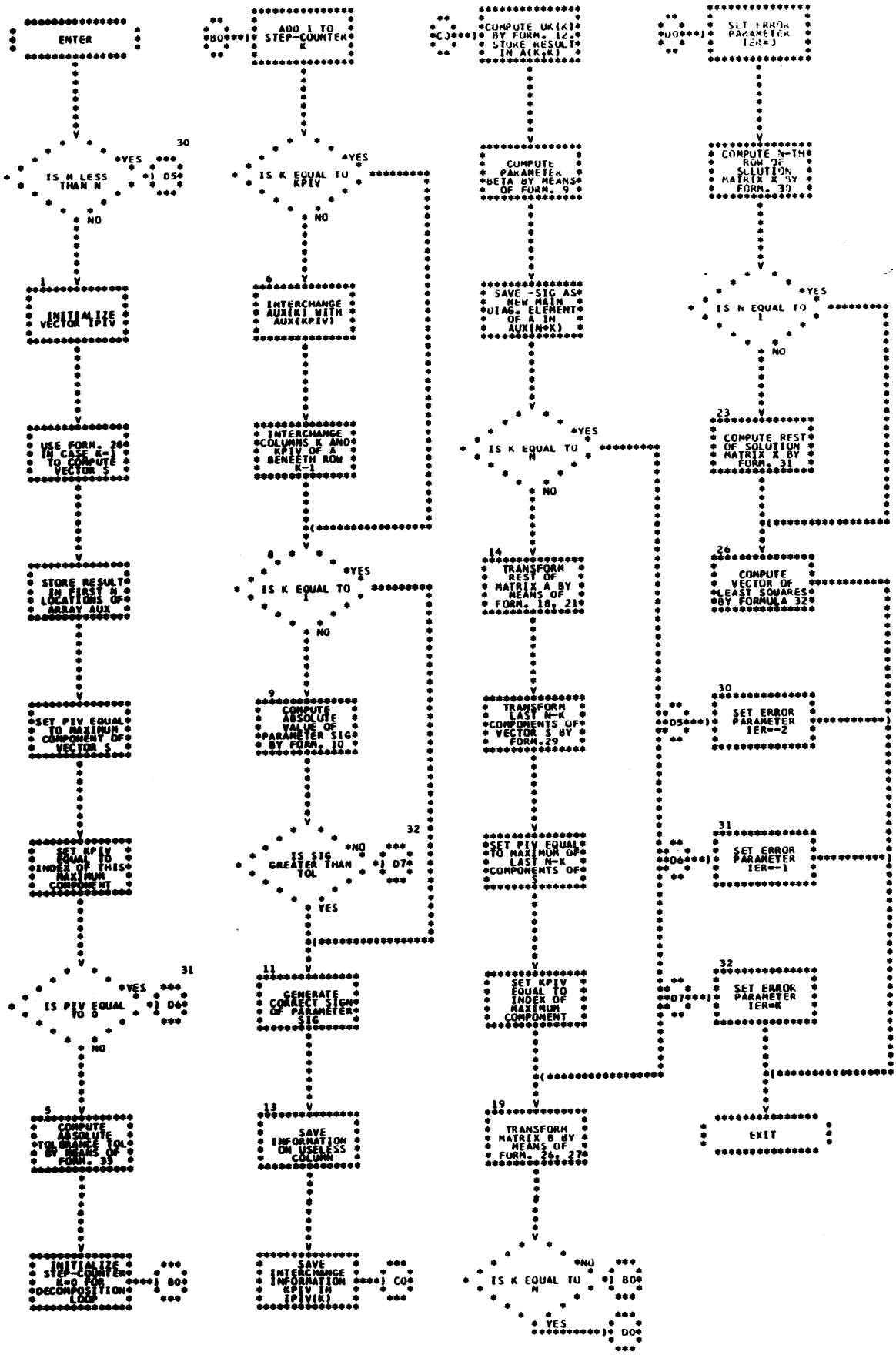

ENTER
IS DECISION PARAMETER IOP CORRECT
NO
YES
IS NUMBER OF CODIAGONALS MUD LESS THAN ZERO
YES
NO
SET MC=MUD+1
IS DIMENSION N LESS THAN MC
YES
B4
NO
SET UP MR=N-MUD AND ERROR PARAMETER IER=0
IS DECISION PARAMETER IOP LESS THAN ZERO
NO
1 D1
YES
C1 2
24
THIS IS BRANCH TO DIVISION OF R BY TRANSPOSE(A)
43
SET ERROR PARAMETER IER=-1
C4
44
EXIT
THIS IS BRANCH TO FACTORIZATION OF MATRIX A
FACTORIZATION OF COEFFICIENT MATRIX A
4
INITIALIZE ADDRESS PARAMETERS IEND=M AND LLDST=MUD
D1
INITIALIZE ROW-INDEX K=1
UPDATE ADDRESS PARAMETERS IST IEND LLDST LMAX
D3
10
INITIALIZE IO=0 AND SET TOLERANCE TOL=AIST*EPS
SET UP I-IST FOR LOOP OVER K-TH ROW OF MATRIX A
INITIALIZE INNER PRODUCT SUM=0
D6
IS LENGTH OF INNER PRODUCT LMAX EQUAL TO ZERO
YES
14
1 A1 2
NO
11
INITIALIZE ADDRESS PARAMETERS LL=IST AND LLD=LLDST
INITIALIZE STEP-COUNTER FOR INNER PRODUCT L=1
1 A1 2

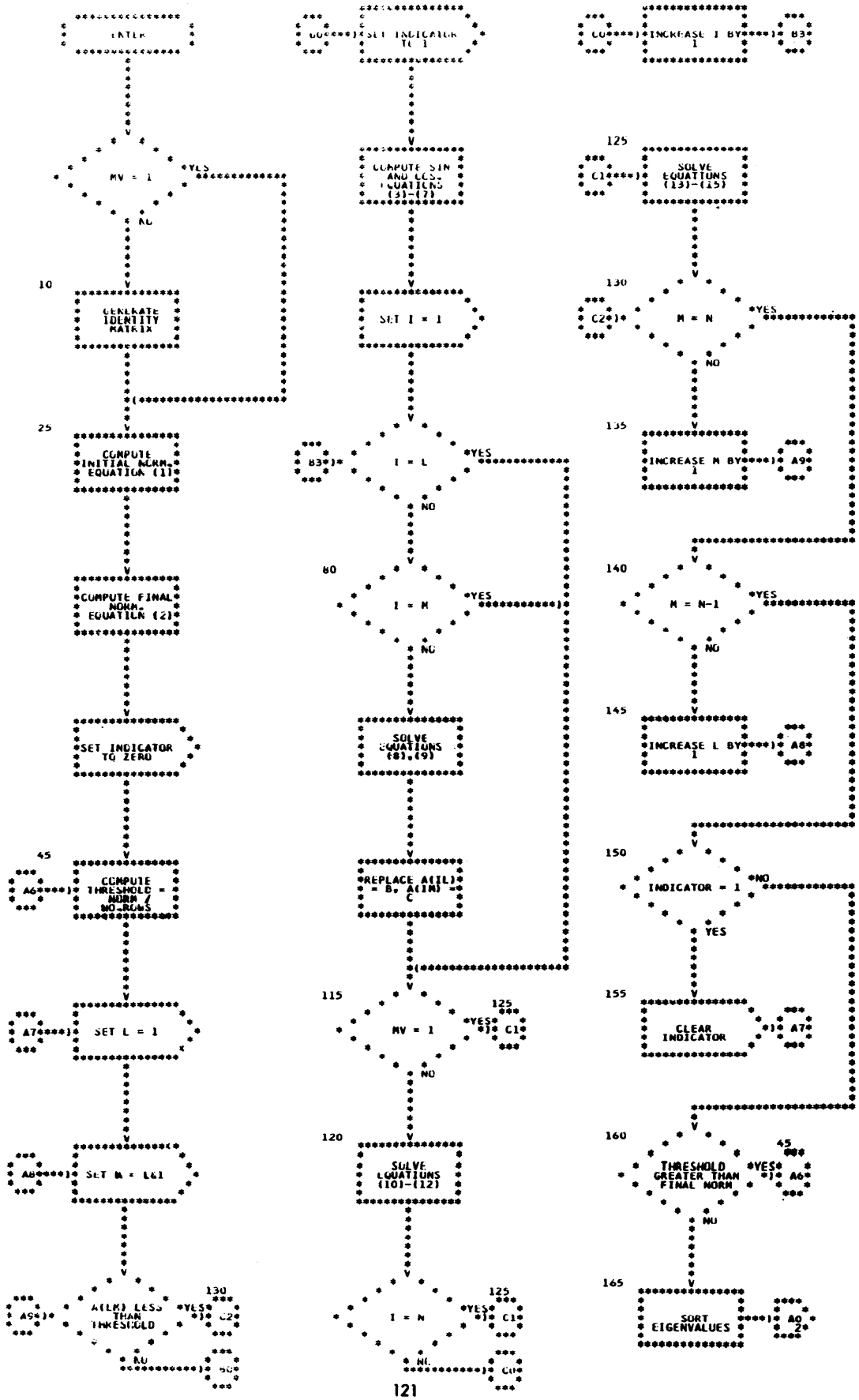

START DIVISION OF MATRIX R BY TRANSPOSE OF MATRIX A
UPDATE ADDRESS PARAMETERS LL=LL-LLD AND LLL=LL+ID
ADD A(LL)*A(LLL) TO INNER PRODUCT SUM
IS LLD LESS THAN MUD
ADD 1 TO ADDRESS STEP LLD
ADD 1 TO STEP-COUNTER L
IS L GREATER THAN LMAX
SET SUM=A(I)-SUM
IS A(I) DIAGONAL ELEMENT
IS RADICAND SUM POSITIVE
IS THERE LOSSNO OF SIGNIFICANT DIGITS IN RADICAND SUM
IS IER EQUAL TO ZERO
UPDATE ERROR PARAMETER IER=K-1
COMPUTE A(I)=SQRT(SUM) AND SET PIV=1/A(I)
COMPUTE FACTORIZED ELEMENT A(I)=PIV*SUM
UPDATE PARAMETERS ID AND LMAX AND ADD 1 TO COUNTER I
IS I GREATER THAN IEND
ADD 1 TO ROW-INDEX K
IS K GREATER THAN N
IS DECISION PARAMETER IDP EQUAL TO ZERO
IS ABSOLUTE VALUE OF IDP EQUAL TO TWO
INITIALIZE ADDRESS PARAMETERS IST=1 AND LLDST=MUD
SET UP LENGTH OF INNER PRODUCT LMAX=N
INITIALIZE ROW-INDEX K=1
IS DIAGONAL ELEMENT A(IST) EQUAL TO ZERO
SET PIV=1/A(IST)
SET UP I=K FOR LOOP OVER K-TH ROW OF MATRIX R
INITIALIZE INNER PRODUCT SUM=0
IS LENGTH OF INNER PRODUCT LMAX EQUAL TO ZERO
SET UP ADDRESS PARAMETERS LL=IST+LLL+1 AND LLD=LLDST
INITIALIZE STEP-COUNTER FOR INNER PRODUCT L=1
UPDATE ADDRESS PARAMETERS LL=LL-LLD AND LLL=LLL-1
ADD A(LL)*A(LLL) TO INNER PRODUCT SUM
IS LLD LESS THAN MUD
ADD 1 TO ADDRESS STEP LLD
ADD 1 TO STEP-COUNTER L
IS L GREATER THAN LMAX
REPLACE R(I,J) BY PRODUCT OF PIV AND R(I,J)-SUM
YES
NO

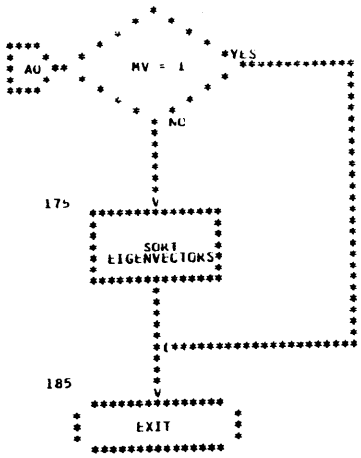

START DIVISION OF MATRIX R BY MATRIX A
A1
ADD M TO SUSCRIPT I
IS I GREATER THAN N*M
NO
C8
2
YES
UPDATE LENGTH OF INNER PRODUCT LMAX
UPDATE ADDRESS PARAMETERS IST AND LLDST
24
ADD 1 TO ROW-INDEX K
IS K GREATER THAN M
NO
C5
2
YES
44
IS ABSOLUTE VALUE OF IOP EQUAL TO THREE
YES
C4
1
NO
B1
35
END OF DIVISION BY TRANSPOSE OF MATRIX A
35
B1
INITIALIZE IST=M+1 *NUD=(2M-MC)/2*
SET UP LENGTH OF INNER PRODUCT LMAX=0
SET UP ROW-INDEX K=M
36
B4
UPDATE ADDRESS PARAMETERS IEND=IST-1, IST=IEND-LMAX
43
IS DIAGONAL ELEMENT A(IST) EQUAL TO ZERO
YES
B4
1
NO
37
SET PIV=1/A(IST) AND INITIALIZE L=IST+1
SET UP I=K FOR LOOP OVER K-TH ROW OF MATRIX R
B8
INITIALIZE INNER PRODUCT SUM=0
40
IS LENGTH OF INNER PRODUCT LMAX EQUAL TO ZERO
YES
C7
1
NO
C1
38
C1
INITIALIZE ADDRESS PARAMETER LLL=I
INITIALIZE STEP-COUNTER FOR INNER PRODUCT LL=L
C3
ADD 1 TO ADDRESS PARAMETER LLL
ADD A(LL)*R(LLL) TO INNER PRODUCT SUM
ADD 1 TO STEP-COUNTER LL
IS LL GREATER THAN IEND
NO
C3
1
YES
40
C7
REPLACE R(I) BY PRODUCT OF PIV AND R(I)-SUM
ADD M TO SUBSCRIPT I
IS I GREATER THAN N*M
NO
B8
1
YES
D1
D1
UPDATE LENGTH OF INNER PRODUCT LMAX
SUBTRACT 1 FROM ROW-INDEX K
36
IS K EQUAL TO ZERO
NO
B4
1
YES
C4
1
44

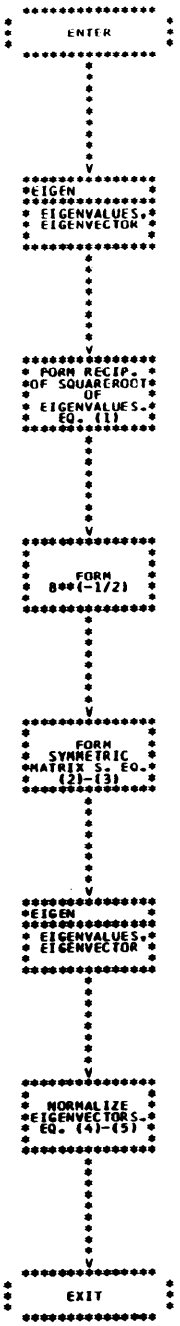

ENTER
IS DIMENSION N POSITIVE
NO
SET IRANK EQUAL TO -1
EXIT
YES
SET IRANK=0 SEARCH FIRST PIVOT INITIALIZE TRAC
INITIALIZE TRIANGULAR FACTORIZATION OVER ALL ROWS
A4
PERFORM AND RECORD SYMMETRIC INTERCHANGE
EVALUATE SCALAR PROD FOR ELEMENTCHANGE IN PIVOT ROW
IS RADICAND INTOLERABLY SMALL
NO
MODIFY CURRENT PIVOT ROW SEARCH NEXT PIVOT ELEMENT
IS FACTORIZATION COMPLETE
NO
A4
YES
YES
IS IRANK ZERO AND FIRST PIVOT POSITIVE
YES
NO
22
SET IRANK TO RANK COMPUTE MATRIX U OF DEPENDENCIES
33
IS IRANK POSITIVE
YES
NO
SET IRANK EQUAL TO -2 IF IRANK IS POSITIVE
CALCULATE AND STORE SYMMETRIC TRANSP(U)*U-I
34
SET IRANK EQUAL TO N
EXIT
INITIALIZE REMAINING FACTORIZATION
A4
EXIT

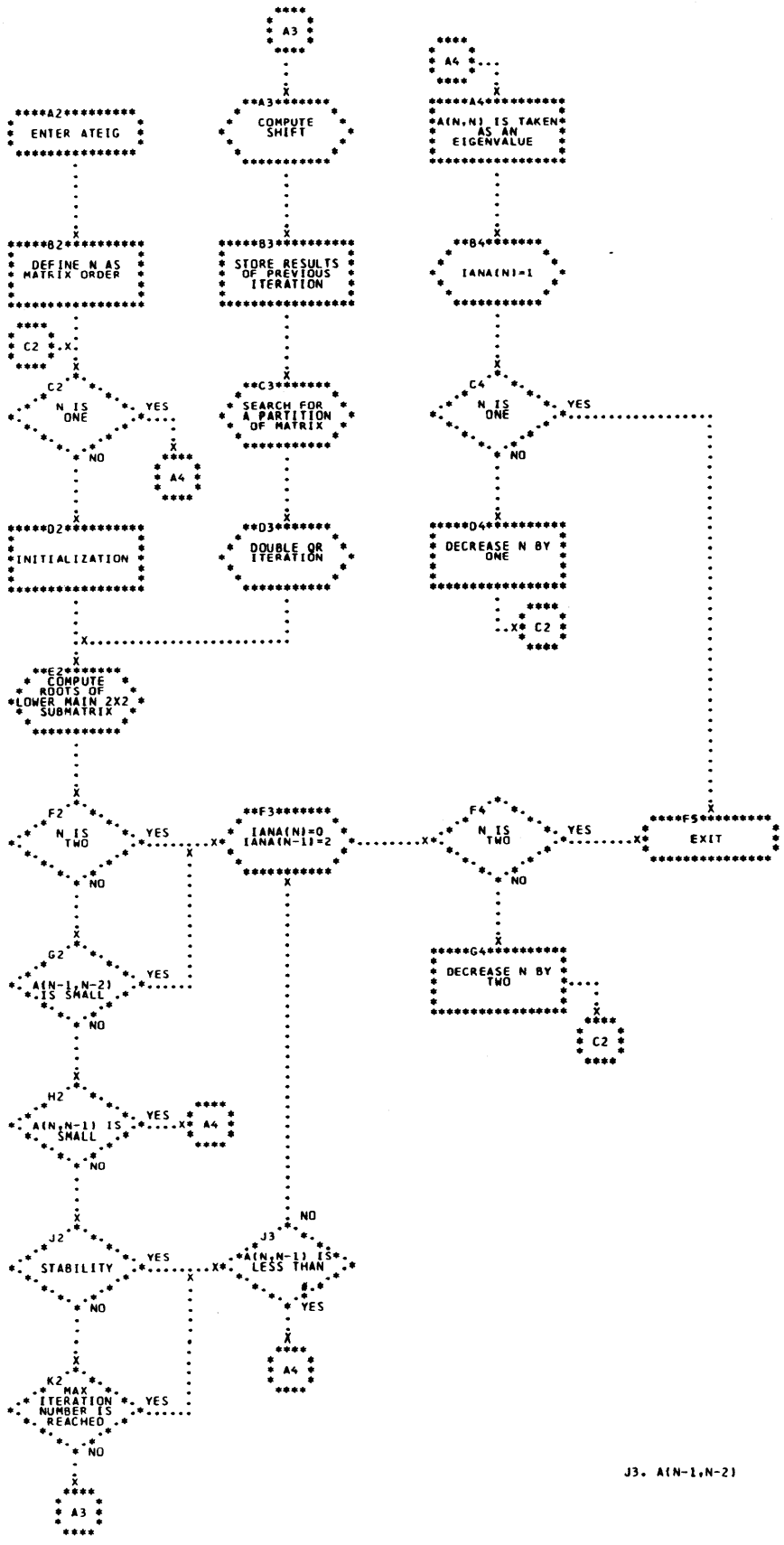

ENTER
IS NUMBER OF ROWS N LESS THAN 1
*NO
YES
SET ERROR PARAMETER IER=-1
EXIT
SET ERROR PARAMETER IER=0
INITIALIZE INDEX OF DIAGONAL ELEMENTS KPIV=0
INITIALIZE K=1 AS COUNTER FOR DIAGONAL LOOP
UPDATE KPIV AND INITIALIZE ROW INDEX IND=KPIV
SET TOL EQUAL TO ABSOLUTE VALUE OF EPS*A(KPIV)
INITIALIZE I=K AS COUNTER FOR LOOP OVER K-TH ROW
B0
INITIALIZE INNER PRODUCTSUM DSUM=0
IS A(IND) IN THE FIRST ROW
*YES
NO
C1
CALCULATE INNER PRODUCTSUM DSUM
TRANSFORM DSUM= =A(IND)-DSUM*
IS A(IND) DIAGONAL ELEMENT
*NO
YES
C1
IS DSUM GREATER THAN THE INTERNAL TOLERANCE TOL
*YES
NO
IS DSUM POSITIVE
*NO
YES
A2
IS IER EQUAL TO 0
*NO
YES
UPDATE ERROR PARAMETER IER=K-1
COMPUTE NEW DIAGONAL ELEMENT A(KPIV)= =SQRT(DSUM)
C0
SET DPIV= =1/A(KPIV)
COMPUTE TERM IN ROW A(IND)= =DSUM*DPIV
UPDATE INDEX FOR TERMS IN K-TH ROW IND=IND+I
IS LAST TERM IN K-TH ROW GENERATED
*YES
NO
ADD 1 TO COUNTER I OF ROW-LOOP
B0
IS LAST DIAGONAL TERM CALCULATED
*YES
NO
ADD 1 TO COUNTER K OF DIAGONAL-LOOP
A7
EXIT


ENTER
IS M LESS THAN N
YES
NO
D5
30
INITIALIZE VECTOR IPIV
1
USE FORM. 28 IN CASE K=1 TO COMPUTE VECTOR S
STORE RESULT IN FIRST M LOCATIONS OF ARRAY AUX
SET PIV EQUAL TO MAXIMUM COMPONENT OF VECTOR S
SET KPIV EQUAL TO INDEX OF THIS MAXIMUM COMPONENT
IS PIV EQUAL TO 0
D4
31
COMPUTE ABSOLUTE TOL, RELATIVE TOL BY MEANS OF FORM. 33
5
INITIALIZE STEP-COUNTER K=0 FOR DECOMPOSITION LOOP
B0
ADD 1 TO STEP-COUNTER K
IS K EQUAL TO KPIV
6
INTERCHANGE AUX(K) WITH AUX(KPIV)
INTERCHANGE COLUMNS K AND KPIV OF A BENEETH ROW K-1
8
IS K EQUAL TO I
9
COMPUTE ABSOLUTE VALUE OF PARAMETER SIG BY FORM. 10
IS SIG GREATER THAN TOL
D7
32
GENERATE CORRECT SIGN OF PARAMETER SIG
11
SAVE INFORMATION ON USELESS COLUMN
13
SAVE INTERCHANGE INFORMATION KPIV IN IPIV(K)
C0
COMPUTE UK(K) BY FORM. 12 AND STORE RESULT IN A(K,K)
C0
COMPUTE PARAMETER BETA BY MEANS OF FORM. 9
SAVE -SIG AS NEW MAIN DIAG. ELEMENT OF A IN AUX(N+K)
IS K EQUAL TO N
14
TRANSFORM REST OF MATRIX A BY MEANS OF FORM. 18, 21
TRANSFORM LAST N-K COMPONENTS OF VECTOR S BY FORM.29
SET PIV EQUAL TO MAXIMUM OF LAST N-K COMPONENTS OF
SET KPIV EQUAL TO INDEX OF MAXIMUM COMPONENT
19
TRANSFORM MATRIX B BY MEANS OF FORM. 26, 27
IS K EQUAL TO N
NO
YES
D0
SET ERROR PARAMETER IER=0
D0
COMPUTE N-TH ROW OF SOLUTION MATRIX X BY FORM. 30
IS N EQUAL TO 1
23
COMPUTE REST OF SOLUTION MATRIX X BY FORM. 31
26
COMPUTE VECTOR OF LEAST SQUARES BY FORMULA 32
D5
30
SET ERROR PARAMETER IER=-2
D6
31
SET ERROR PARAMETER IER=-1
D7
32
SET ERROR PARAMETER IER=K
EXIT

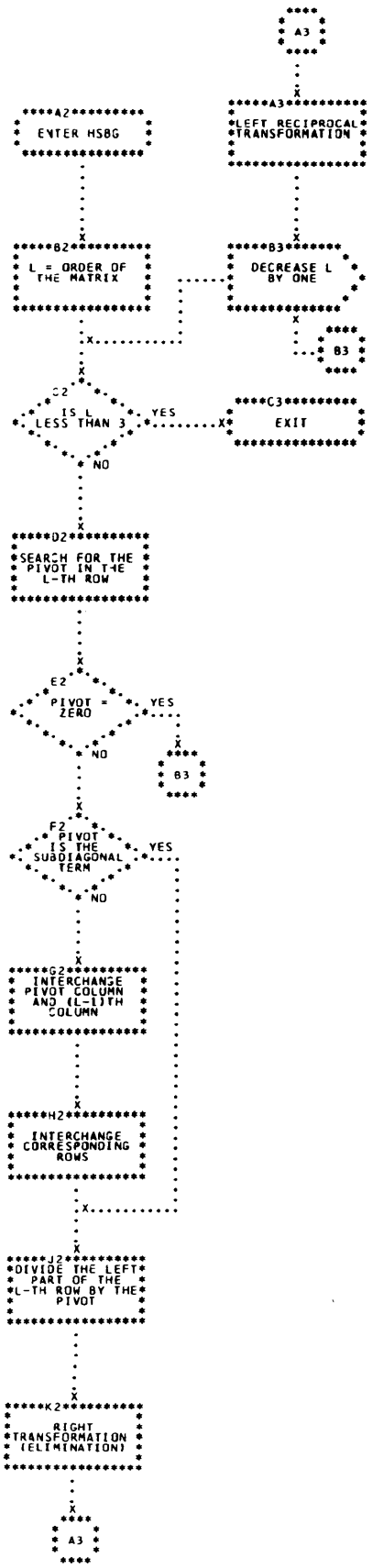

ENTER
MV = 1
YES
NO
10
GENERATE IDENTITY MATRIX
25
COMPUTE INITIAL NORM. EQUATION (1)
COMPUTE FINAL NORM. EQUATION (2)
SET INDICATOR TO ZERO
45
COMPUTE THRESHOLD = NORM / NO.ROWS
SET L = 1
SET M = L+1
A(LM) LESS THAN THRESHOLD
130
A5
C2
NO
GO
A6
A7
A8
SET INDICATOR TO 1
COMPUTE SIN AND COS. EQUATIONS (3)-(7)
SET I = 1
I = L
B3
I = M
80
SOLVE EQUATIONS (8),(9)
REPLACE A(IL) = B, A(IM) = C
115
MV = 1
125
C1
120
SOLVE EQUATIONS (10)-(12)
I = N
INCREASE I BY 1
SOLVE EQUATIONS (13)-(15)
M = N
135
INCREASE M BY 1
A9
140
M = N-1
145
INCREASE L BY 1
150
INDICATOR = 1
NO
155
CLEAR INDICATOR
160
THRESHOLD GREATER THAN FINAL NORM
165
SORT EIGENVALUES
A0
2

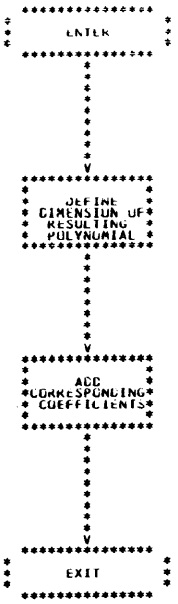

AO
MV = 1
YES
NO
175
SORT
EIGENVECTORS
185
EXIT

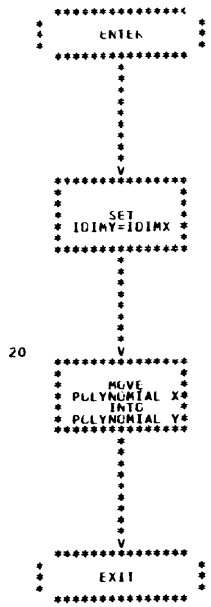

ENTER
EIGEN
EIGENVALUES,
EIGENVECTOR
FORM RECIP.
OF SQUAREROOT
OF
EIGENVALUES.
EQ. (1)
FORM
B**(-1/2)
FORM
SYMMETRIC
MATRIX S. EQ.
(2)-(3)
EIGEN
EIGENVALUES,
EIGENVECTOR
NORMALIZE
EIGENVECTORS.
EQ. (4)-(5)
EXIT

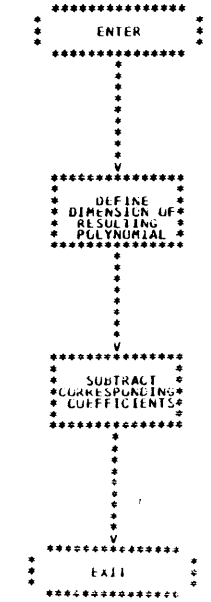

A3
A4
A2
ENTER ATEIG
A3
COMPUTE SHIFT
A4
A(N,N) IS TAKEN AS AN EIGENVALUE
B2
DEFINE N AS MATRIX ORDER
B3
STORE RESULTS OF PREVIOUS ITERATION
B4
IANA(N)=1
C2
C2
N IS ONE
YES
NO
A4
C3
SEARCH FOR A PARTITION OF MATRIX
C4
N IS ONE
YES
NO
D2
INITIALIZATION
D3
DOUBLE OR ITERATION
D4
DECREASE N BY ONE
C2
E2
COMPUTE ROOTS OF LOWER MAIN 2X2 SUBMATRIX
F2
N IS TWO
YES
NO
F3
IANA(N)=0 IANA(N-1)=2
F4
N IS TWO
YES
NO
F5
EXIT
G2
A(N-1,N-2) IS SMALL
YES
NO
G4
DECREASE N BY TWO
C2
H2
A(N,N-1) IS SMALL
YES
NO
A4
J2
STABILITY
YES
NO
J3
A(N,N-1) IS LESS THAN
NO
YES
A4
K2
MAX ITERATION NUMBER IS REACHED
YES
NO
A3

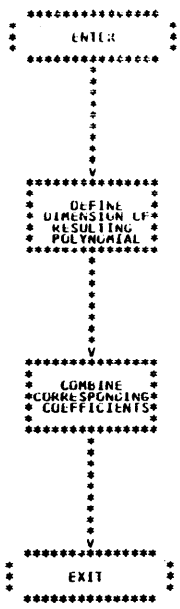
J3. A(N-1,N-2)

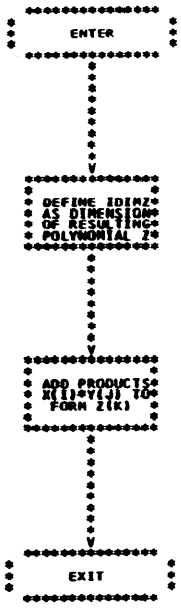

A3
A3
LEFT RECIPROCAL TRANSFORMATION
A2
ENTER HSBG
B2
L = ORDER OF THE MATRIX
B3
DECREASE L BY ONE
B3
C2
IS L LESS THAN 3
YES
C3
EXIT
NO
D2
SEARCH FOR THE PIVOT IN THE L-TH ROW
E2
PIVOT = ZERO
YES
B3
NO
F2
PIVOT IS THE SUBDIAGONAL TERM
YES
NO
G2
INTERCHANGE PIVOT COLUMN AND (L-1)TH COLUMN
H2
INTERCHANGE CORRESPONDING ROWS
J2
DIVIDE THE LEFT PART OF THE L-TH ROW BY THE PIVOT
K2
RIGHT TRANSFORMATION (ELIMINATION)
A3

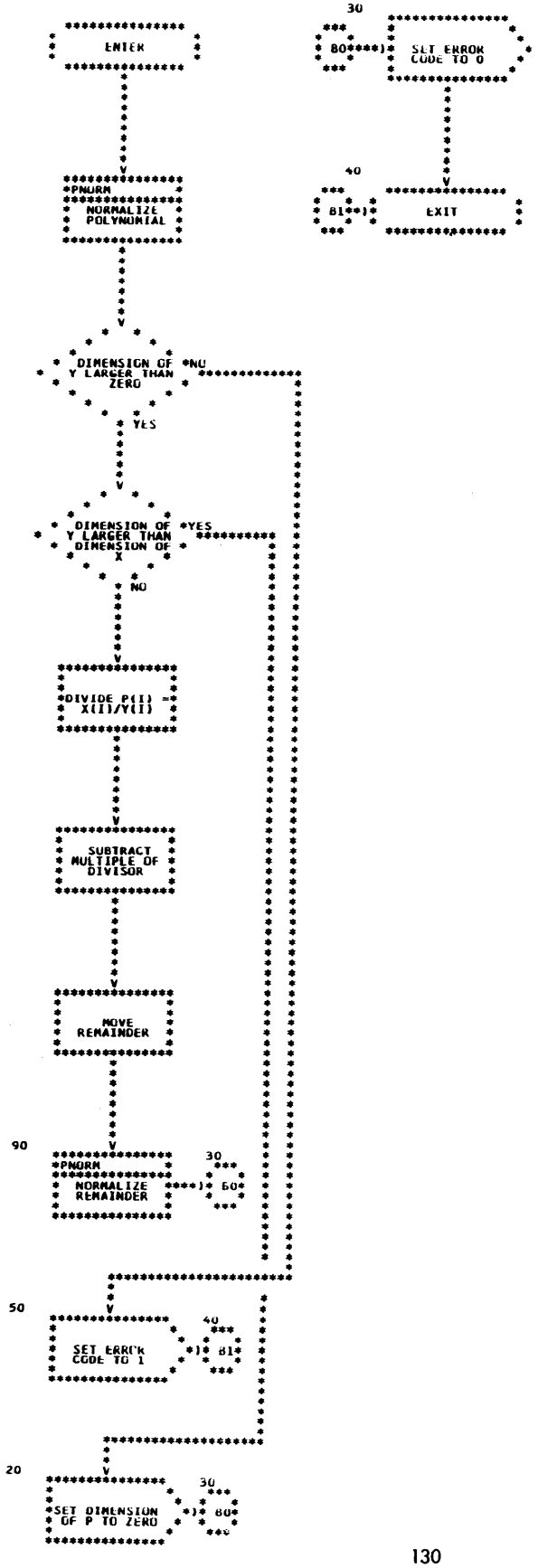

ENTER
DEFINE
DIMENSION OF
RESULTING
POLYNOMIAL
ADD
CORRESPONDING
COEFFICIENTS
EXIT

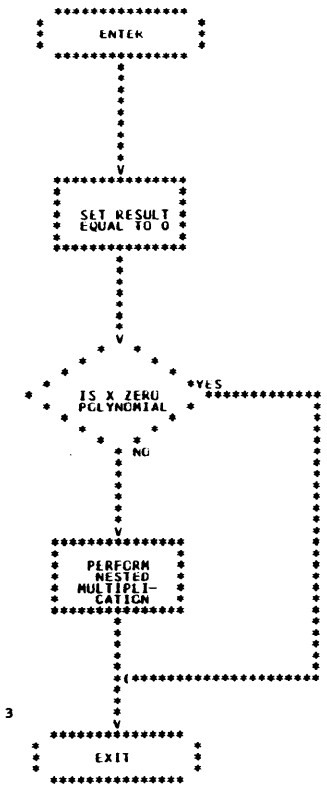

ENTER
SET IDIMY=IDIMX
20
MOVE POLYNOMIAL X INTO POLYNOMIAL Y
EXIT


ENTER
DEFINE DIMENSION OF RESULTING POLYNOMIAL
SUBTRACT CORRESPONDING COEFFICIENTS
EXIT

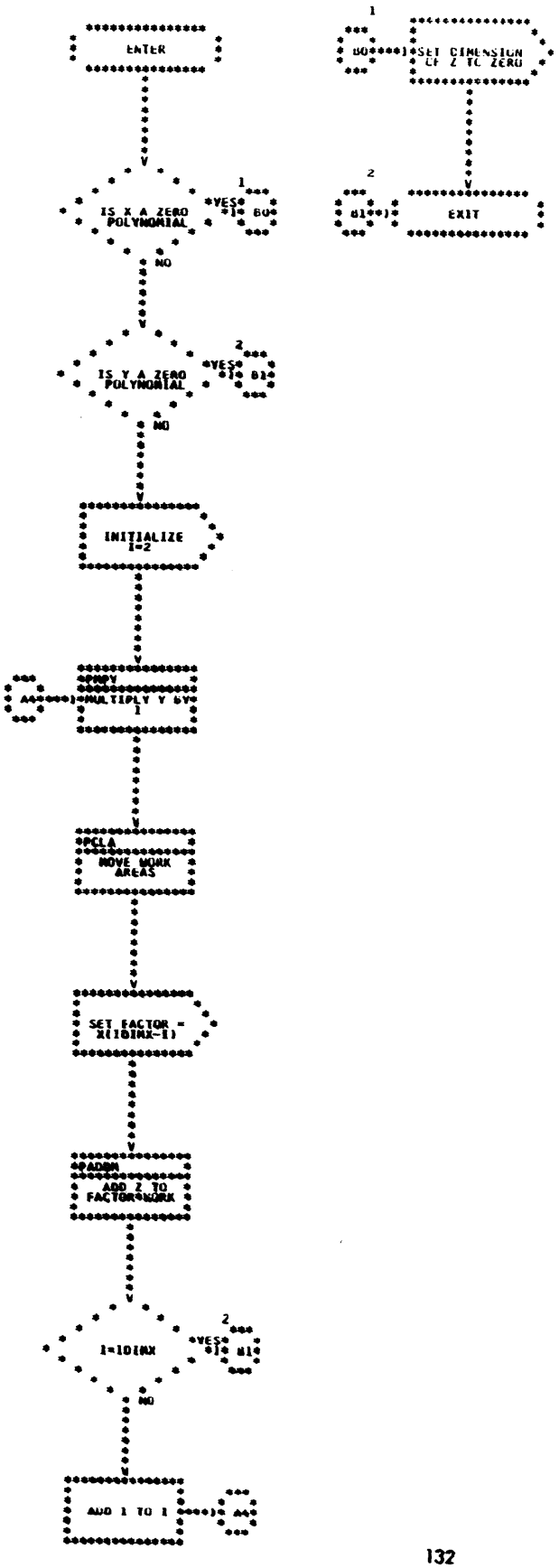

ENTER
DEFINE DIMENSION OF RESULTING POLYNOMIAL
COMBINE CORRESPONDING COEFFICIENTS
EXIT


ENTER
DEFINE IDIMZ AS DIMENSION OF RESULTING POLYNOMIAL Z
ADD PRODUCTS X(I)*Y(J) TO FORM Z(K)
EXIT

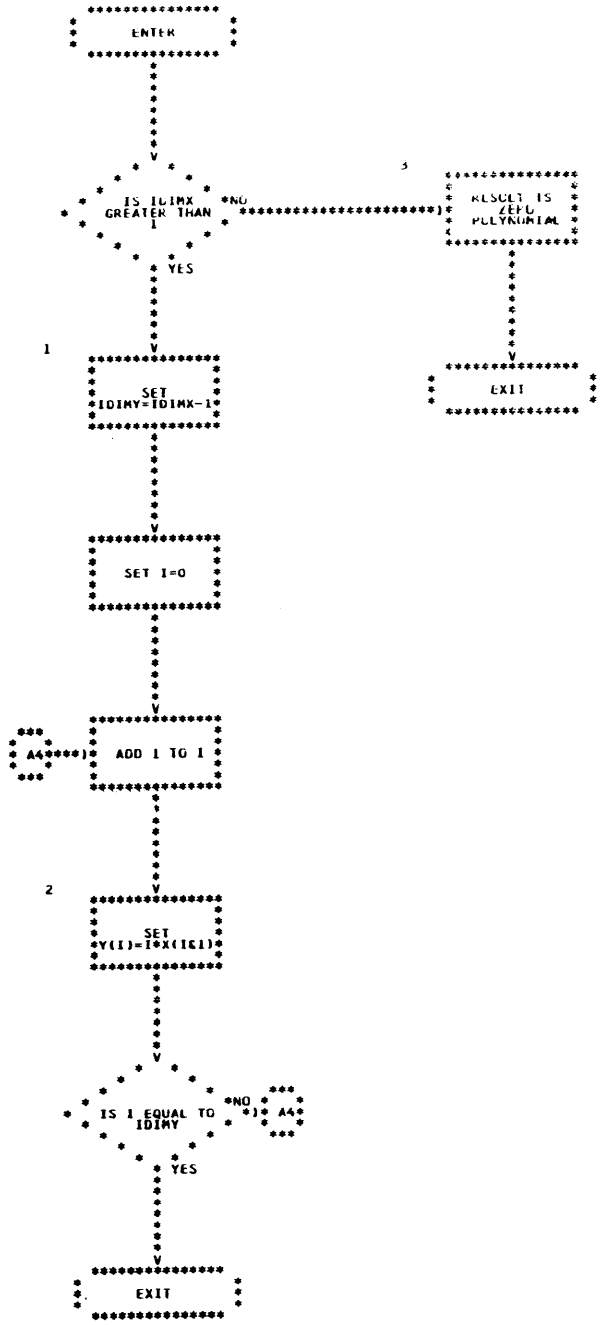

ENTER
*PNORM
NORMALIZE POLYNOMIAL
DIMENSION OF Y LARGER THAN ZERO
*NO
YES
DIMENSION OF Y LARGER THAN DIMENSION OF X
*YES
NO
*DIVIDE P(I) = X(I)/Y(I)
SUBTRACT MULTIPLE OF DIVISOR
MOVE REMAINDER
90
*PNORM
NORMALIZE REMAINDER
30
60
50
SET ERROR CODE TO 1
40
81
20
*SET DIMENSION OF P TO ZERO
30
80
30
80*
SET ERROR CODE TO 0
40
81*
EXIT

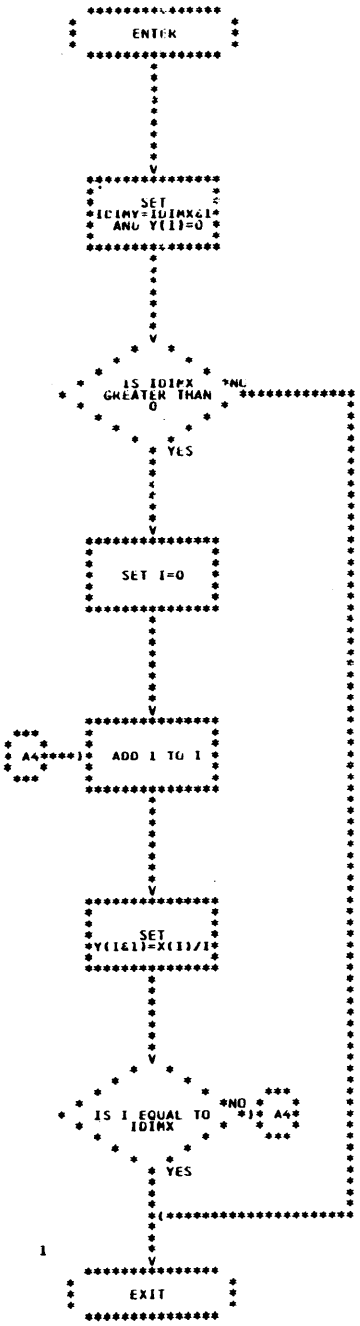

ENTER
SET RESULT EQUAL TO 0
IS X ZERO POLYNOMIAL
YES
NO
PERFORM NESTED MULTIPLI-CATION
EXIT

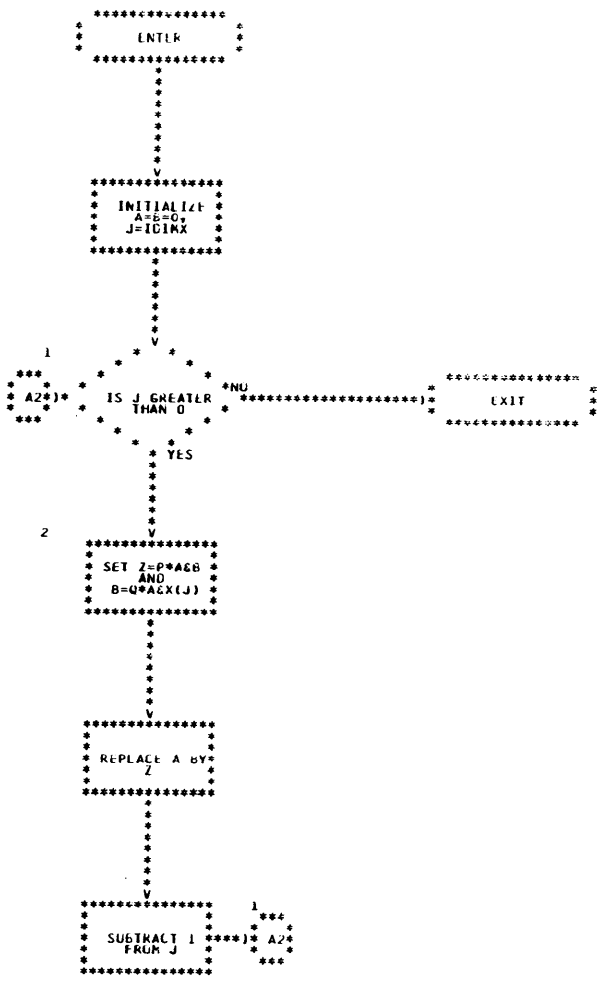
3

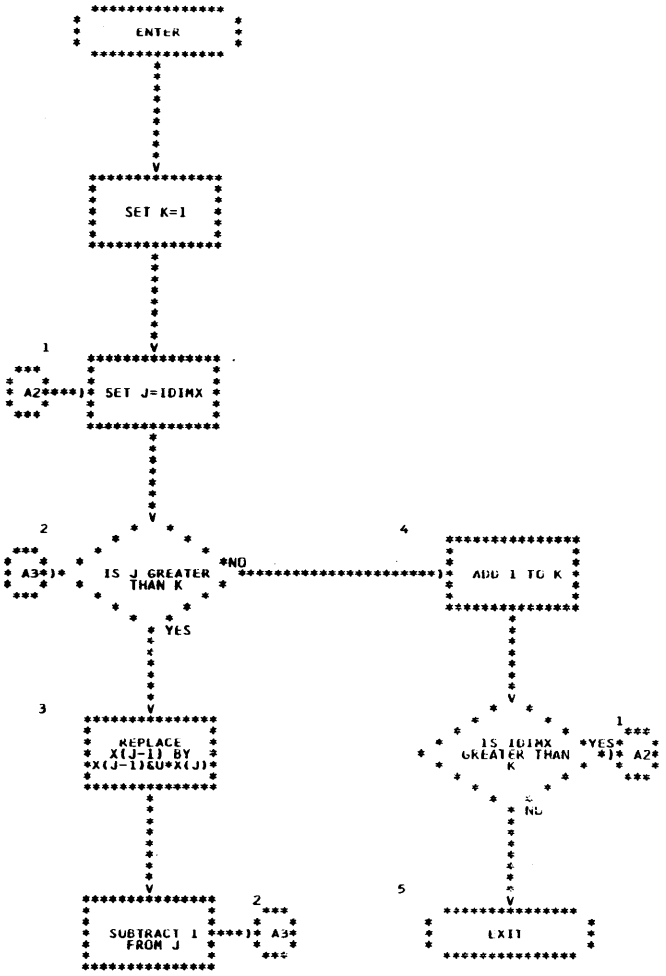

ENTER
IS X A ZERO POLYNOMIAL
YES
NO
B0
IS Y A ZERO POLYNOMIAL
YES
NO
B1
INITIALIZE I=2
AMPY
MULTIPLY Y BY I
A4
PCLA
MOVE WORK AREAS
SET FACTOR = X(IDIMX-I)
PADDM
ADD Z TO FACTOR*WORK
I=IDIMX
YES
NO
B1
ADD 1 TO I
A4
SET DIMENSION OF Z TO ZERO
B0
EXIT
B1

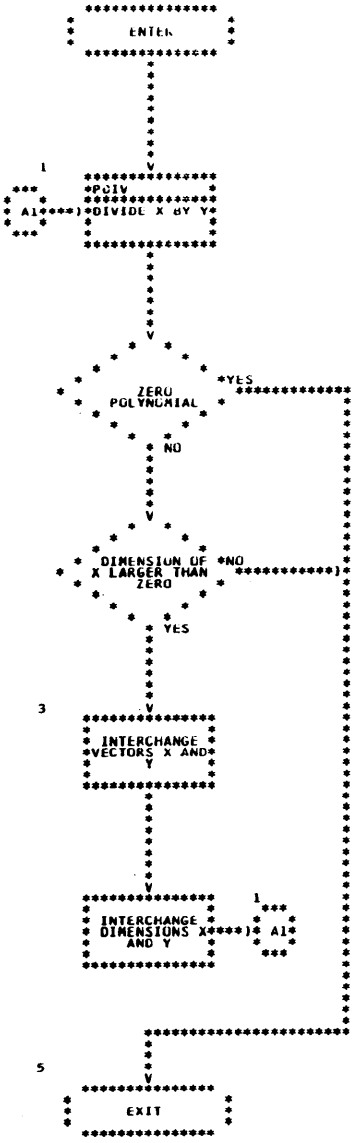

ENTER
INITIALIZE QUADRATIC SYNTHETIC DIVISION
COMPUTE VALUE OF DERIVATIVE BY SUBROUTINE PQSD
COMPUTE VALUE OF POLYNOMIAL
EXIT

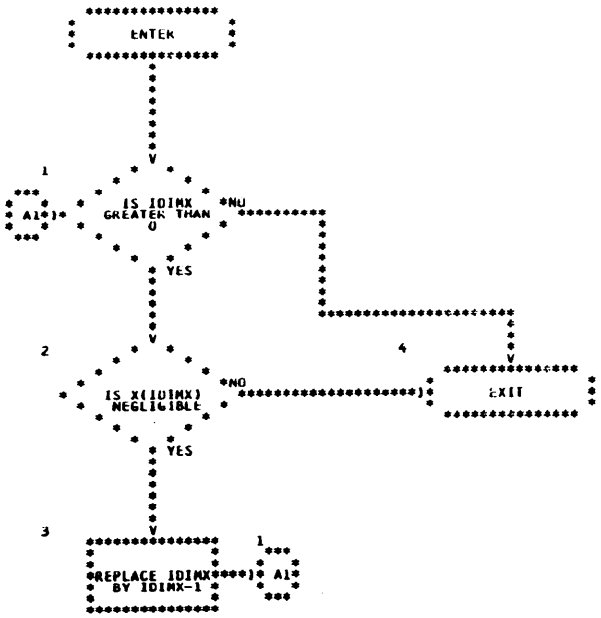

ENTER
IS IDIMX GREATER THAN 1
NO
YES
RESULT IS ZERO POLYNOMIAL
EXIT
SET IDIMY=IDIMX-1
SET I=0
A4
ADD 1 TO I
SET Y(I)=I*X(IC1)
IS I EQUAL TO IDIMY
NO
A4
YES
EXIT

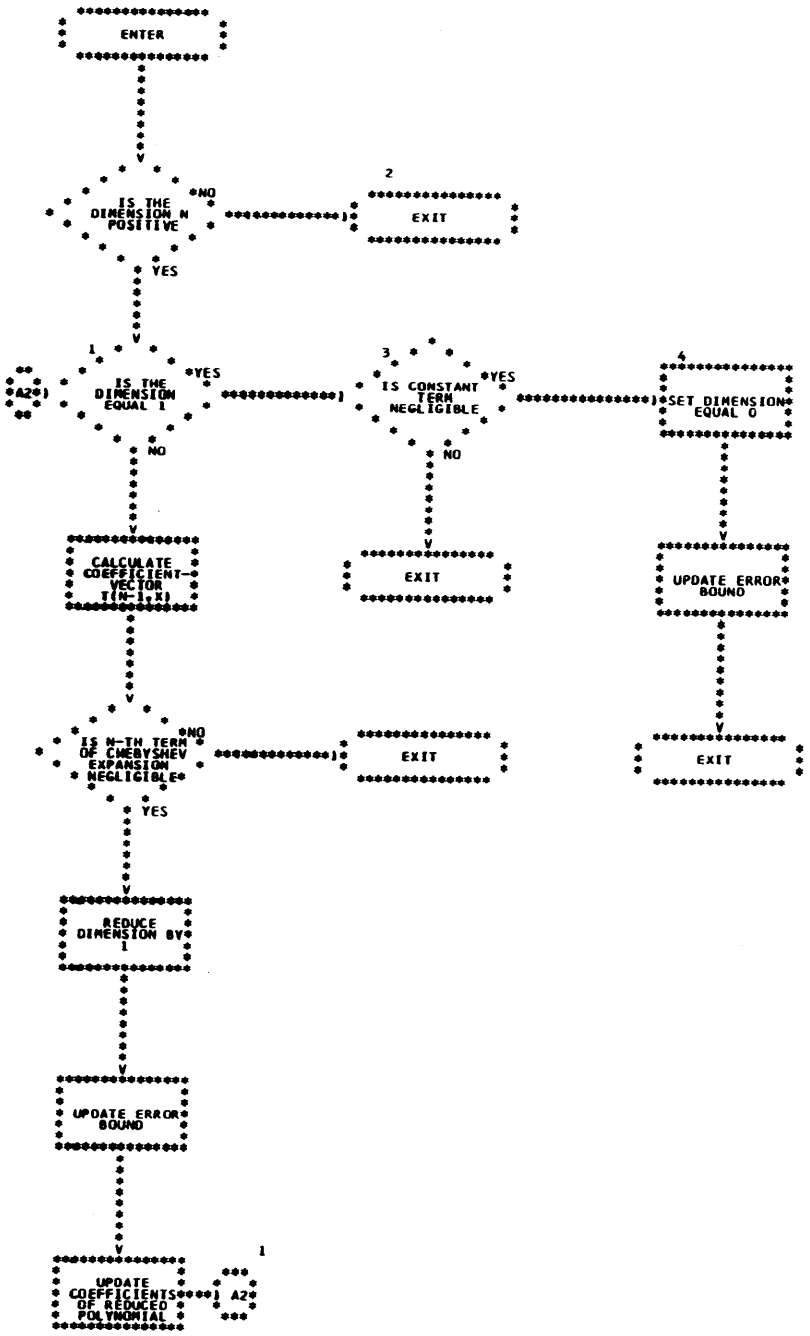

ENTER
SET IDIMY=IDIMX&I AND Y(I)=0
IS IDIMX GREATER THAN 0
*NO
YES
SET I=0
ADD 1 TO I
A4
SET Y(I&I)=X(I)/I
IS I EQUAL TO IDIMX
*NO
A4
YES
1
EXIT

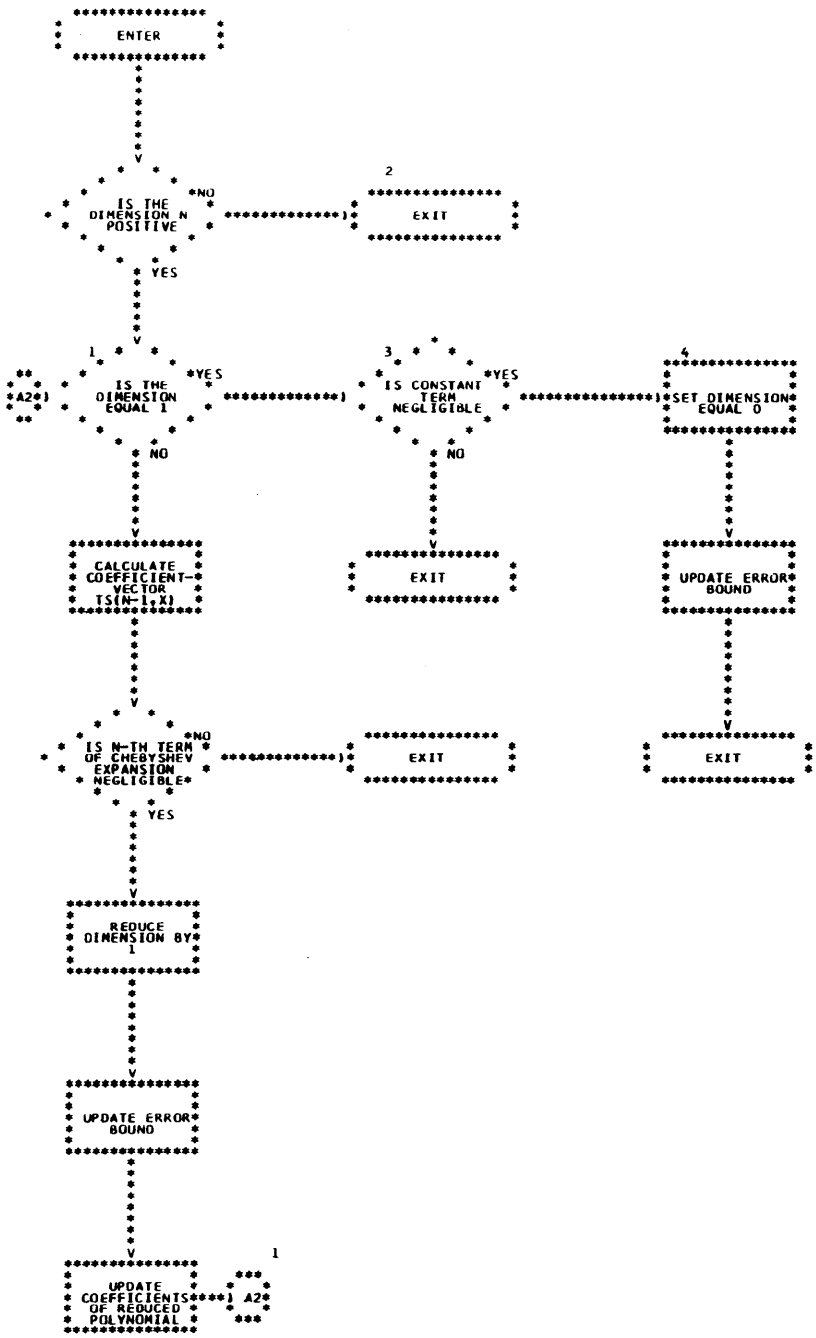

ENTER
INITIALIZE A=B=0, J=ICIMX
1
A2
IS J GREATER THAN 0
NO
EXIT
YES
2
SET Z=P*A&B AND B=Q*A&X(J)
REPLACE A BY Z
SUBTRACT 1 FROM J
1
A2

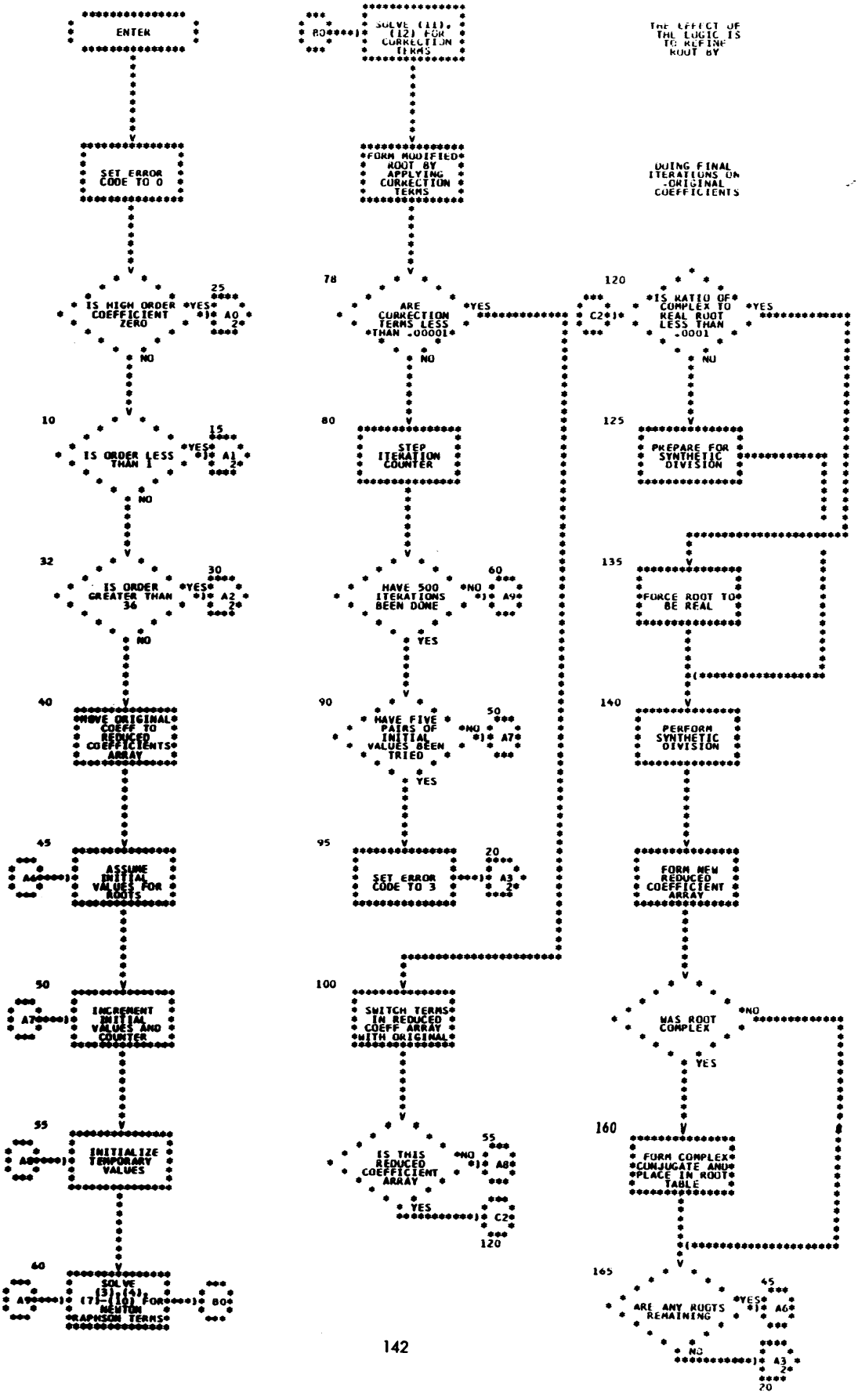

ENTER
SET K=1
1
A2
SET J=IDIMX
2
A3
IS J GREATER THAN K
NO
YES
4
ADD 1 TO K
3
REPLACE X(J-1) BY X(J-1)&U*X(J)
SUBTRACT 1 FROM J
2
A3
1
IS IDIMX GREATER THAN K
YES
A2
NO
5
EXIT

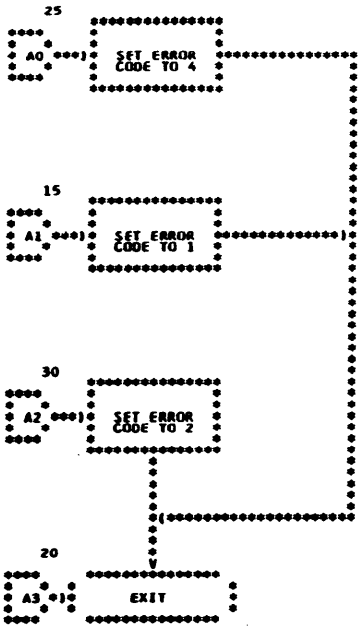

ENTER
1
PDIV
DIVIDE X BY Y
A1
ZERO POLYNOMIAL
YES
NO
DIMENSION OF X LARGER THAN ZERO
NO
YES
3
INTERCHANGE VECTORS X AND Y
INTERCHANGE DIMENSIONS X AND Y
1
A1
5
EXIT

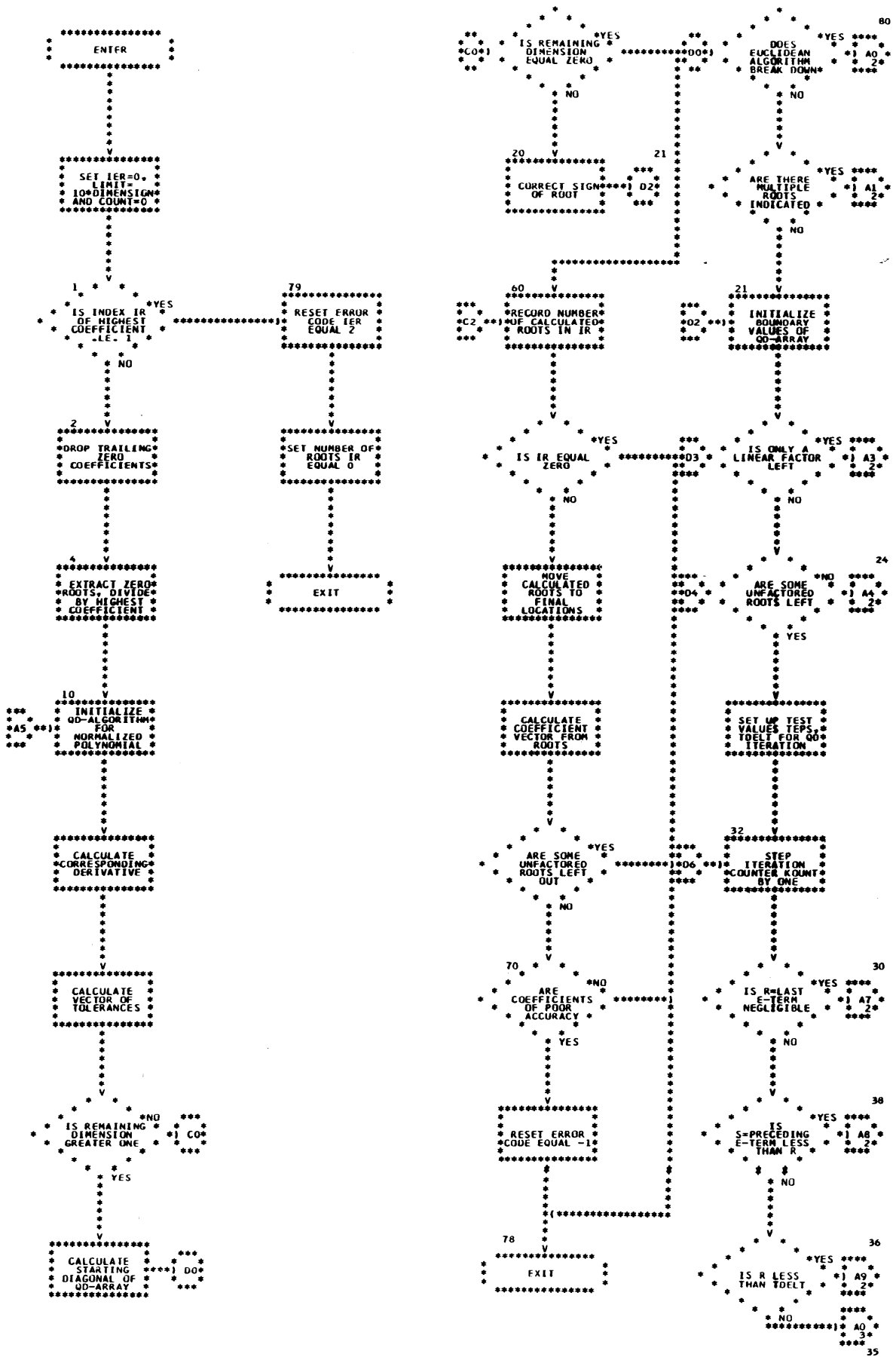

ENTER
1
A1
IS IDIMX GREATER THAN 0
*NO
YES
2
4
IS X(IDIMX) NEGLIGIBLE
*NO
EXIT
YES
3
1
REPLACE IDIMX BY IDIMX-1
A1

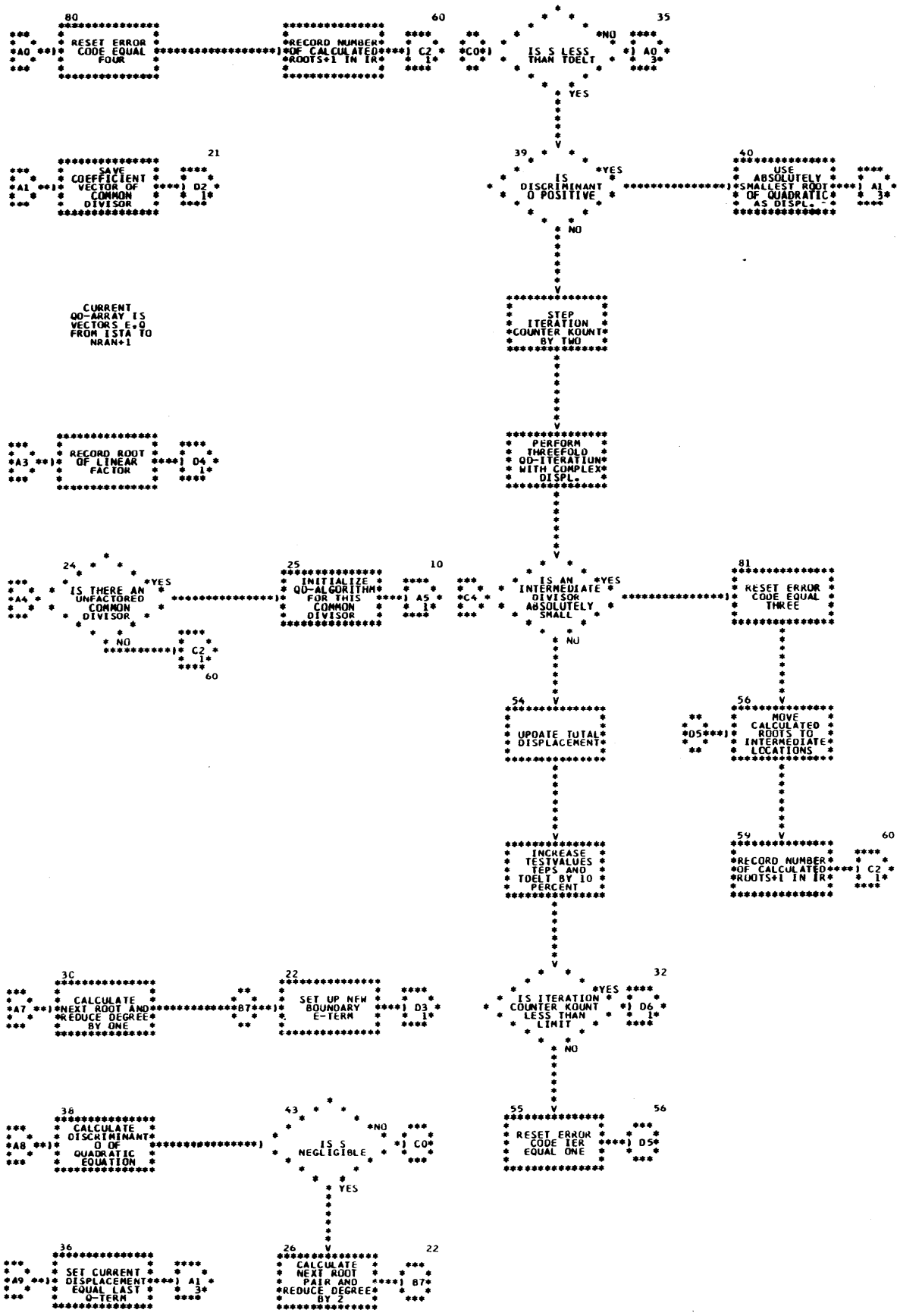

ENTER
IS THE DIMENSION N POSITIVE
NO
2
EXIT
YES
1
IS THE DIMENSION EQUAL 1
YES
A2
3
IS CONSTANT TERM NEGLIGIBLE
YES
4
SET DIMENSION EQUAL 0
NO
CALCULATE COEFFICIENT-A VECTOR T(N-1,X)
EXIT
UPDATE ERROR BOUND
IS N-TH TERM OF CHEBYSHEV EXPANSION NEGLIGIBLE
NO
EXIT
EXIT
YES
REDUCE DIMENSION BY 1
UPDATE ERROR BOUND
UPDATE COEFFICIENTS OF REDUCED POLYNOMIAL
1
A2

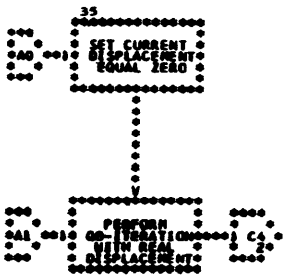

ENTER
IS THE DIMENSION N POSITIVE
*NO
EXIT
2
YES
1
IS THE DIMENSION EQUAL 1
*YES
A2*
3
IS CONSTANT TERM NEGLIGIBLE
*YES
4
SET DIMENSION EQUAL 0
NO
NO
CALCULATE COEFFICIENT-VECTOR TS(N-1,X)
EXIT
UPDATE ERROR BOUND
IS N-TH TERM OF CHEBYSHEV EXPANSION NEGLIGIBLE*
*NO
EXIT
EXIT
YES
REDUCE DIMENSION BY 1
UPDATE ERROR BOUND
UPDATE COEFFICIENTS OF REDUCED POLYNOMIAL
1
A2*

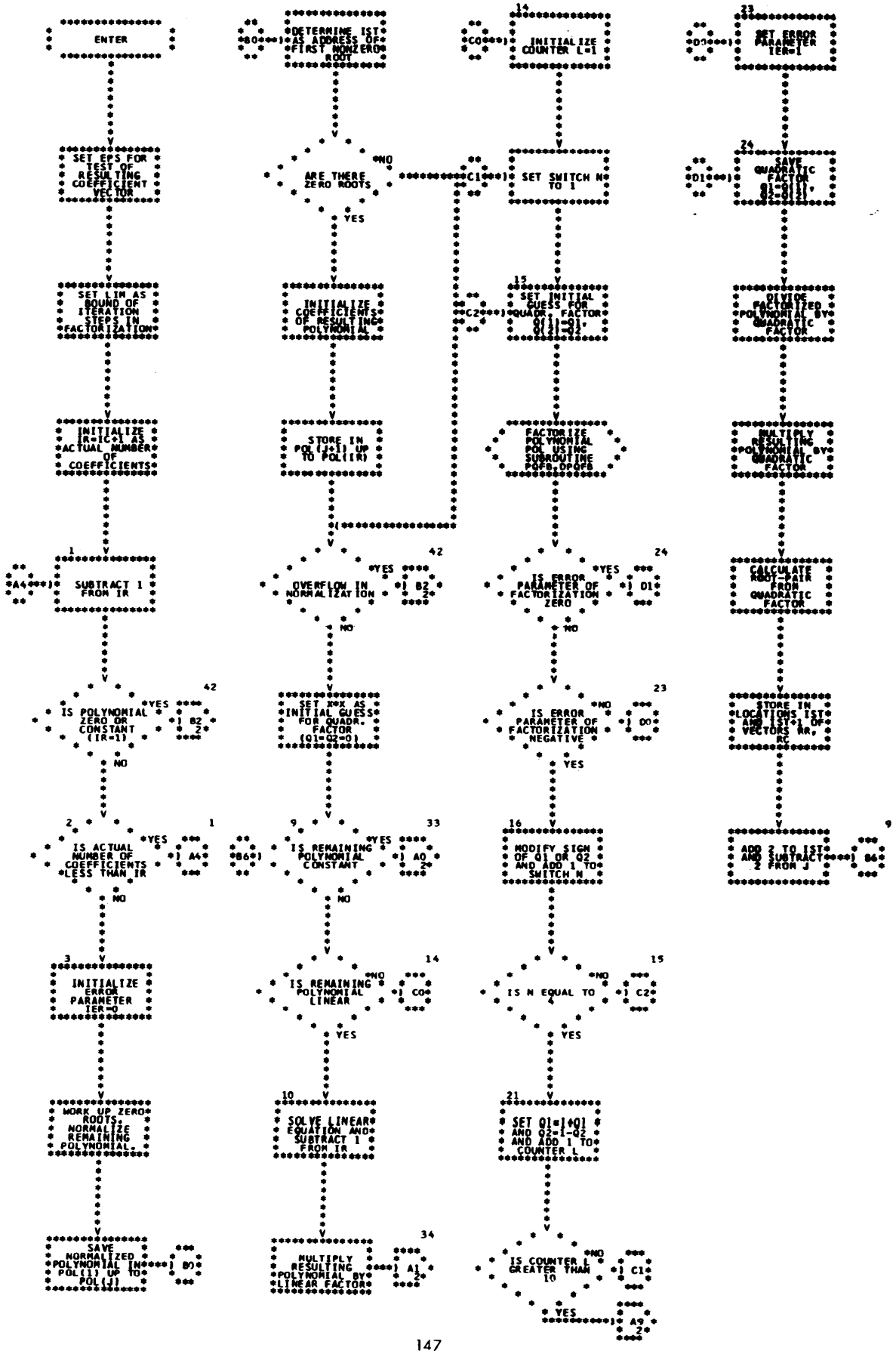

ENTER
SET ERROR CODE TO 0
IS HIGH ORDER COEFFICIENT ZERO
YES
25
A0
2
NO
10
IS ORDER LESS THAN 1
YES
15
A1
2
NO
32
IS ORDER GREATER THAN 36
YES
30
A2
2
NO
40
MOVE ORIGINAL COEFF TO REDUCED COEFFICIENTS ARRAY
45
A6
ASSUME INITIAL VALUES FOR ROOTS
50
A7
INCREMENT INITIAL VALUES AND COUNTER
55
A8
INITIALIZE TEMPORARY VALUES
60
A9
SOLVE (3)-(4), (7)-(10) FOR NEWTON RAPHSON TERMS
80
B0
SOLVE (11), (12) FOR CORRECTION TERMS
FORM MODIFIED ROOT BY APPLYING CORRECTION TERMS
78
ARE CORRECTION TERMS LESS THAN .00001
YES
NO
80
STEP ITERATION COUNTER
90
HAVE 500 ITERATIONS BEEN DONE
NO
60
A9
YES
95
HAVE FIVE PAIRS OF INITIAL VALUES BEEN TRIED
NO
50
A7
YES
95
SET ERROR CODE TO 3
20
A3
2
100
SWITCH TERMS IN REDUCED COEFF ARRAY WITH ORIGINAL
55
IS THIS REDUCED COEFFICIENT ARRAY
NO
A8
YES
C2
120
THE EFFECT OF THE LOGIC IS TO REFINE ROOT BY
DOING FINAL ITERATIONS ON ORIGINAL COEFFICIENTS
120
C2
IS RATIO OF COMPLEX TO REAL ROOT LESS THAN .0001
YES
NO
125
PREPARE FOR SYNTHETIC DIVISION
135
FORCE ROOT TO BE REAL
140
PERFORM SYNTHETIC DIVISION
FORM NEW REDUCED COEFFICIENT ARRAY
WAS ROOT COMPLEX
NO
YES
160
FORM COMPLEX CONJUGATE AND PLACE IN ROOT TABLE
165
ARE ANY ROOTS REMAINING
YES
45
A6
NO
A3
2
20

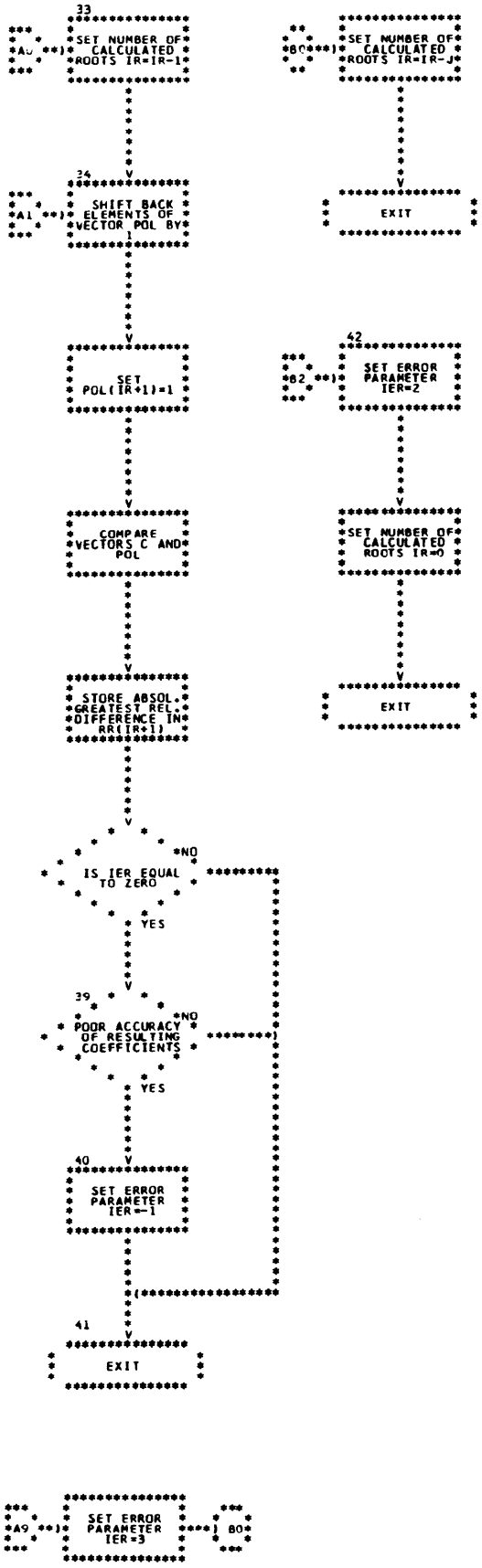

25
A0
SET ERROR CODE TO 4
15
A1
SET ERROR CODE TO 1
30
A2
SET ERROR CODE TO 2
20
A3
EXIT

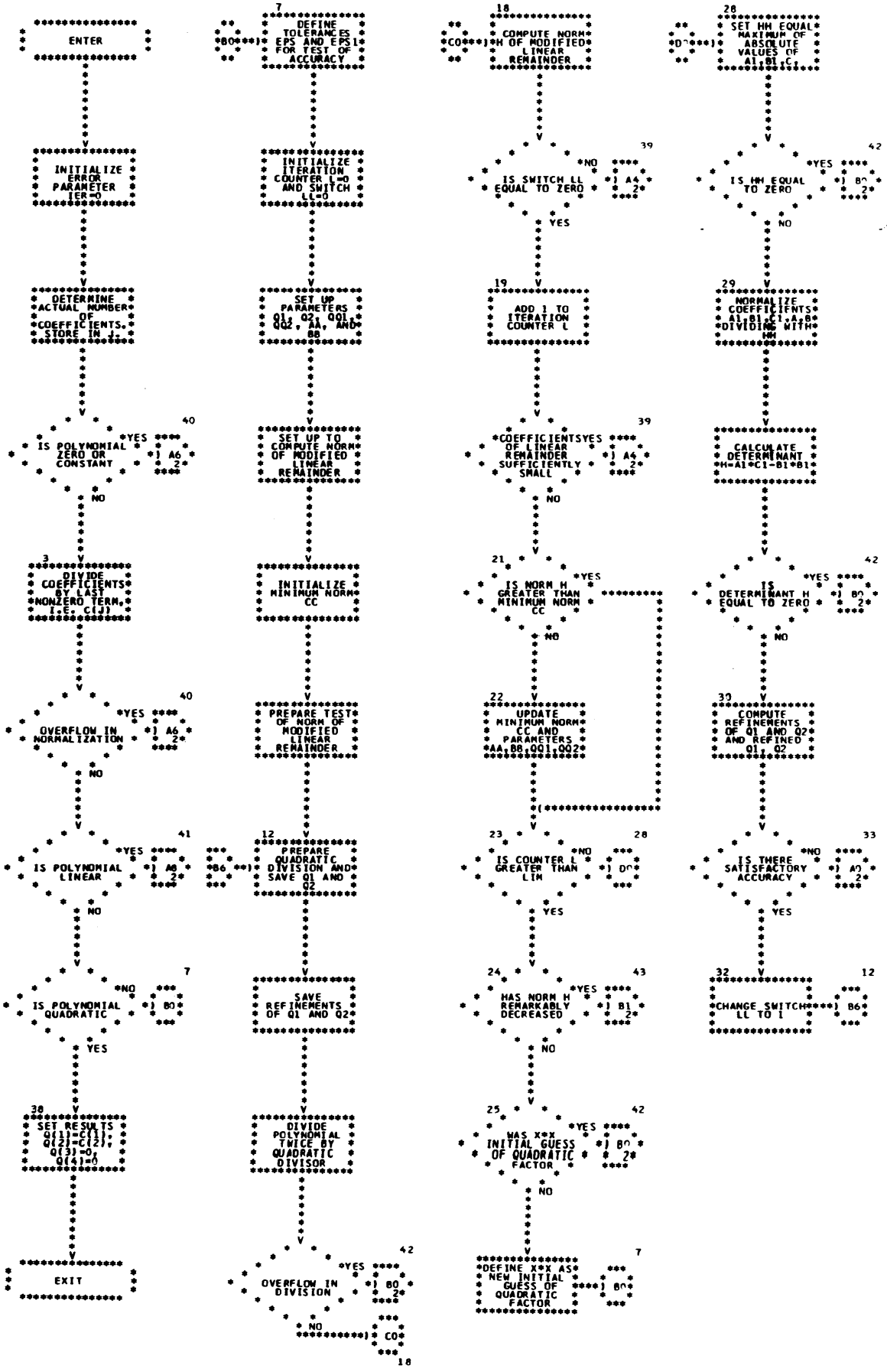

ENTER
SET IER=0, LIMIT=10*DIMENSION AND COUNT=0
1
IS INDEX IR OF HIGHEST COEFFICIENT .LE. 1
YES
NO
79
RESET ERROR CODE IER EQUAL 2
2
DROP TRAILING ZERO COEFFICIENTS
SET NUMBER OF ROOTS IR EQUAL 0
EXIT
4
EXTRACT ZERO ROOTS, DIVIDE BY HIGHEST COEFFICIENT
10
INITIALIZE QD-ALGORITHM FOR NORMALIZED POLYNOMIAL
A5
CALCULATE CORRESPONDING DERIVATIVE
CALCULATE VECTOR OF TOLERANCES
IS REMAINING DIMENSION GREATER ONE
NO
YES
C0
CALCULATE STARTING DIAGONAL OF QD-ARRAY
D0
IS REMAINING DIMENSION EQUAL ZERO
C0
D0
20
CORRECT SIGN OF ROOT
21
D2
60
RECORD NUMBER OF CALCULATED ROOTS IN IR
C2
D2
IS IR EQUAL ZERO
D3
MOVE CALCULATED ROOTS TO FINAL LOCATIONS
D4
CALCULATE COEFFICIENT VECTOR FROM ROTS
ARE SOME UNFACTORED ROOTS LEFT OUT
D6
70
ARE COEFFICIENTS OF POOR ACCURACY
RESET ERROR CODE EQUAL -1
78
EXIT
80
DOES EUCLIDEAN ALGORITHM BREAK DOWN
A0
ARE THERE MULTIPLE ROOTS INDICATED
A1
21
INITIALIZE BOUNDARY VALUES OF QD-ARRAY
D2
IS ONLY A LINEAR FACTOR LEFT
A3
24
ARE SOME UNFACTORED ROOTS LEFT
A4
SET UP TEST VALUES TEPS, TDELT FOR QD ITERATION
32
STEP ITERATION COUNTER KOUNT BY ONE
30
IS R=LAST E-TERM NEGLIGIBLE
A7
38
IS S=PRECEDING E-TERM LESS THAN R
A8
36
IS R LESS THAN TDELT
A9
A0
3
35

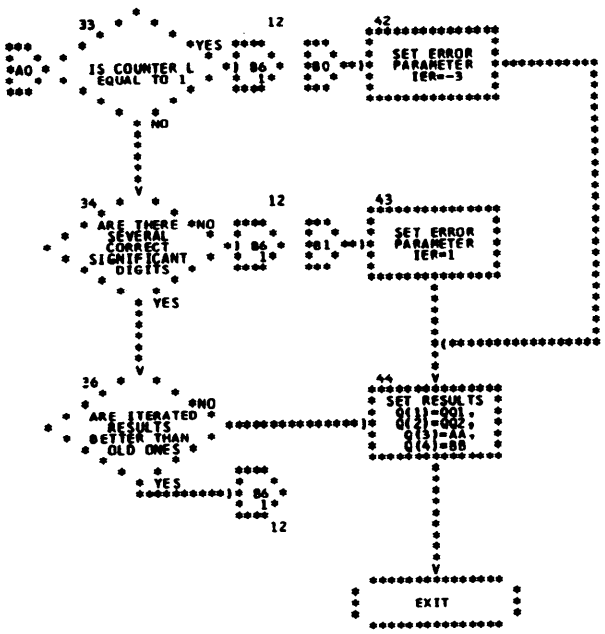

80
RESET ERROR CODE EQUAL FOUR
*A0
60
RECORD NUMBER *OF CALCULATED* *ROOTS+1 IN IR*
C2 1
C0
35
IS S LESS THAN TOELT
*NO
A0 3
YES
21
SAVE COEFFICIENT VECTOR OF COMMON DIVISOR
*A1
D2 1
39
IS DISCRIMINANT O POSITIVE
*YES
40
USE ABSOLUTELY SMALLEST ROOT OF QUADRATIC AS DISPL.
A1 3
NO
CURRENT QD-ARRAY IS VECTORS E,O FROM ISTA TO NRAN+1
STEP ITERATION COUNTER KOUNT BY TWO
RECORD ROOT OF LINEAR FACTOR
*A3
D4 1
PERFORM THREEFOLD QD-ITERATION WITH COMPLEX DISPL.
24
IS THERE AN UNFACTORED COMMON DIVISOR
*YES
*A4
25
INITIALIZE QD-ALGORITHM FOR THIS COMMON DIVISOR
10
A5 1
*C4
IS AN INTERMEDIATE DIVISOR ABSOLUTELY SMALL
81
RESET ERROR CODE EQUAL THREE
NO
C2 1
60
54
UPDATE TOTAL DISPLACEMENT
56
*D5
MOVE CALCULATED ROOTS TO INTERMEDIATE LOCATIONS
INCREASE TESTVALUES EPS AND TOELI BY 10 PERCENT
59
RECORD NUMBER *OF CALCULATED* *ROOTS+1 IN IR*
C2 1
60
3C
CALCULATE NEXT ROOT AND REDUCE DEGREE BY ONE
*A7
B7
22
SET UP NEW BOUNDARY E-TERM
D3 1
32
IS ITERATION COUNTER KOUNT LESS THAN LIMIT
*YES
D6 1
38
CALCULATE DISCRIMINANT O OF QUADRATIC EQUATION
*A8
43
IS S NEGLIGIBLE
*NO
C0
YES
55
RESET ERROR CODE IER EQUAL ONE
56
D5
36
SET CURRENT DISPLACEMENT EQUAL LAST Q-TERM
*A9
A1 3
26
CALCULATE NEXT ROOT PAIR AND REDUCE DEGREE BY 2
22
B7

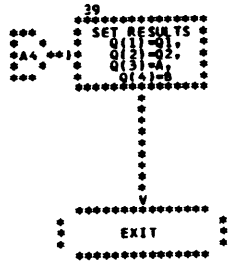

35
A0
SET CURRENT
DISPLACEMENT
EQUAL ZERO
A1
PERFORM
ONE ITERATION
WITH REAL
DISPLACEMENT
C4
2

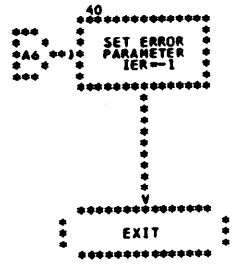

ENTER
SET EPS FOR TEST OF RESULTING COEFFICIENT VECTOR
SET LIM AS BOUND OF ITERATION STEPS IN FACTORIZATION
INITIALIZE IR=IC+1 AS ACTUAL NUMBER OF COEFFICIENTS
1
SUBTRACT 1 FROM IR
A4
IS POLYNOMIAL ZERO OR CONSTANT (IR=1)
YES
42
B2 2
NO
2
IS ACTUAL NUMBER OF COEFFICIENTS LESS THAN IR
YES
1
A4
NO
3
INITIALIZE ERROR PARAMETER IER=0
WORK UP ZERO ROOTS, NORMALIZE REMAINING POLYNOMIAL.
SAVE NORMALIZED POLYNOMIAL IN POL(1) UP TO POL(J)
B0
B0
DETERMINE IST AS ADDRESS OF FIRST NONZERO ROOT
ARE THERE ZERO ROOTS
NO
YES
C1
INITIALIZE COEFFICIENTS OF RESULTING POLYNOMIAL
C2
STORE IN POL(J+1) UP TO POL(IR)
OVERFLOW IN NORMALIZATION
YES
42
B2 2
NO
SET X=X AS INITIAL GUESS FOR QUADR. FACTOR (Q1=Q2=0)
9
IS REMAINING POLYNOMIAL CONSTANT
YES
33
A0 2
NO
B6
IS REMAINING POLYNOMIAL LINEAR
NO
14
C0
YES
10
SOLVE LINEAR EQUATION AND SUBTRACT 1 FROM IR
MULTIPLY RESULTING POLYNOMIAL BY LINEAR FACTOR
34
A1 2
14
INITIALIZE COUNTER L=1
C0
SET SWITCH N TO 1
C1
15
SET INITIAL GUESS FOR QUADR. FACTOR Q(1)=Q1, Q(2)=Q2
C2
FACTORIZE POLYNOMIAL POL USING SUBROUTINE PQFB,DPQFB
IS ERROR PARAMETER OF FACTORIZATION ZERO
YES
24
D1 2
NO
IS ERROR PARAMETER OF FACTORIZATION NEGATIVE
NO
23
D0 2
YES
16
MODIFY SIGN OF Q1 OR Q2 AND ADD 1 TO SWITCH N
IS N EQUAL TO 4
NO
15
C2
YES
21
SET Q1=|Q1| AND Q2=-Q2 AND ADD 1 TO COUNTER L
IS COUNTER L GREATER THAN 10
NO
C1
YES
A9 2
23
SET ERROR PARAMETER IER=1
D0
24
SAVE QUADRATIC FACTOR Q1=Q(1), Q2=Q(2)
D1
DIVIDE FACTORIZED POLYNOMIAL BY QUADRATIC FACTOR
MULTIPLY RESULTING POLYNOMIAL BY QUADRATIC FACTOR
CALCULATE ROOT-PAIR FROM QUADRATIC FACTOR
STORE IN LOCATIONS IST AND IST+1 OF VECTORS RR, RC
ADD 2 TO IST AND SUBTRACT 2 FROM J
9
B6

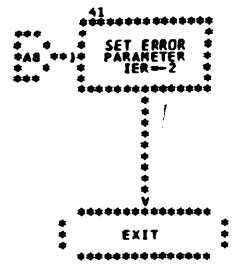

33
SET NUMBER OF CALCULATED ROOTS IR=IR-1
A0
34
SHIFT BACK ELEMENTS OF VECTOR POL BY 1
A1
SET POL(IR+1)=1
COMPARE VECTORS C AND POL
STORE ABSOL. GREATEST REL. DIFFERENCE IN RR(IR+1)
NO
IS IER EQUAL TO ZERO
YES
39
NO
POOR ACCURACY OF RESULTING COEFFICIENTS
YES
40
SET ERROR PARAMETER IER=-1
41
EXIT
B0
SET NUMBER OF CALCULATED ROOTS IR=IR-J
EXIT
42
B2
SET ERROR PARAMETER IER=2
SET NUMBER OF CALCULATED ROOTS IR=0
EXIT
A9
SET ERROR PARAMETER IER=3
B0

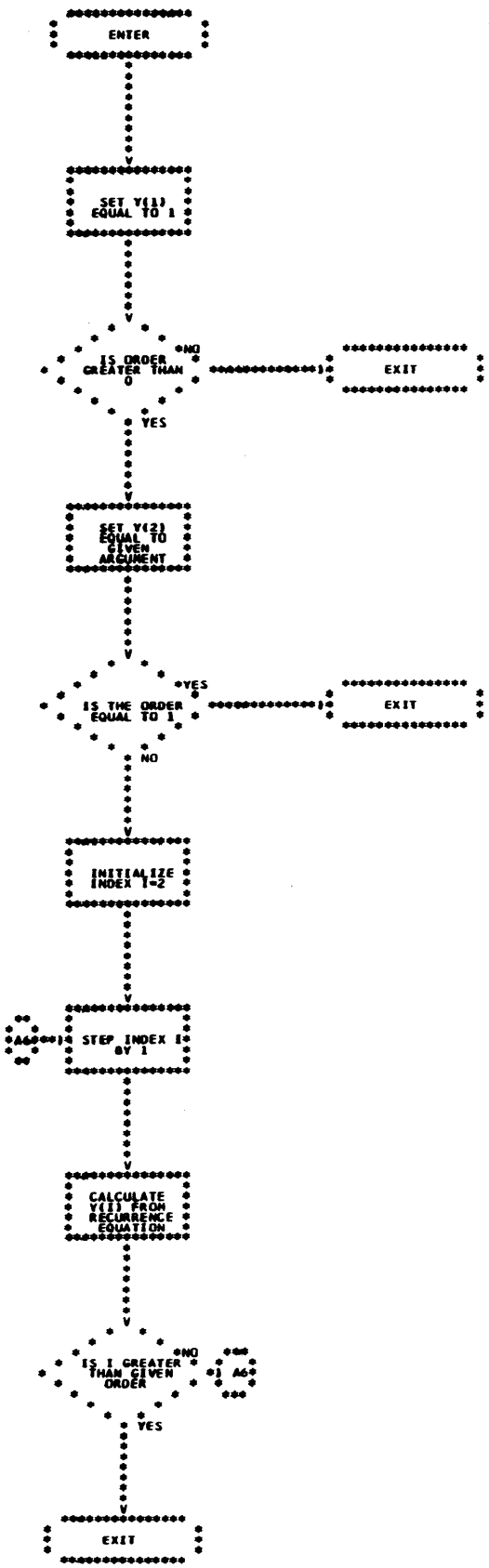

ENTER
INITIALIZE ERROR PARAMETER IER=0
DETERMINE ACTUAL NUMBER OF COEFFICIENTS. STORE IN J.
IS POLYNOMIAL ZERO OR CONSTANT
YES
NO
A6
DIVIDE COEFFICIENTS BY LAST NONZERO TERM, I.E. C(J)
OVERFLOW IN NORMALIZATION
IS POLYNOMIAL LINEAR
IS POLYNOMIAL QUADRATIC
SET RESULTS Q(1)=C(1), Q(2)=C(2), Q(3)=0, Q(4)=0
EXIT
B0
DEFINE TOLERANCES EPS AND EPS1 FOR TEST OF ACCURACY
INITIALIZE ITERATION COUNTER L=0 AND SWITCH LL=0
SET UP PARAMETERS Q11, Q12, QQ1, QQ2, AA, AND BB
SET UP TO COMPUTE NORM OF MODIFIED LINEAR REMAINDER
INITIALIZE MINIMUM NORM= CC
PREPARE TEST OF NORM OF MODIFIED LINEAR REMAINDER
B6
PREPARE QUADRATIC DIVISION AND SAVE Q1 AND Q2
SAVE REFINEMENTS OF Q1 AND Q2
DIVIDE POLYNOMIAL TWICE BY QUADRATIC DIVISOR
OVERFLOW IN DIVISION
C0
COMPUTE NORM H OF MODIFIED LINEAR REMAINDER
IS SWITCH LL EQUAL TO ZERO
A4
ADD 1 TO ITERATION COUNTER L
COEFFICIENTS OF LINEAR REMAINDER SUFFICIENTLY SMALL
IS NORM H GREATER THAN MINIMUM NORM CC
UPDATE MINIMUM NORM CC AND PARAMETERS AA,BB,QQ1,QQ2
IS COUNTER L GREATER THAN LIM
D0
HAS NORM H REMARKABLY DECREASED
B1
WAS X**X INITIAL GUESS OF QUADRATIC FACTOR
DEFINE X**X AS NEW INITIAL GUESS OF QUADRATIC FACTOR
D0
SET HH EQUAL MAXIMUM OF ABSOLUTE VALUES OF A1,B1,C1
IS HH EQUAL TO ZERO
NORMALIZE COEFFICIENTS A1,B1,C1,A,B DIVIDING THEM WITH HH
CALCULATE DETERMINANT H=A1*C1-B1*B1
IS DETERMINANT H EQUAL TO ZERO
COMPUTE REFINEMENTS OF Q1 AND Q2 AND REFINED Q11, Q12
IS THERE SATISFACTORY ACCURACY
CHANGE SWITCH LL TO 1

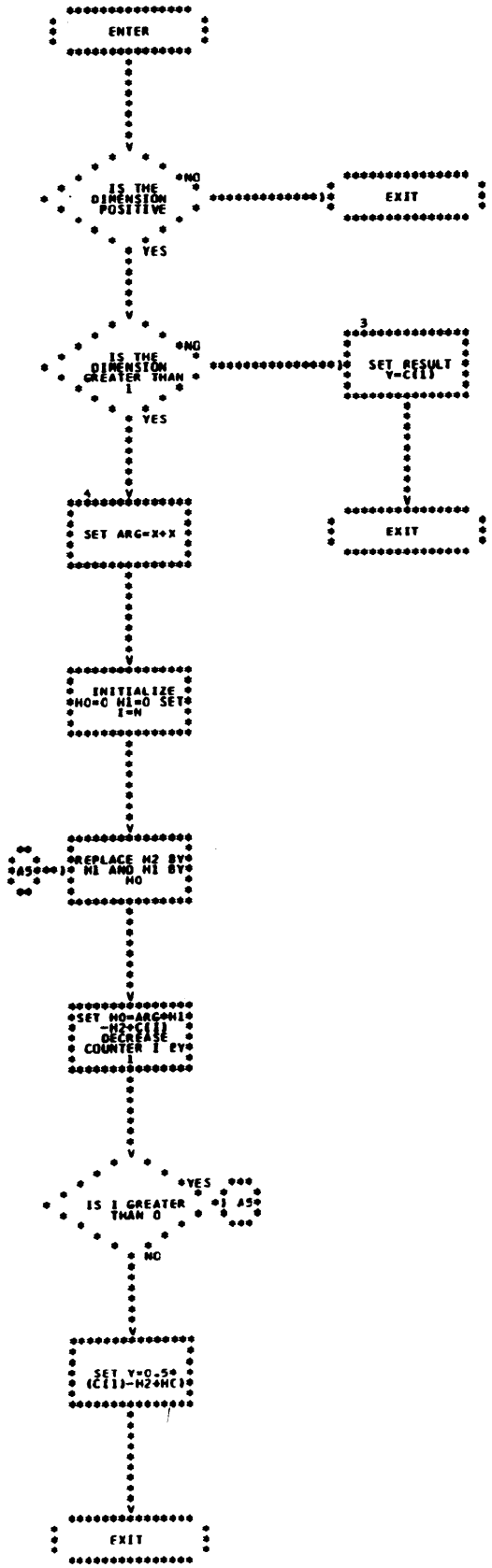

33
IS COUNTER L EQUAL TO 1
*YES
12
*A0
*B0
42
SET ERROR PARAMETER IER=-3
NO
34
ARE THERE SEVERAL CORRECT SIGNIFICANT DIGITS
*NO
12
*B1
43
SET ERROR PARAMETER IER=1
YES
36
ARE ITERATED RESULTS BETTER THAN OLD ONES
*NO
YES
12
44
SET RESULTS Q(1)=01, Q(2)=02, Q(3)=AA, Q(4)=BB
EXIT

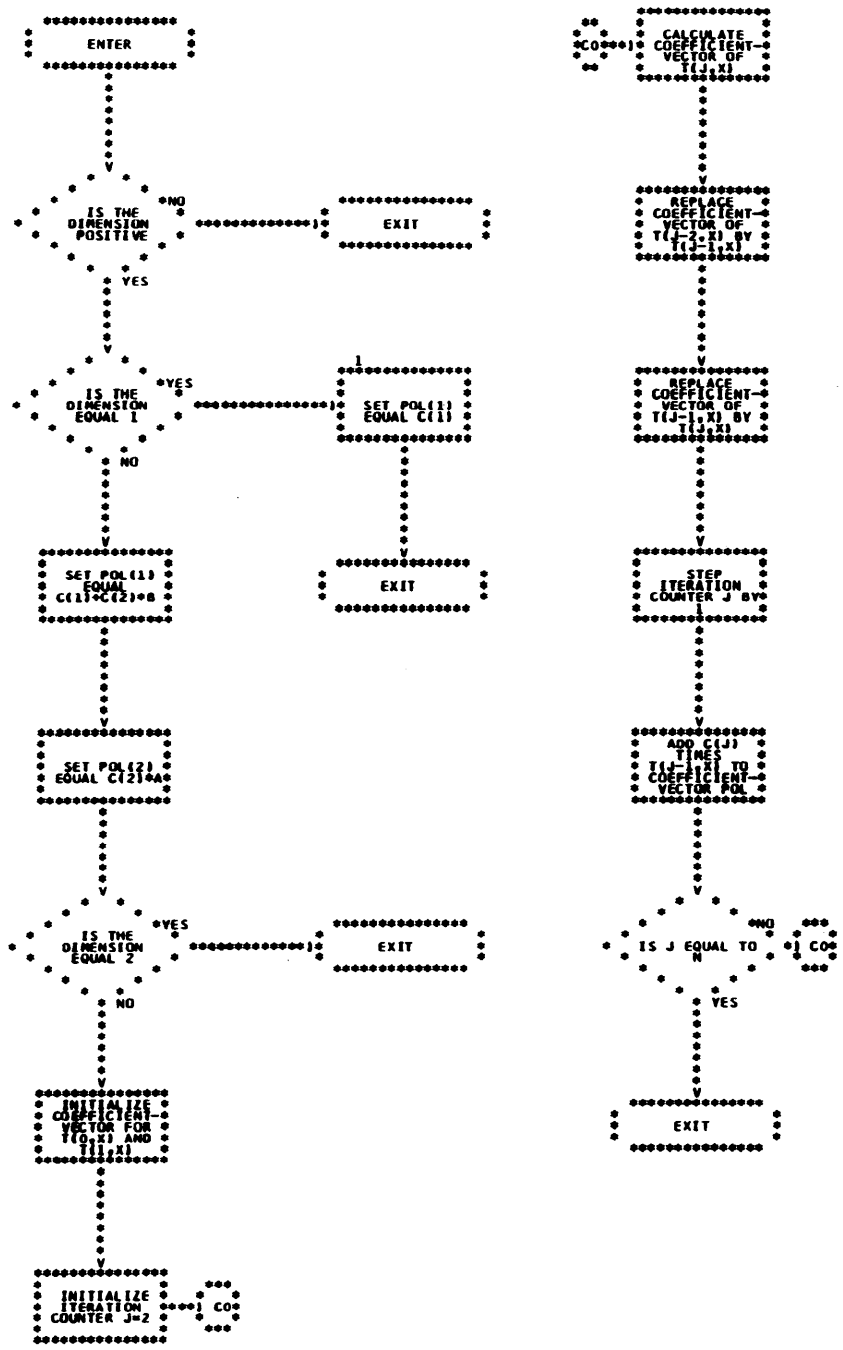

39
*A4
SET RESULTS Q(1)=01, Q(2)=02, Q(3)=A, Q(4)=B
EXIT

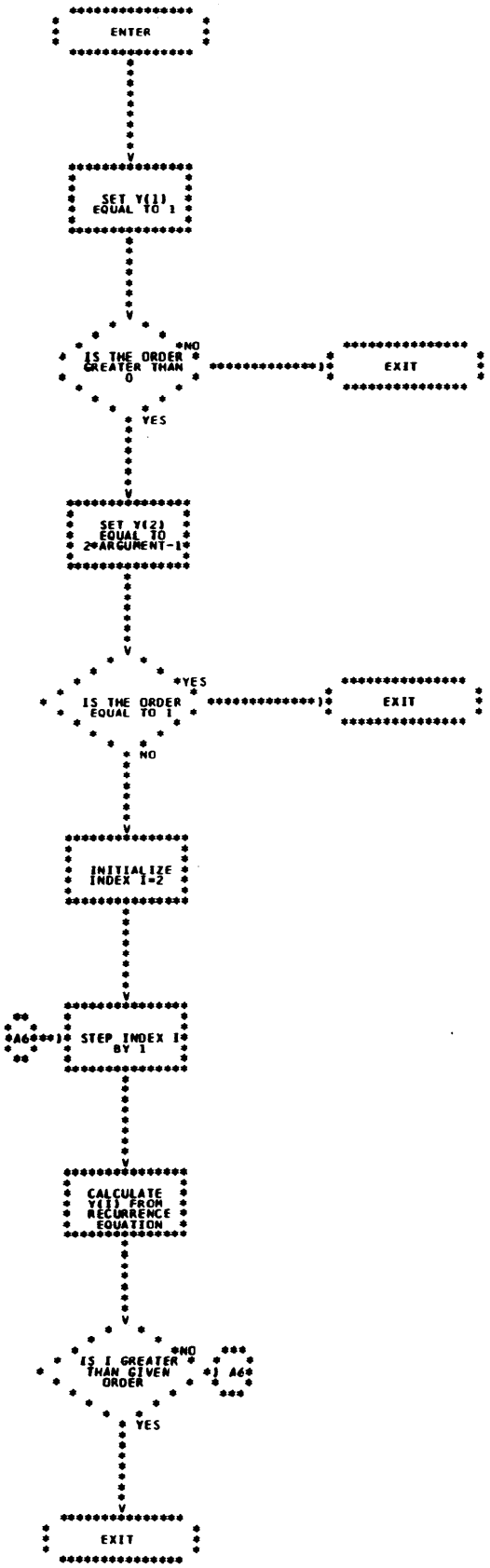

40
*A6
SET ERROR PARAMETER IER=-1
EXIT

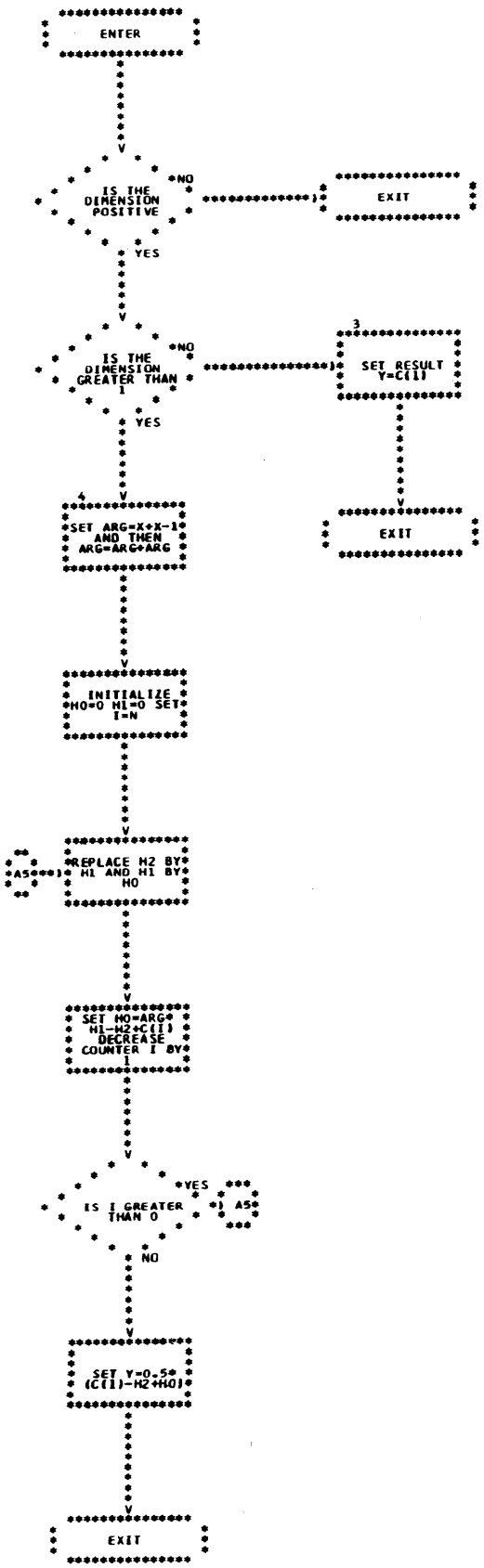

41
*A8
SET ERROR PARAMETER IER=-2
EXIT

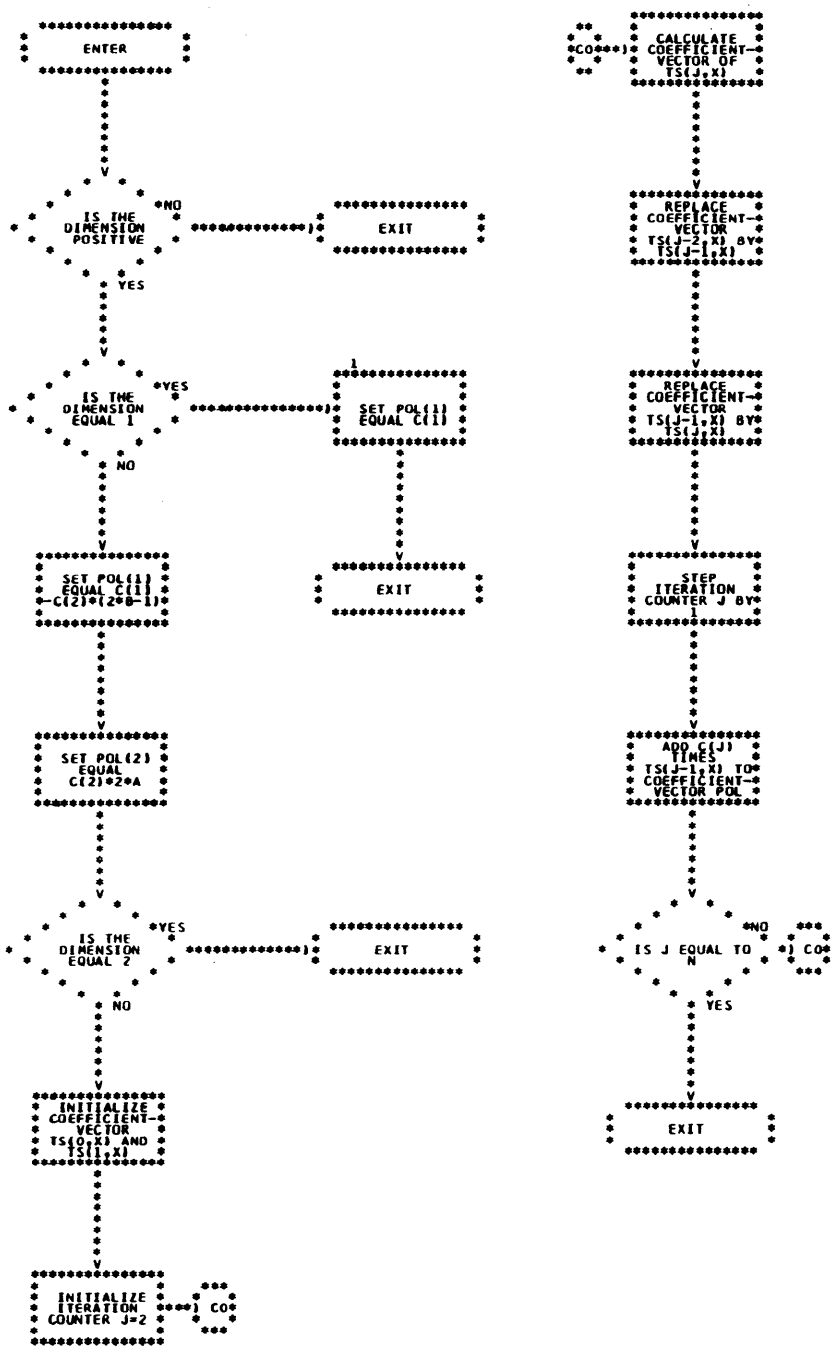

ENTER
SET Y(1) EQUAL TO 1
IS ORDER GREATER THAN 0
NO
EXIT
YES
SET Y(2) EQUAL TO GIVEN ARGUMENT
IS THE ORDER EQUAL TO 1
YES
EXIT
NO
INITIALIZE INDEX I=2
A6
STEP INDEX I BY 1
CALCULATE Y(I) FROM RECURRENCE EQUATION
IS I GREATER THAN GIVEN ORDER
NO
A6
YES
EXIT

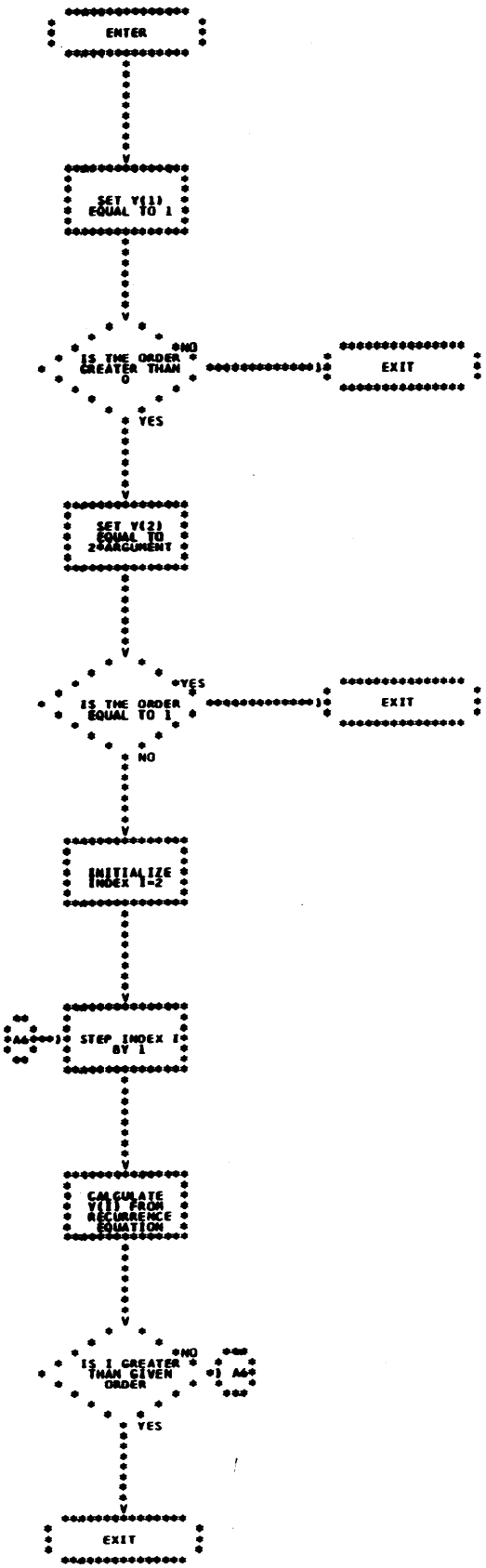

ENTER
IS THE DIMENSION POSITIVE
NO
EXIT
YES
IS THE DIMENSION GREATER THAN I
NO
3
SET RESULT Y=C(I)
YES
4
SET ARG=X+X
EXIT
INITIALIZE NO=0 N1=0 SET I=N
REPLACE N2 BY N1 AND N1 BY NO
AS
SET NO=ARG*N1 -N2*C(I) DECREASE COUNTER I BY 1
IS I GREATER THAN 0
YES
I AS
NO
SET Y=0.5* (C(1)-N2*NO)
EXIT

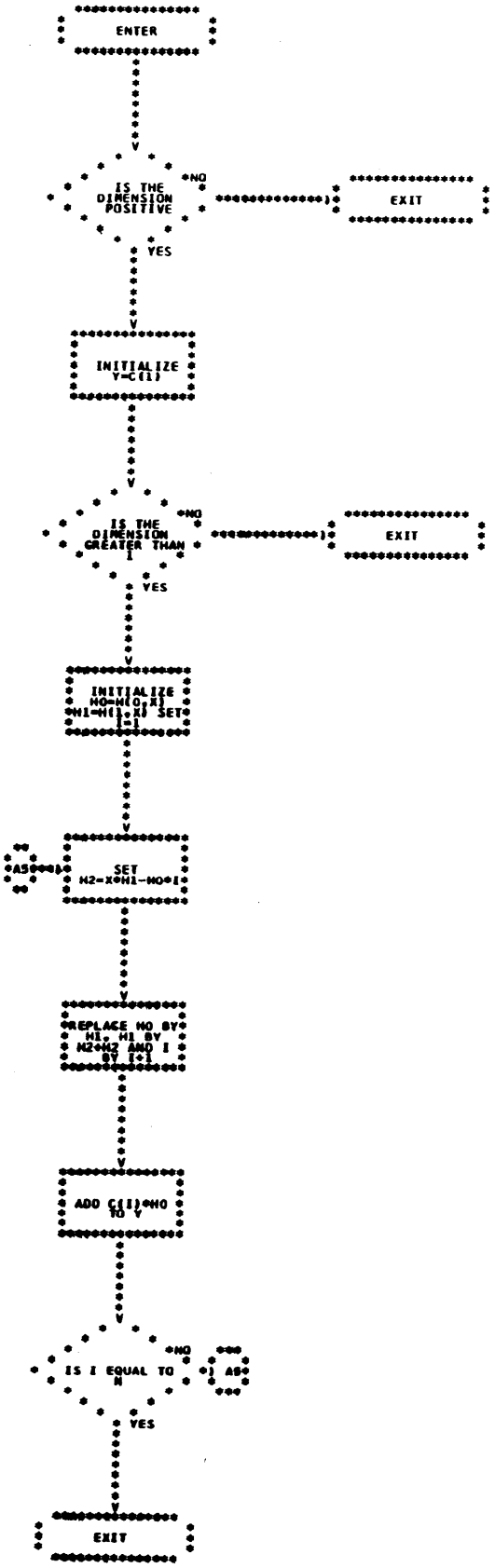

ENTER
IS THE DIMENSION POSITIVE
NO
EXIT
YES
IS THE DIMENSION EQUAL 1
YES
1
SET POL(1) EQUAL C(1)
NO
EXIT
SET POL(1) EQUAL C(1)+C(2)*B
SET POL(2) EQUAL C(2)*A
IS THE DIMENSION EQUAL 2
YES
EXIT
NO
INITIALIZE COEFFICIENT-VECTOR FOR T(0,X) AND T(1,X)
INITIALIZE ITERATION COUNTER J=2
CO
CO
CALCULATE COEFFICIENT-VECTOR OF T(J,X)
REPLACE COEFFICIENT-VECTOR OF T(J-2,X) BY T(J-1,X)
REPLACE COEFFICIENT-VECTOR OF T(J-1,X) BY T(J,X)
STEP ITERATION COUNTER J BY 1
ADD C(J) TIMES T(J-1,X) TO COEFFICIENT-VECTOR POL
IS J EQUAL TO N
NO
CO
YES
EXIT

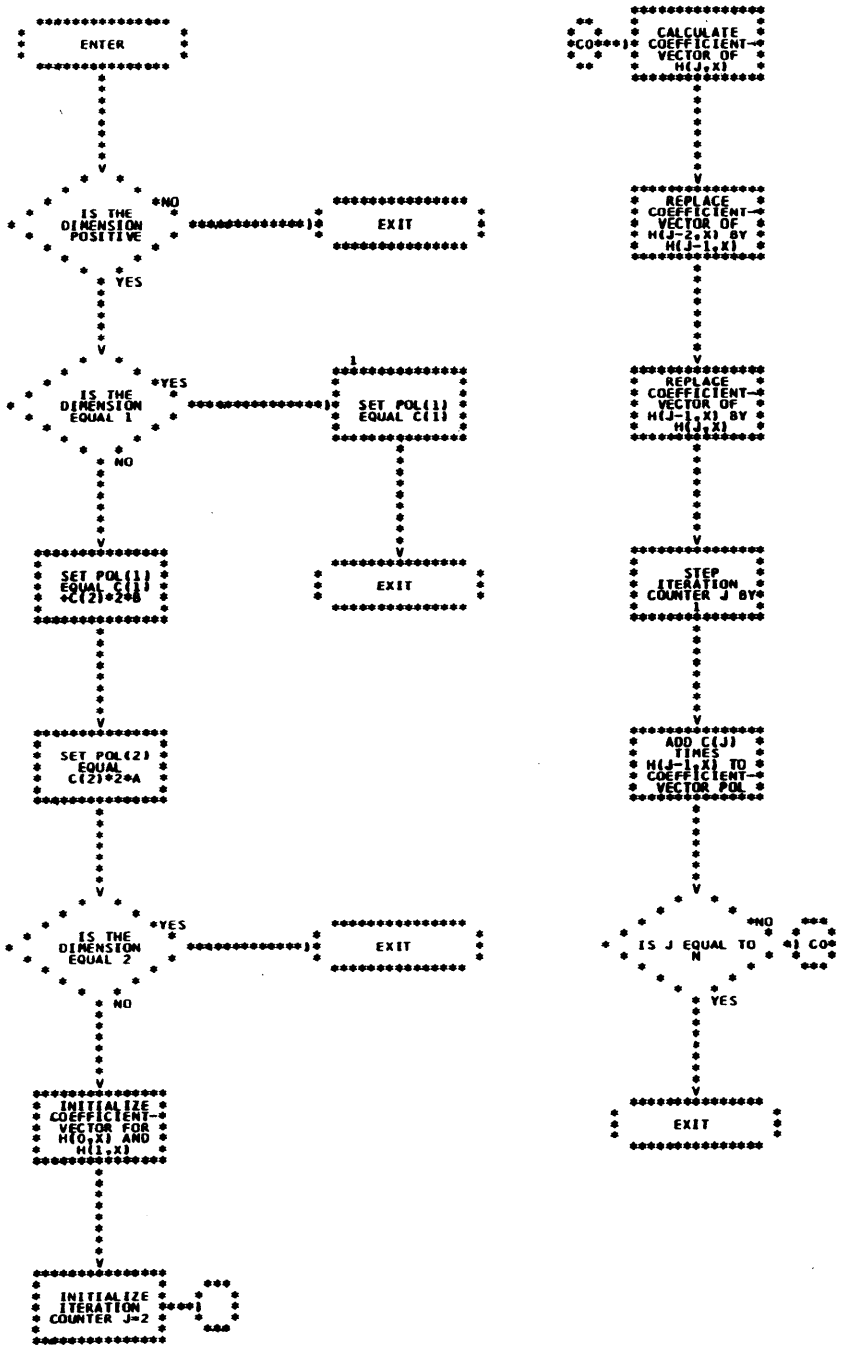

ENTER
SET Y(1) EQUAL TO 1
IS THE ORDER GREATER THAN 0
NO
EXIT
YES
SET Y(2) EQUAL TO 2*ARGUMENT-1
IS THE ORDER EQUAL TO 1
YES
EXIT
NO
INITIALIZE INDEX I=2
A6
STEP INDEX I BY 1
CALCULATE Y(I) FROM RECURRENCE EQUATION
IS I GREATER THAN GIVEN ORDER
NO
A6
YES
EXIT

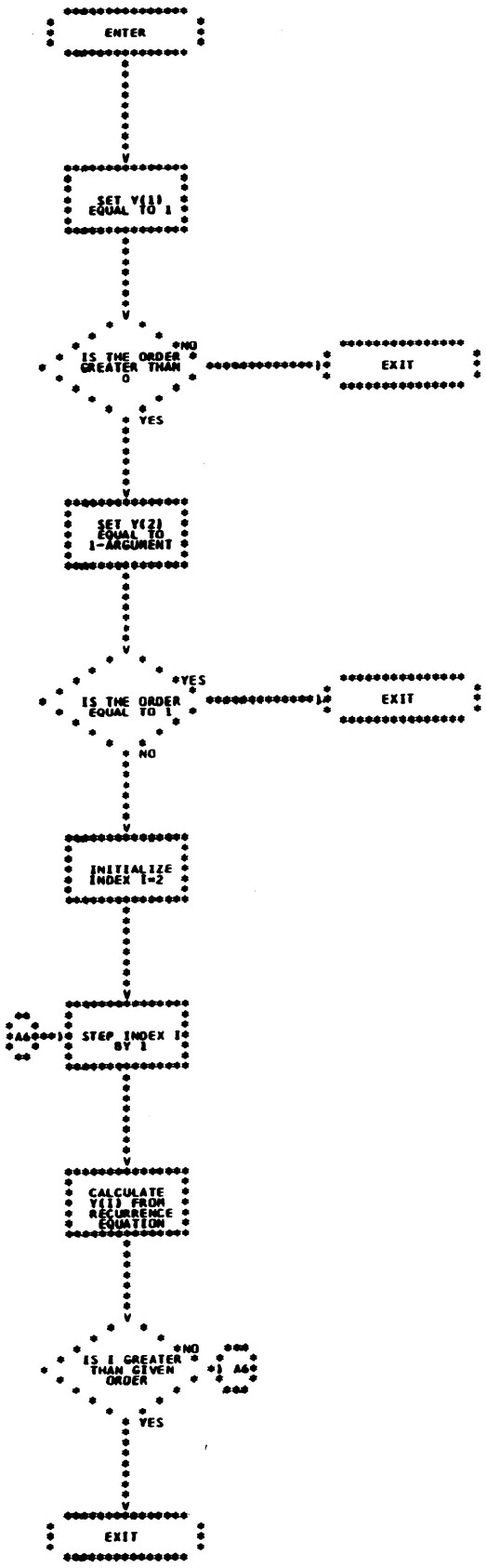

ENTER
IS THE DIMENSION POSITIVE
NO
EXIT
YES
3
IS THE DIMENSION GREATER THAN 1
NO
SET RESULT Y=C(1)
YES
EXIT
4
SET ARG=X*X-1 AND THEN ARG=ARG*ARG
INITIALIZE H0=0 H1=0 SET I=N
A5
REPLACE H2 BY H1 AND H1 BY H0
SET H0=ARG* H1-H2+C(I) DECREASE COUNTER I BY 1
IS I GREATER THAN 0
YES
A5
NO
SET Y=0.5* (C(1)-H2+H0)
EXIT

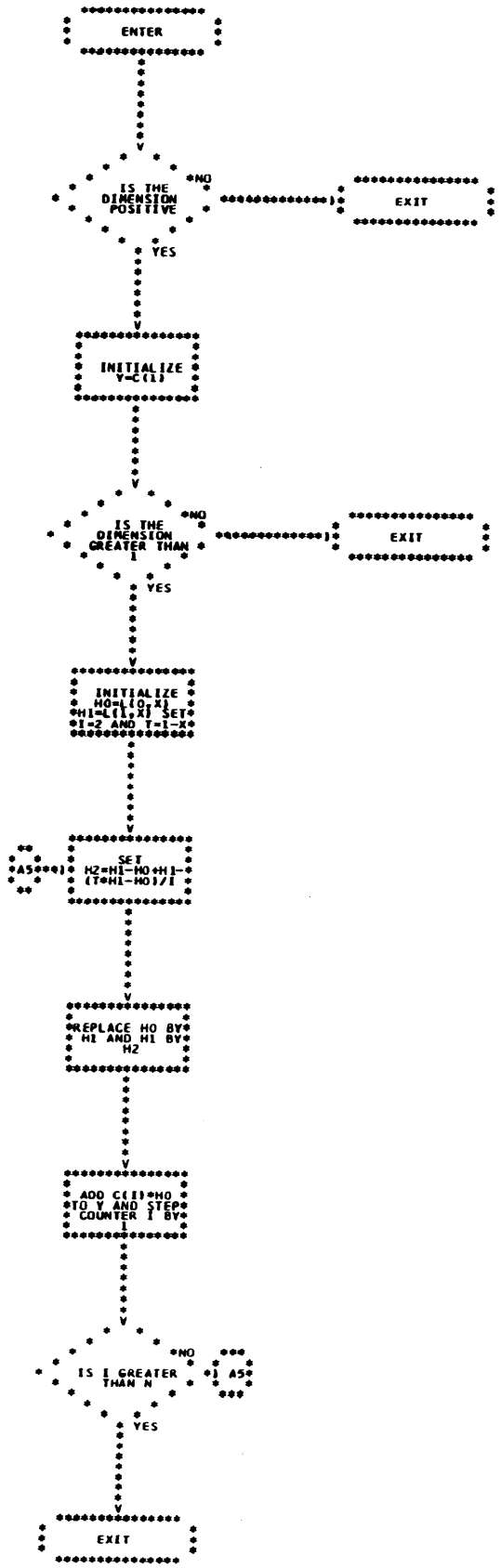

ENTER
IS THE DIMENSION POSITIVE
NO
EXIT
YES
IS THE DIMENSION EQUAL 1
YES
SET POL(1) EQUAL C(1)
NO
SET POL(1) EQUAL C(1) +C(2)*(2**B-1)
EXIT
SET POL(2) EQUAL C(2)*2**A
IS THE DIMENSION EQUAL 2
YES
EXIT
NO
INITIALIZE COEFFICIENT-VECTOR TS(0,X) AND TS(1,X)
INITIALIZE ITERATION COUNTER J=2
CO
CO
CALCULATE COEFFICIENT-VECTOR OF TS(J,X)
REPLACE COEFFICIENT-VECTOR TS(J-2,X) BY TS(J-1,X)
REPLACE COEFFICIENT-VECTOR TS(J-1,X) BY TS(J,X)
STEP ITERATION COUNTER J BY 1
ADD C(J) TIMES TS(J-1,X) TO COEFFICIENT-VECTOR POL
IS J EQUAL TO N
NO
CO
YES
EXIT

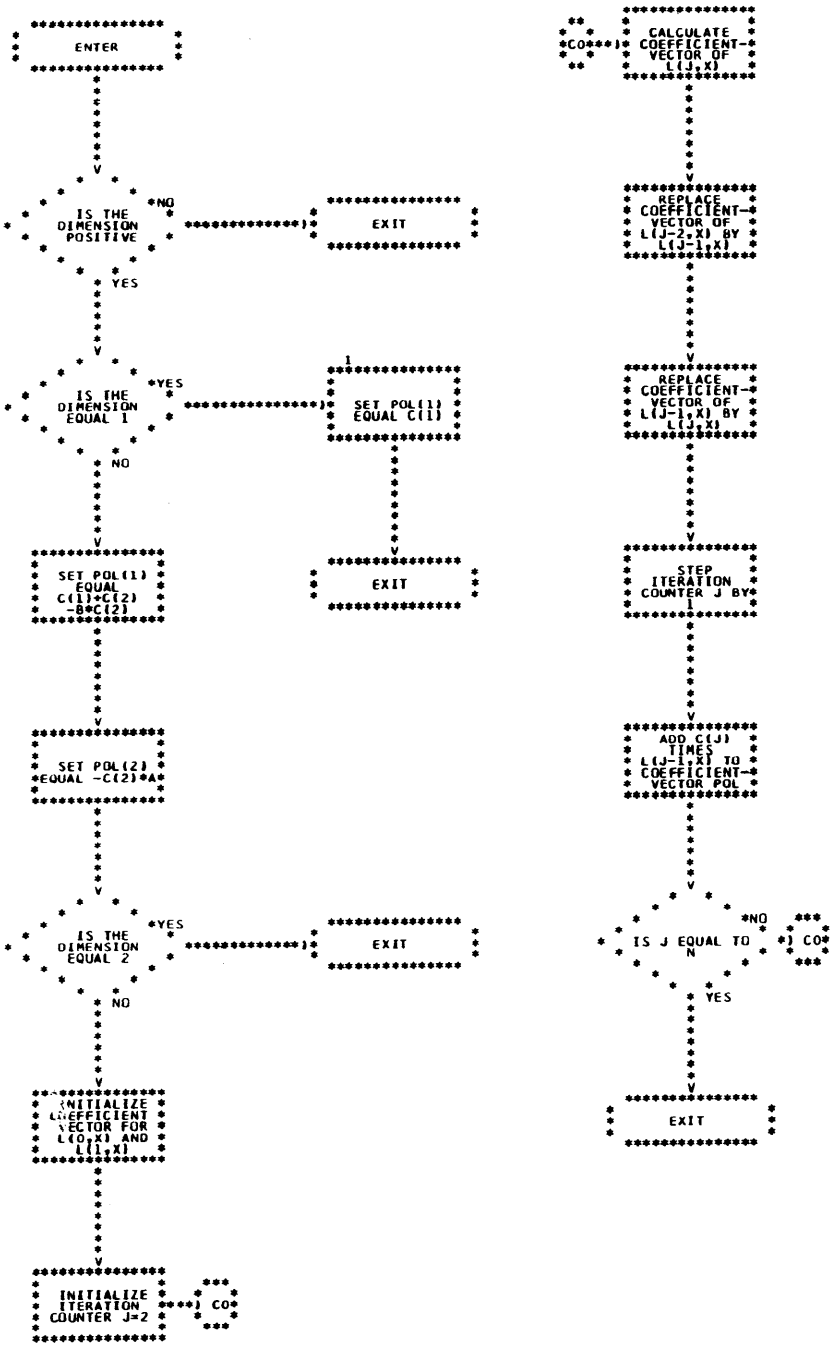

ENTER
SET Y(1) EQUAL TO 1
IS THE ORDER GREATER THAN 0
NO
EXIT
YES
SET Y(2) EQUAL TO 2**ARGUMENT
IS THE ORDER EQUAL TO 1
YES
EXIT
NO
INITIALIZE INDEX I=2
A6
STEP INDEX I BY 1
CALCULATE Y(I) FROM RECURRENCE EQUATION
IS I GREATER THAN GIVEN ORDER
NO
A6
YES
EXIT

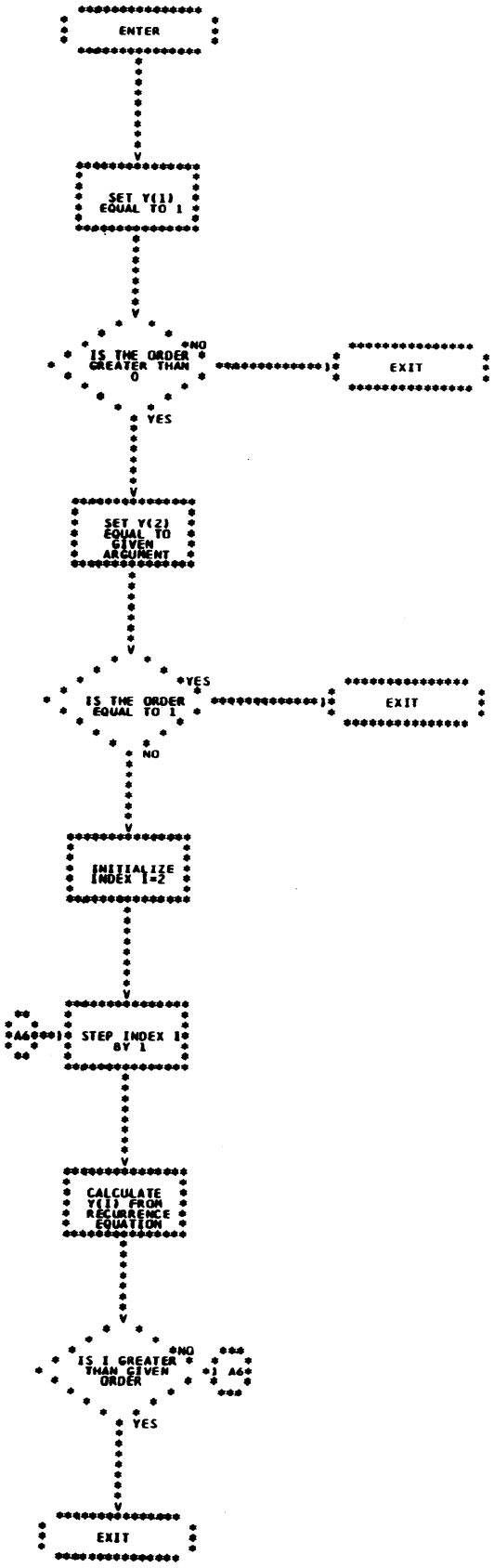

ENTER
IS THE DIMENSION POSITIVE
NO
EXIT
YES
INITIALIZE Y=C(1)
IS THE DIMENSION GREATER THAN I
NO
EXIT
YES
INITIALIZE H0=H(0,I) H1=H(1,I) SET I=1
AS
SET H2=X*H1-H0*I
REPLACE H0 BY H1, H1 BY H2+H2 AND I BY I+1
ADD C(I)*H0 TO Y
IS I EQUAL TO N
NO
AS
YES
EXIT

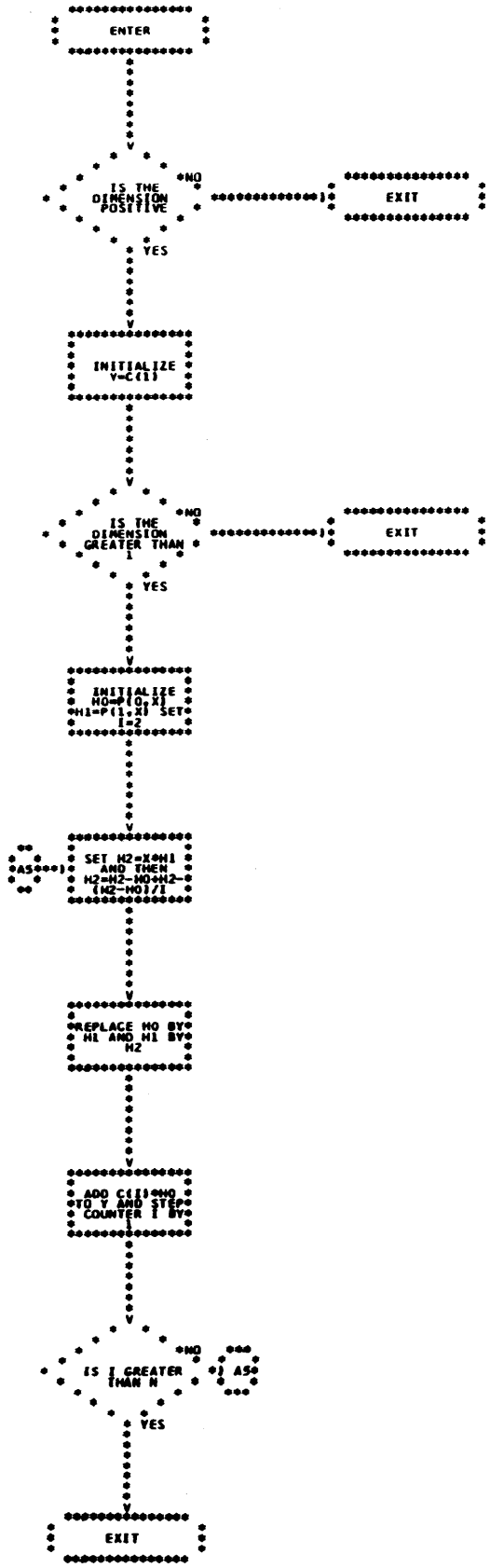

ENTER
IS THE DIMENSION POSITIVE
NO
EXIT
YES
IS THE DIMENSION EQUAL 1
YES
1
SET POL(1) EQUAL C(1)
NO
SET POL(1) EQUAL C(1) +C(2)*2+B
EXIT
SET POL(2) EQUAL C(2)*2+A
IS THE DIMENSION EQUAL 2
YES
EXIT
NO
INITIALIZE COEFFICIENT-VECTOR FOR H(0,X) AND H(1,X)
INITIALIZE ITERATION COUNTER J=2
CO
CALCULATE COEFFICIENT-VECTOR OF H(J,X)
REPLACE COEFFICIENT-VECTOR OF H(J-2,X) BY H(J-1,X)
REPLACE COEFFICIENT-VECTOR OF H(J-1,X) BY H(J,X)
STEP ITERATION COUNTER J BY 1
ADD C(J) TIMES H(J-1,X) TO COEFFICIENT-VECTOR POL
IS J EQUAL TO N
NO
CO
YES
EXIT

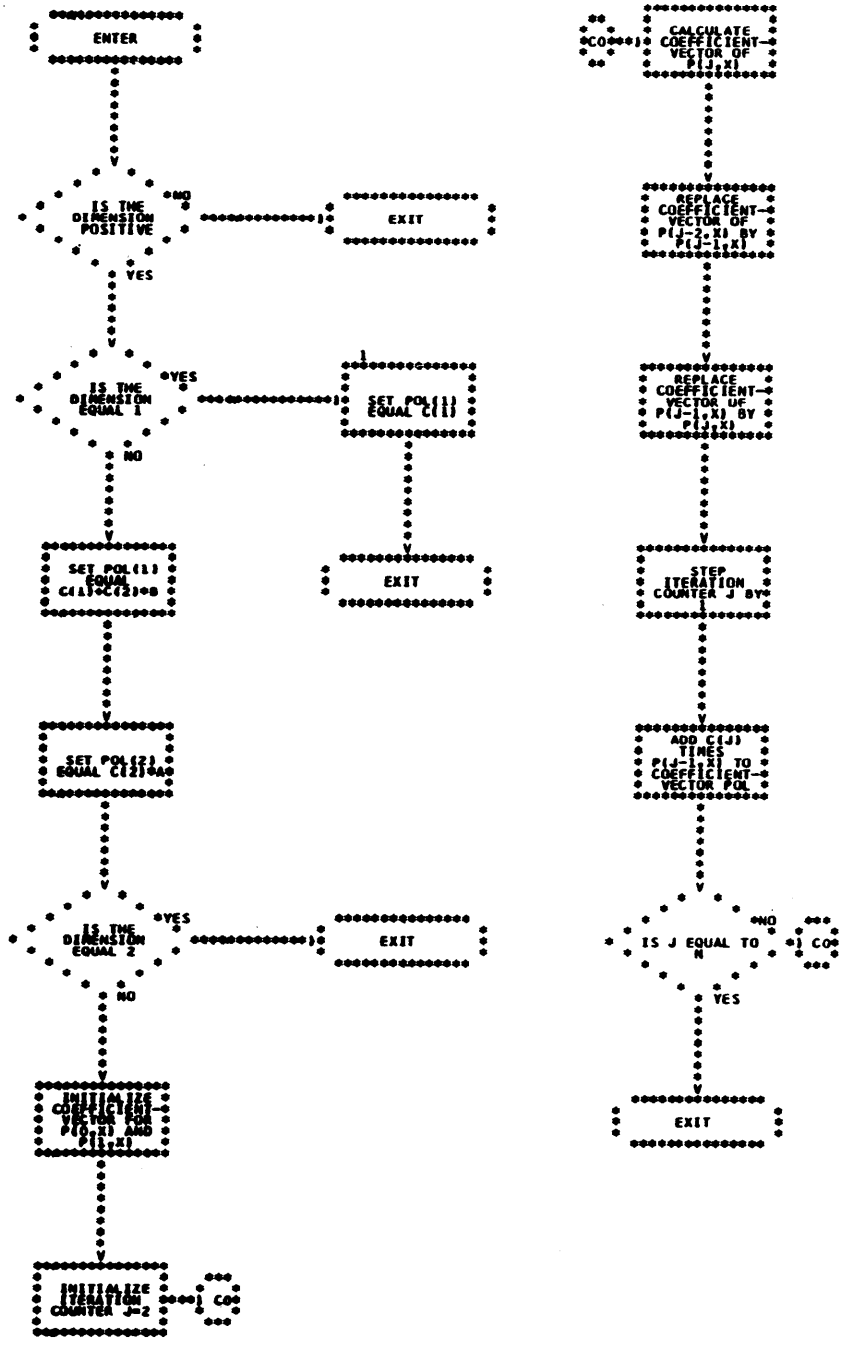

ENTER
SET Y(1) EQUAL TO 1
IS THE ORDER GREATER THAN 0
*NO
EXIT
YES
SET Y(2) EQUAL TO I-ARGUMENT
IS THE ORDER EQUAL TO 1
*YES
EXIT
NO
INITIALIZE INDEX I=2
A6
STEP INDEX I BY 1
CALCULATE Y(I) FROM RECURRENCE EQUATION
IS I GREATER THAN GIVEN ORDER
*NO
*I A6
YES
EXIT

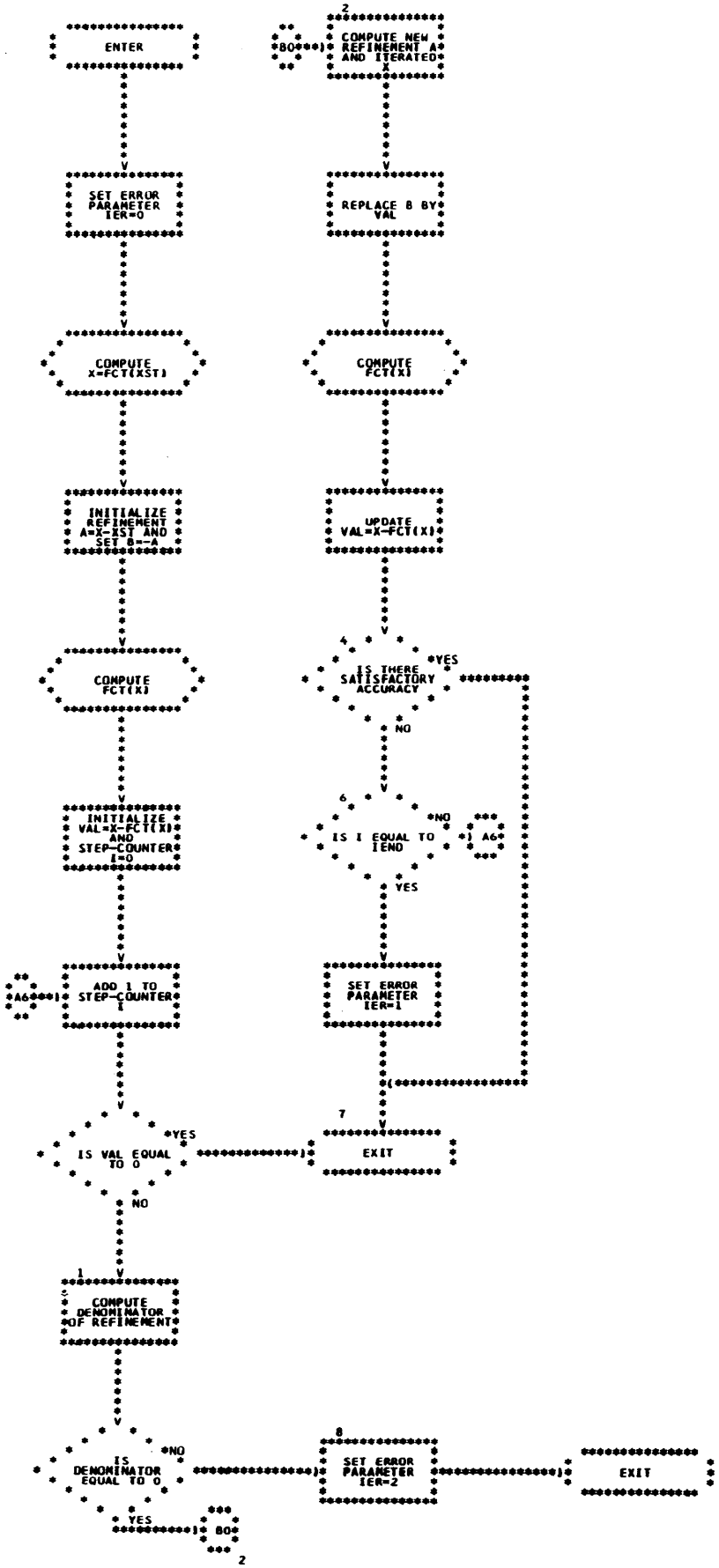

ENTER
IS THE DIMENSION POSITIVE
NO
EXIT
YES
INITIALIZE Y=C(1)
IS THE DIMENSION GREATER THAN 1
NO
EXIT
YES
INITIALIZE H0=L(1,X) H1=L(1,X) SET I=2 AND T=1-X
SET H2=H1-H0+H1-(T*H1-H0)/I
A5
REPLACE H0 BY H1 AND H1 BY H2
ADD C(I)*H0 TO Y AND STEP COUNTER I BY 1
IS I GREATER THAN N
NO
I A5
YES
EXIT

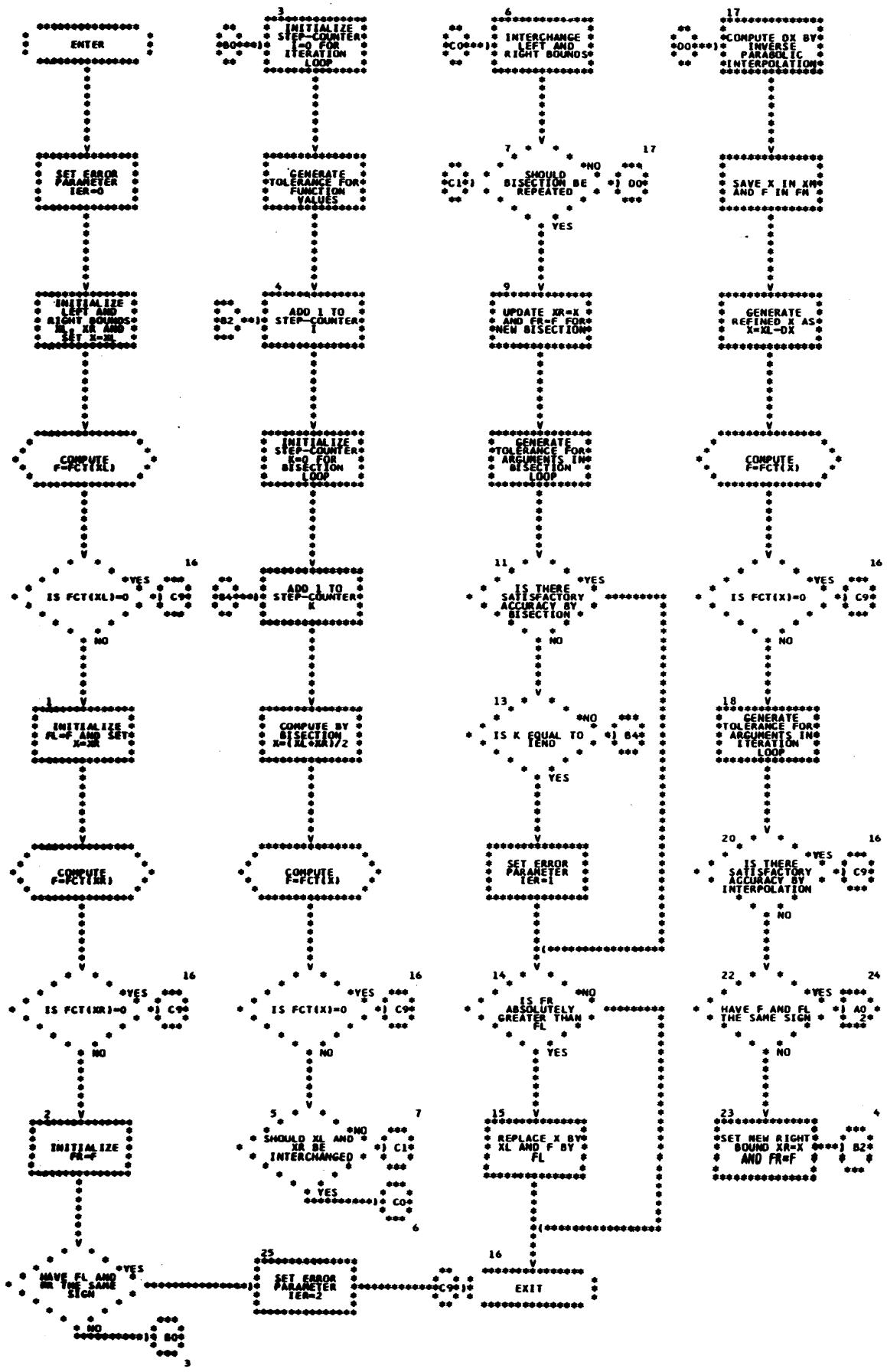

ENTER
IS THE DIMENSION POSITIVE
NO
EXIT
YES
IS THE DIMENSION EQUAL 1
YES
1
SET POL(1) EQUAL C(1)
EXIT
NO
SET POL(1) EQUAL C(1)+C(2) -B*C(2)
SET POL(2) EQUAL -C(2)*A
IS THE DIMENSION EQUAL 2
YES
EXIT
NO
INITIALIZE COEFFICIENT VECTOR FOR L(0,X) AND L(1,X)
INITIALIZE ITERATION COUNTER J=2
CO
CO
CALCULATE COEFFICIENT-VECTOR OF L(J,X)
REPLACE COEFFICIENT-VECTOR OF L(J-2,X) BY L(J-1,X)
REPLACE COEFFICIENT-VECTOR OF L(J-1,X) BY L(J,X)
STEP ITERATION COUNTER J BY 1
ADD C(J) TIMES L(J-1,X) TO COEFFICIENT-VECTOR POL
IS J EQUAL TO N
NO
CO
YES
EXIT

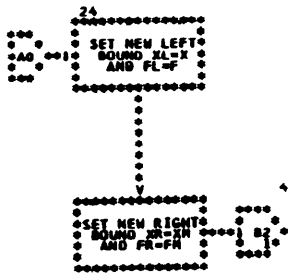

ENTER
SET Y(1) EQUAL TO 1
IS THE ORDER GREATER THAN 0
NO
EXIT
YES
SET Y(2) EQUAL TO GIVEN ARGUMENT
IS THE ORDER EQUAL TO 1
YES
EXIT
NO
INITIALIZE INDEX I=2
A6
STEP INDEX I BY 1
CALCULATE Y(I) FROM RECURRENCE EQUATION
IS I GREATER THAN GIVEN ORDER
NO
A6
YES
EXIT

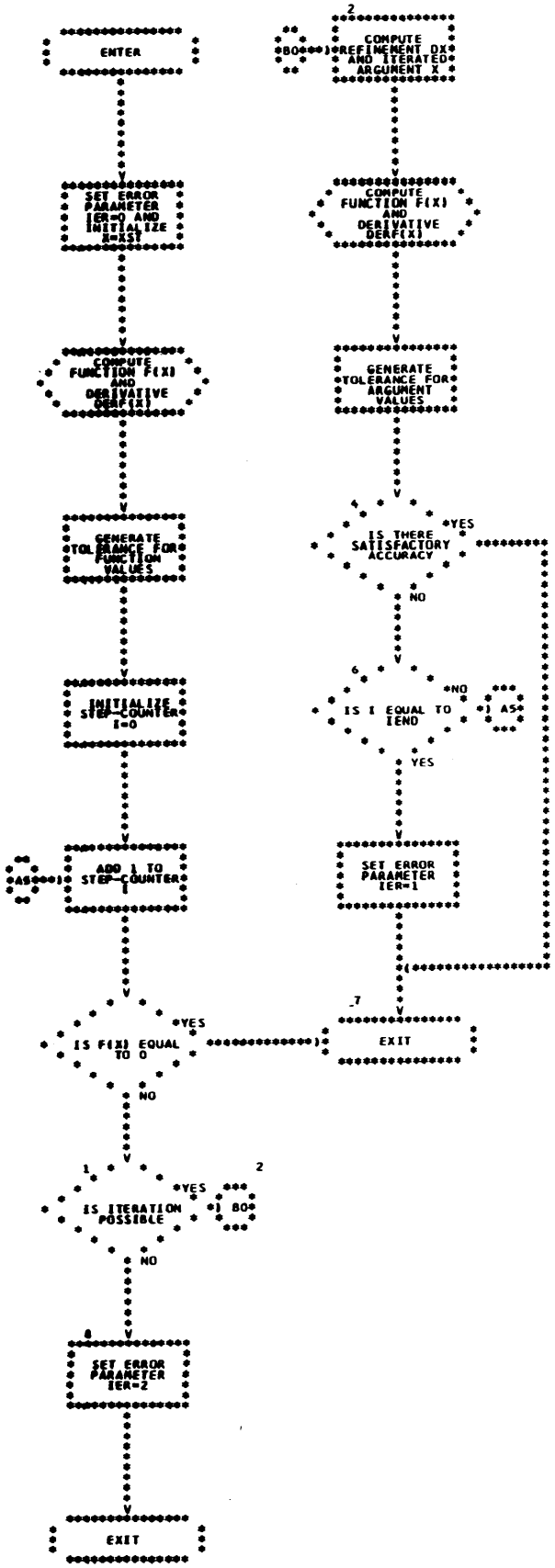

ENTER
IS THE DIMENSION POSITIVE
NO
EXIT
YES
INITIALIZE Y=C(1)
IS THE DIMENSION GREATER THAN 1
NO
EXIT
YES
INITIALIZE H0=P(0,X) H1=P(1,X) SET I=2
SET H2=X*H1 AND THEN H2=H2-H0*H2-(H2-H0)/I
AS
REPLACE H0 BY H1 AND H1 BY H2
ADD C(I)*H0 TO Y AND STEP COUNTER I BY 1
IS I GREATER THAN N
NO
AS
YES
EXIT

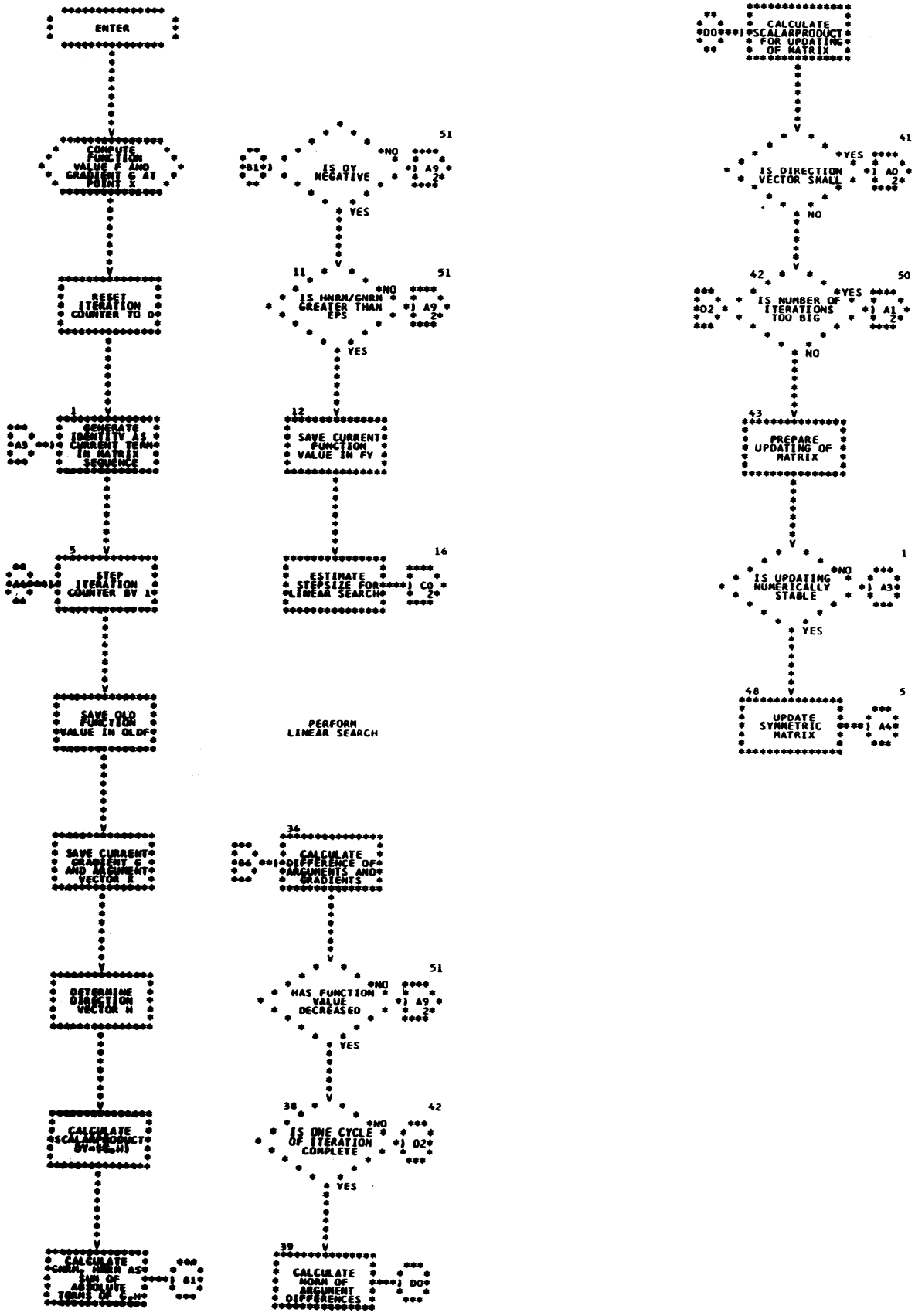

ENTER
IS THE DIMENSION POSITIVE
NO
EXIT
YES
IS THE DIMENSION EQUAL 1
YES
SET POL(1) EQUAL C(1)
NO
EXIT
SET POL(1) EQUAL C(1)+C(2)*B
SET POL(2) EQUAL C(2)*A
IS THE DIMENSION EQUAL 2
YES
EXIT
NO
INITIALIZE COEFFICIENT-VECTOR FOR P(0,X) AND P(1,X)
INITIALIZE ITERATION COUNTER J=2
CO
CO
CALCULATE COEFFICIENT-VECTOR OF P(J,X)
REPLACE COEFFICIENT-VECTOR OF P(J-2,X) BY P(J-1,X)
REPLACE COEFFICIENT-VECTOR OF P(J-1,X) BY P(J,X)
STEP ITERATION COUNTER J BY 1
ADD C(J) TIMES P(J-1,X) TO COEFFICIENT-VECTOR POL
IS J EQUAL TO N
NO
CO
YES
EXIT

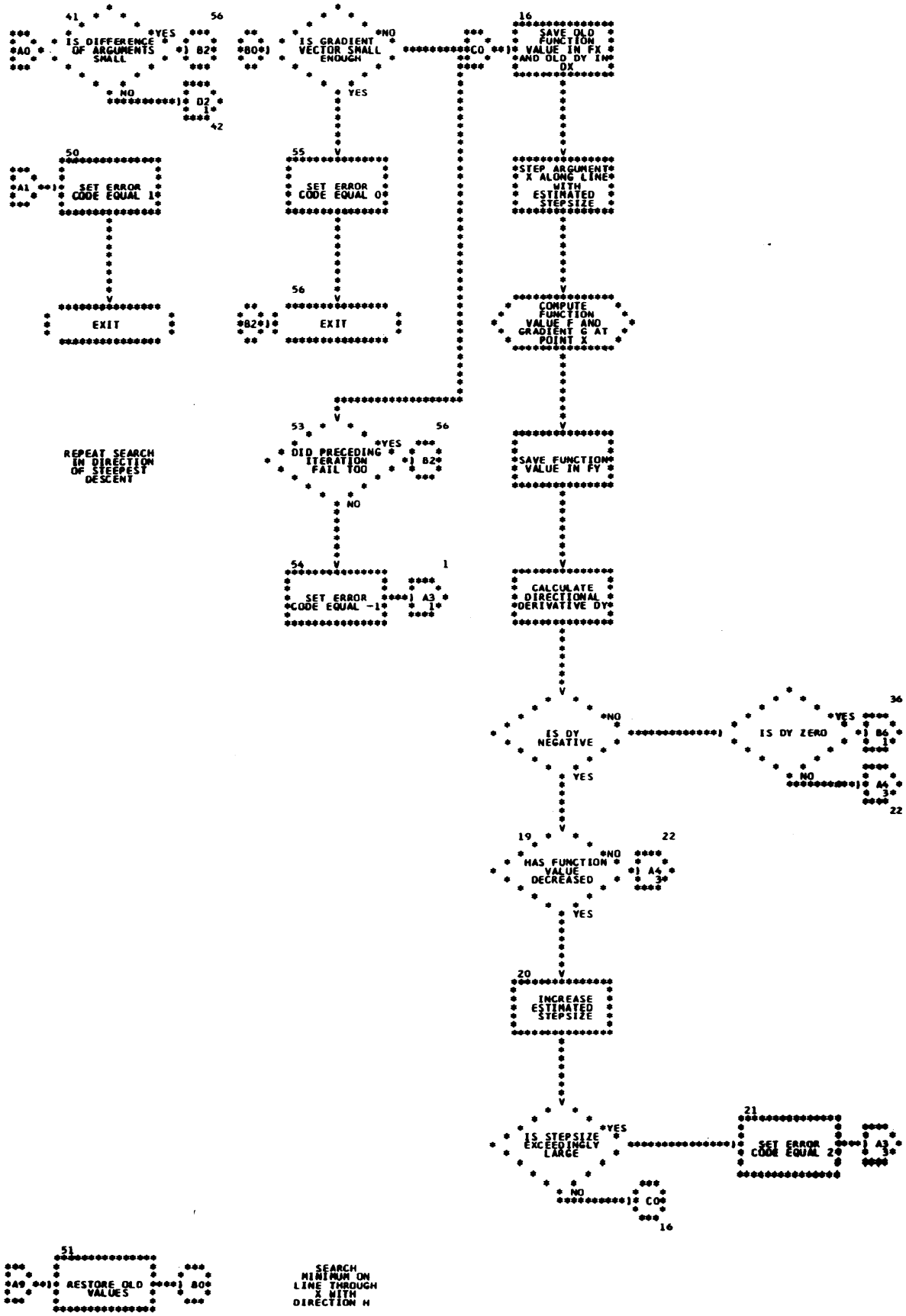

ENTER
SET ERROR PARAMETER IER=0
COMPUTE X=FCT(XST)
INITIALIZE REFINEMENT A=Y-XST AND SET B=-A
COMPUTE FCT(X)
INITIALIZE VAL=X-FCT(X) AND STEP-COUNTER I=0
ADD 1 TO STEP-COUNTER I
IS VAL EQUAL TO 0
YES
NO
COMPUTE DENOMINATOR OF REFINEMENT
IS DENOMINATOR EQUAL TO 0
NO
YES
SET ERROR PARAMETER IER=2
EXIT
2
COMPUTE NEW REFINEMENT A AND ITERATED X
REPLACE B BY VAL
COMPUTE FCT(X)
UPDATE VAL=X-FCT(X)
4
IS THERE SATISFACTORY ACCURACY
YES
NO
6
IS I EQUAL TO IEND
NO
A6
YES
SET ERROR PARAMETER IER=1
7
EXIT
1
2
8
EXIT

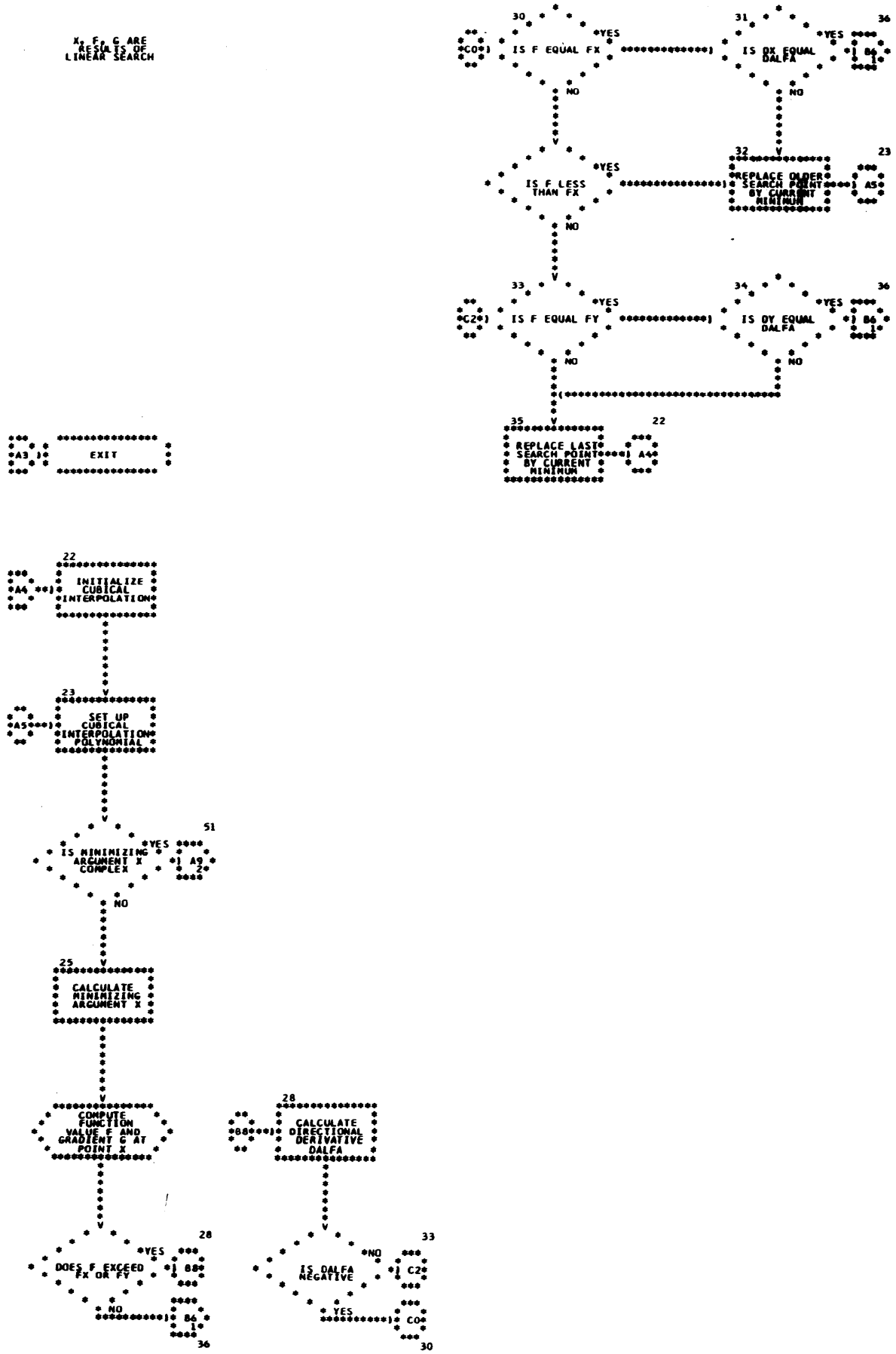

ENTER
SET ERROR PARAMETER IER=0
INITIALIZE LEFT AND RIGHT BOUNDS XL, XR AND SET X=XL
COMPUTE F=FCT(XL)
IS FCT(XL)=0
YES
C9
NO
INITIALIZE FL=F AND SET X=XR
COMPUTE F=FCT(XR)
IS FCT(XR)=0
YES
C9
NO
INITIALIZE FR=F
HAVE FL AND FR THE SAME SIGN
YES
NO
B0
3
INITIALIZE STEP-COUNTER I=0 FOR ITERATION LOOP
B0
GENERATE TOLERANCE FOR FUNCTION VALUES
ADD 1 TO STEP-COUNTER I
B2
INITIALIZE STEP-COUNTER K=0 FOR BISECTION LOOP
ADD 1 TO STEP-COUNTER K
B4
COMPUTE BY BISECTION X=(XL+XR)/2
COMPUTE F=FCT(X)
IS FCT(X)=0
YES
C9
NO
SHOULD XL AND XR BE INTERCHANGED
YES
C0
NO
C1
6
INTERCHANGE LEFT AND RIGHT BOUNDS
C0
SHOULD BISECTION BE REPEATED
NO
YES
C1
D0
9
UPDATE XR=X AND FR=F FOR NEW BISECTION
GENERATE TOLERANCE FOR ARGUMENTS IN BISECTION LOOP
IS THERE SATISFACTORY ACCURACY BY BISECTION
YES
NO
C9
IS K EQUAL TO IEND
NO
B4
YES
SET ERROR PARAMETER IER=1
IS FR ABSOLUTELY GREATER THAN FL
NO
YES
C9
REPLACE X BY XL AND F BY FL
SET ERROR PARAMETER IER=2
C9
EXIT
17
COMPUTE DX BY INVERSE PARABOLIC INTERPOLATION
D0
SAVE X IN XM AND F IN FM
GENERATE REFINED X AS X=XL-DX
COMPUTE F=FCT(X)
IS FCT(X)=0
YES
C9
NO
GENERATE TOLERANCE FOR ARGUMENTS IN ITERATION LOOP
IS THERE SATISFACTORY ACCURACY BY INTERPOLATION
YES
C9
NO
HAVE F AND FL THE SAME SIGN
YES
A0
A2
NO
SET NEW RIGHT BOUND XR=X AND FR=F
B2
4

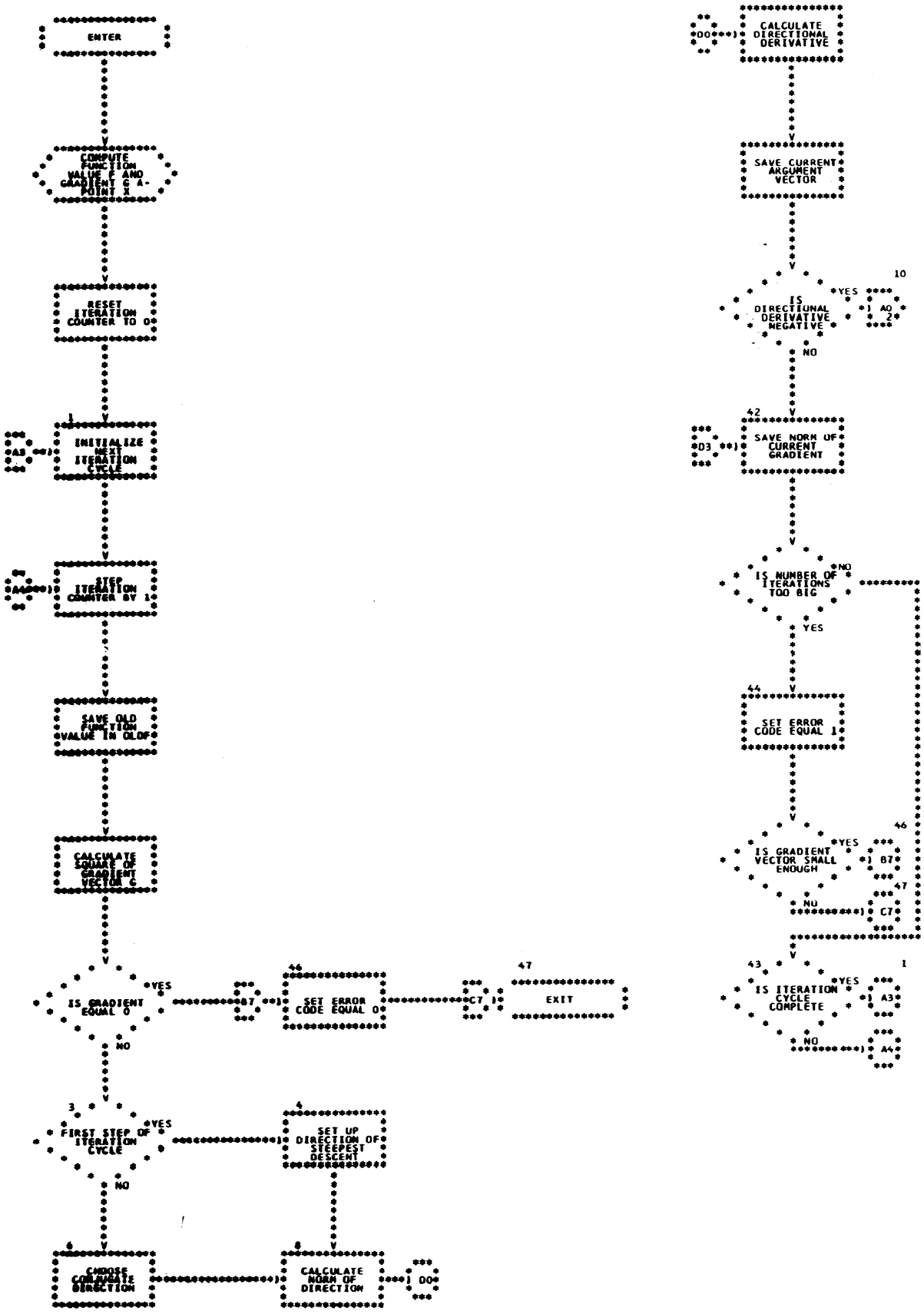

24
SET NEW LEFT
BOUND XL=X
AND FL=F
A0
SET NEW RIGHT
BOUND XR=XN
AND FR=FN
B2
4

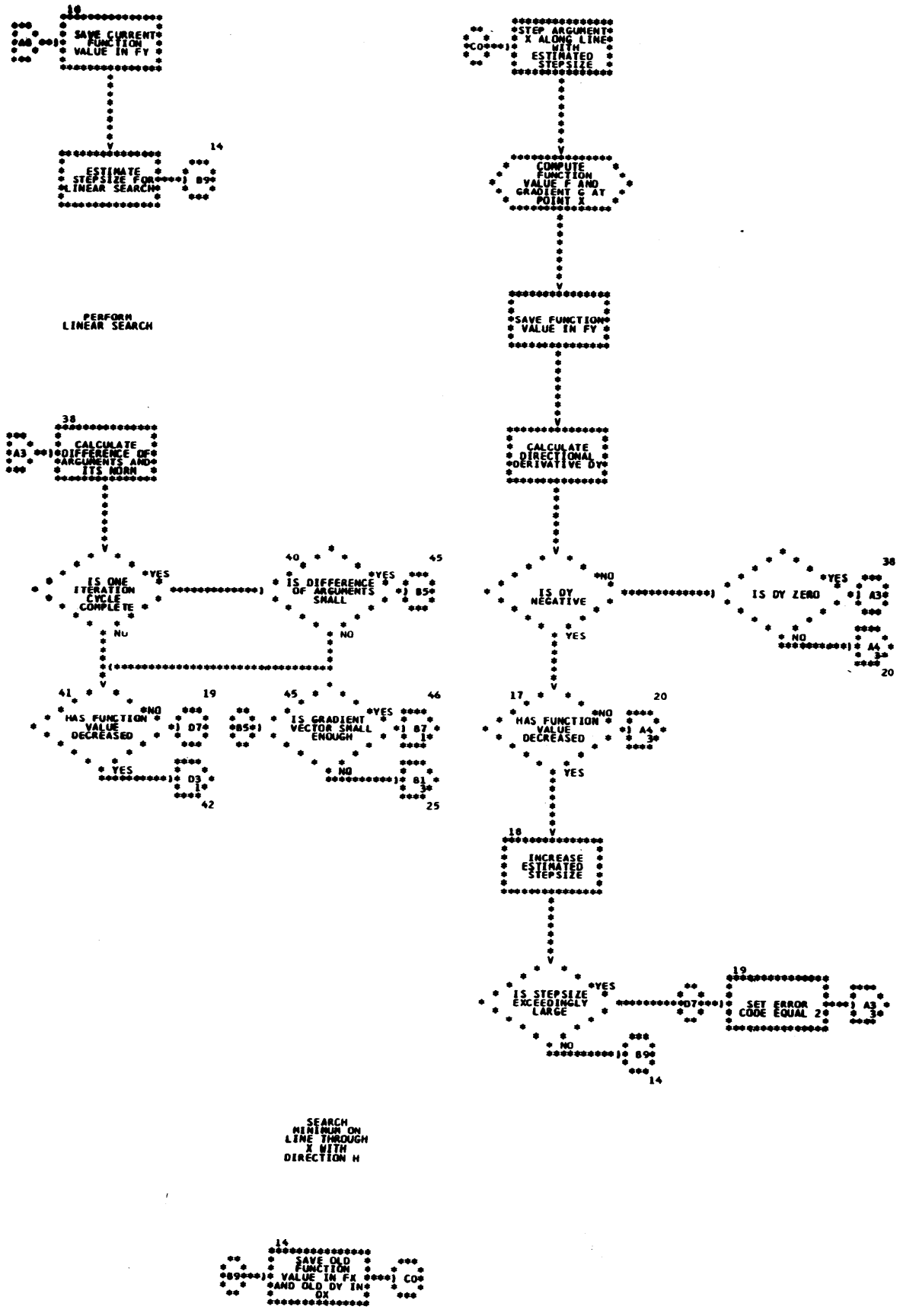

ENTER
SET ERROR PARAMETER IER=0 AND INITIALIZE X=XST
COMPUTE FUNCTION F(X) AND DERIVATIVE DERF(X)
GENERATE TOLERANCE FOR FUNCTION VALUES
INITIALIZE STEP-COUNTER I=0
ADD 1 TO STEP-COUNTER I
IS F(X) EQUAL TO 0
YES
NO
IS ITERATION POSSIBLE
YES
NO
SET ERROR PARAMETER IER=2
EXIT
B0
COMPUTE REFINEMENT DX AND ITERATED ARGUMENT X
COMPUTE FUNCTION F(X) AND DERIVATIVE DERF(X)
GENERATE TOLERANCE FOR ARGUMENT VALUES
IS THERE SATISFACTORY ACCURACY
YES
NO
IS I EQUAL TO IEND
NO
A5
YES
SET ERROR PARAMETER IER=1
EXIT
A9
B0

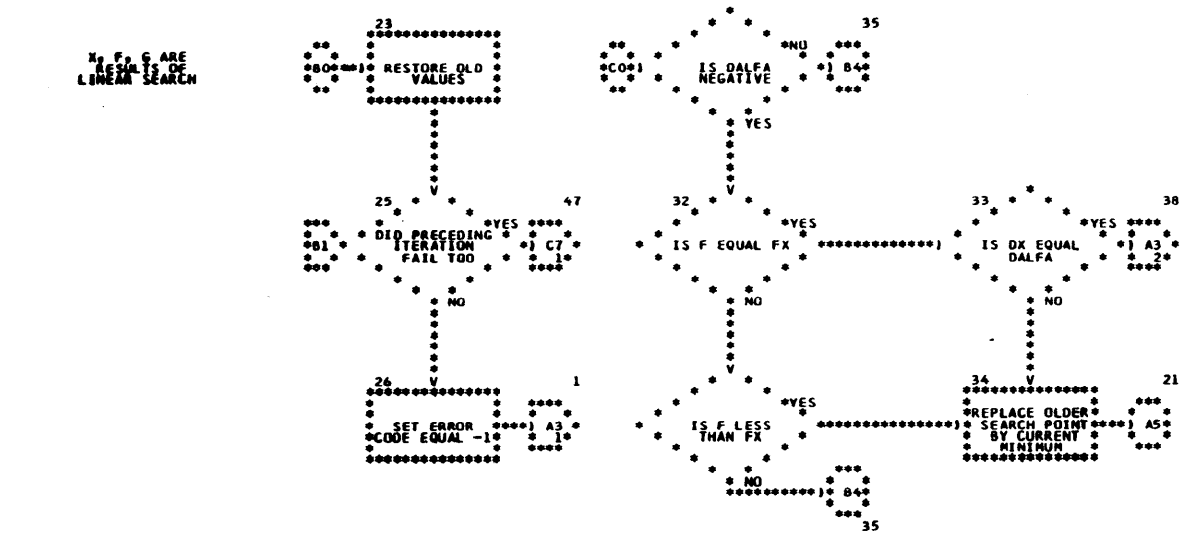

ENTER
COMPUTE FUNCTION VALUE F AND GRADIENT G AT POINT X
RESET ITERATION COUNTER TO 0
1
GENERATE IDENTITY AS CURRENT TERM IN MATRIX SEQUENCE
A3
5
STEP ITERATION COUNTER BY 1
SAVE OLD FUNCTION VALUE IN OLDF
SAVE CURRENT GRADIENT G AND ARGUMENT VECTOR X
DETERMINE DIRECTION VECTOR H
CALCULATE SCALARPRODUCT SV=(G,H)
CALCULATE GNRM, HNRM AS SUM OF ABSOLUTE TERMS OF G,H
B1
51
IS SV NEGATIVE
NO
A9
2
YES
11
51
IS HNRM/GNRM GREATER THAN EPS
NO
A9
2
YES
12
SAVE CURRENT FUNCTION VALUE IN FV
16
ESTIMATE STEPSIZE FOR LINEAR SEARCH
C0
2
PERFORM LINEAR SEARCH
34
CALCULATE DIFFERENCE OF ARGUMENTS AND GRADIENTS
B6
51
HAS FUNCTION VALUE DECREASED
NO
A9
2
YES
38
42
IS ONE CYCLE OF ITERATION COMPLETE
NO
D2
YES
39
CALCULATE NORM OF ARGUMENT DIFFERENCES
D0
DO
CALCULATE SCALARPRODUCT FOR UPDATING OF MATRIX
41
IS DIRECTION VECTOR SMALL
YES
A0
2
NO
42
50
IS NUMBER OF ITERATIONS TOO BIG
YES
A1
2
NO
43
PREPARE UPDATING OF MATRIX
1
IS UPDATING NUMERICALLY STABLE
NO
A3
YES
48
5
UPDATE SYMMETRIC MATRIX
A4


41
IS DIFFERENCE OF ARGUMENTS SMALL
YES
56
B2
NO
D2 1
42
A0
50
SET ERROR CODE EQUAL 1
A1
EXIT
IS GRADIENT VECTOR SMALL ENOUGH
NO
B0
YES
55
SET ERROR CODE EQUAL 0
56
B2
EXIT
CO
16
SAVE OLD FUNCTION VALUE IN FX AND OLD DY IN DX
STEP ARGUMENT X ALONG LINE WITH ESTIMATED STEPSIZE
COMPUTE FUNCTION VALUE F AND GRADIENT G AT POINT X
SAVE FUNCTION VALUE IN FY
CALCULATE DIRECTIONAL DERIVATIVE DY
IS DY NEGATIVE
NO
YES
IS DY ZERO
YES
36
B6 1
NO
A4 3
22
HAS FUNCTION VALUE DECREASED
19
NO
22
A4 3
YES
INCREASE ESTIMATED STEPSIZE
20
IS STEPSIZE EXCEEDINGLY LARGE
YES
21
SET ERROR CODE EQUAL 2
A3 3
NO
CO
16
DID PRECEDING ITERATION FAIL TOO
53
YES
56
B2
NO
SET ERROR CODE EQUAL -1
54
A3 1
1
REPEAT SEARCH IN DIRECTION OF STEEPEST DESCENT
RESTORE OLD VALUES
51
A9
B0
SEARCH MINIMUM ON LINE THROUGH X WITH DIRECTION H

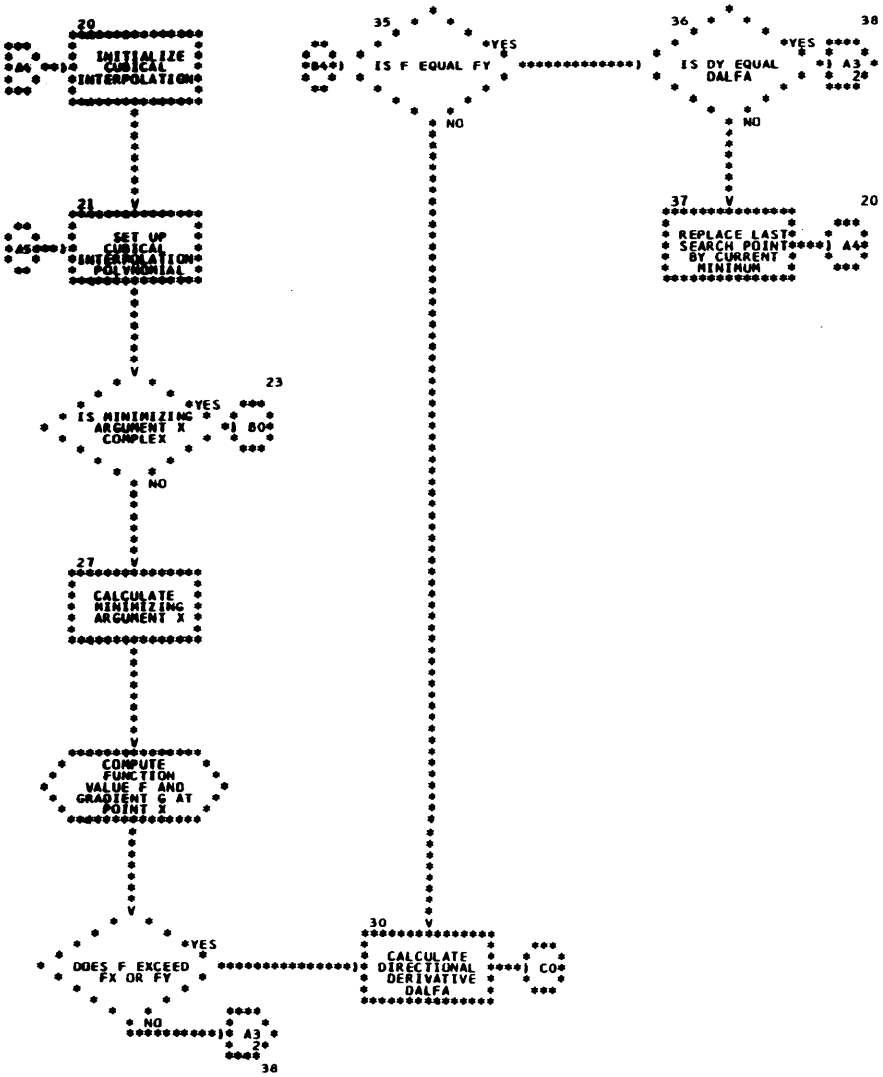

X, F, G ARE RESULTS OF LINEAR SEARCH
30
IS F EQUAL FX
YES
NO
C0
31
IS DX EQUAL DALFA
YES
NO
36
B6
1
32
REPLACE OLDER SEARCH POINT BY CURRENT MINIMUM
23
A5
IS F LESS THAN FX
YES
NO
33
IS F EQUAL FY
YES
NO
C2
34
IS DY EQUAL DALFA
YES
NO
36
B6
1
35
REPLACE LAST SEARCH POINT BY CURRENT MINIMUM
22
A4
A3
EXIT
22
INITIALIZE CUBICAL INTERPOLATION
A4
23
SET UP CUBICAL INTERPOLATION POLYNOMIAL
A5
IS MINIMIZING ARGUMENT X COMPLEX
YES
51
A9
2
NO
25
CALCULATE MINIMIZING ARGUMENT X
COMPUTE FUNCTION VALUE F AND GRADIENT G AT POINT X
28
CALCULATE DIRECTIONAL DERIVATIVE DALFA
B8
DOES F EXCEED FX OR FY
YES
28
B8
NO
B6
1
36
IS DALFA NEGATIVE
NO
33
C2
YES
C0
30

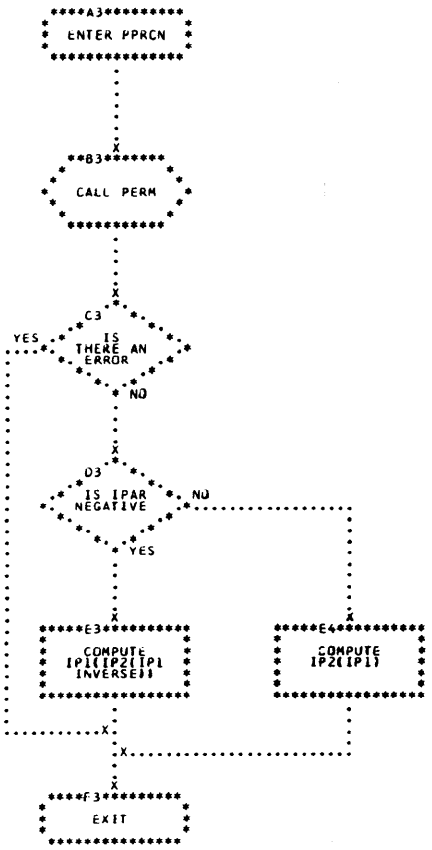

ENTER
COMPUTE FUNCTION VALUE F AND GRADIENT G A-POINT X
RESET ITERATION COUNTER TO 0
1
INITIALIZE NEXT ITERATION CYCLE
A5
STEP ITERATION COUNTER BY 1
A4
SAVE OLD FUNCTION VALUE IN OLDF
CALCULATE SQUARE OF GRADIENT VECTOR G
IS GRADIENT EQUAL 0
YES
B7
NO
44
SET ERROR CODE EQUAL 0
C7
47
EXIT
3
FIRST STEP OF ITERATION CYCLE
YES
4
SET UP DIRECTION OF STEEPEST DESCENT
NO
6
CHOOSE CONJUGATE DIRECTION
8
CALCULATE NORM OF DIRECTION
D0
CALCULATE DIRECTIONAL DERIVATIVE
SAVE CURRENT ARGUMENT VECTOR
10
IS DIRECTIONAL DERIVATIVE NEGATIVE
YES
A0
2
NO
42
SAVE NORM OF CURRENT GRADIENT
D3
IS NUMBER OF ITERATIONS TOO BIG
NO
YES
44
SET ERROR CODE EQUAL 1
46
IS GRADIENT VECTOR SMALL ENOUGH
YES
B7
NO
47
C7
43
IS ITERATION CYCLE COMPLETE
YES
1
A3
NO
A4

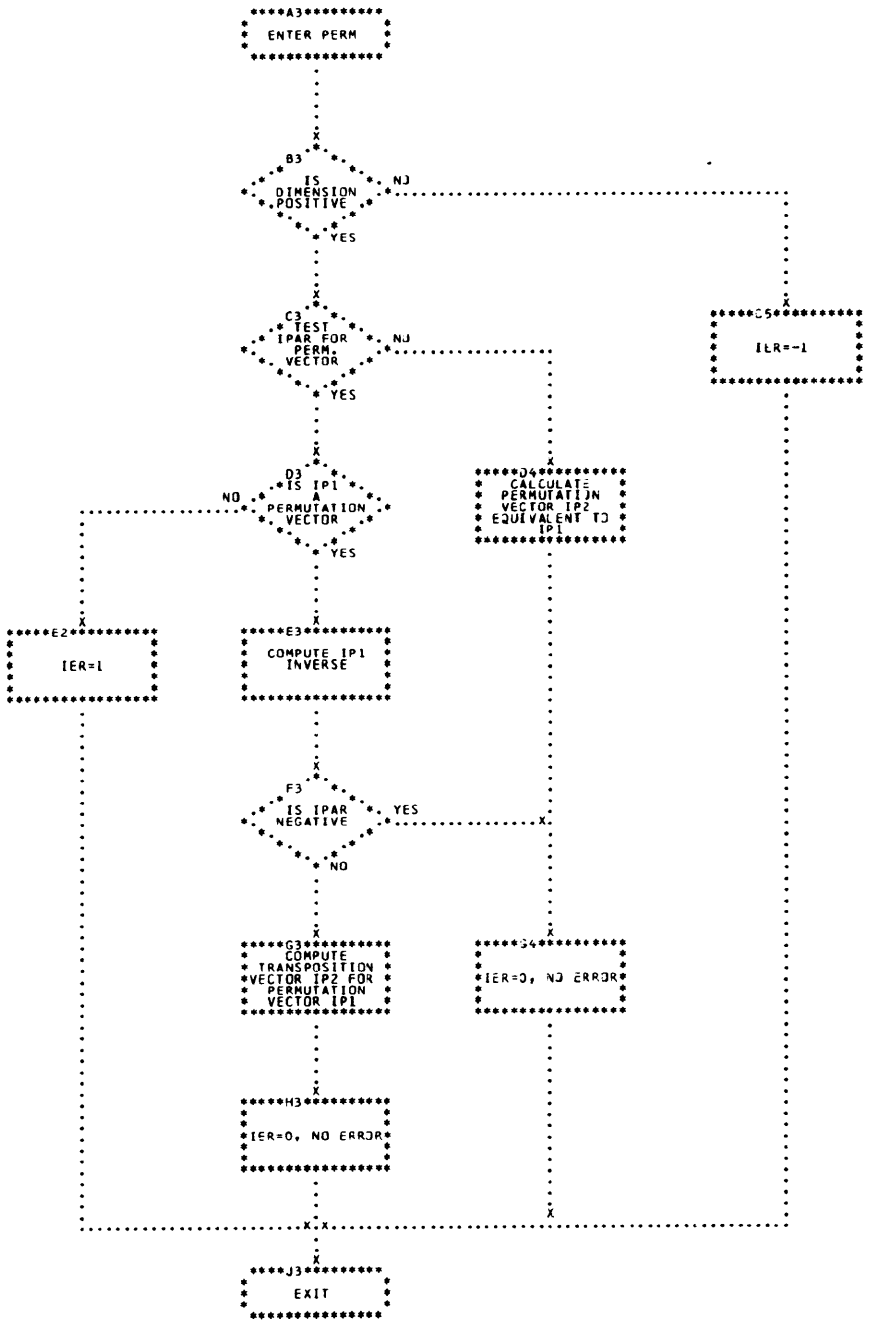

10
SAVE CURRENT FUNCTION VALUE IN FY
A0
ESTIMATE STEPSIZE FOR LINEAR SEARCH
14
B9
PERFORM LINEAR SEARCH
38
A3
CALCULATE DIFFERENCE OF ARGUMENTS AND ITS NORM
IS ONE ITERATION CYCLE COMPLETE
YES
NO
40
IS DIFFERENCE OF ARGUMENTS SMALL
YES
45
B5
NO
41
HAS FUNCTION VALUE DECREASED
NO
19
D7
B5
YES
D3
I
42
45
IS GRADIENT VECTOR SMALL ENOUGH
YES
46
B7
1
NO
B1
3
25
SEARCH MINIMUM ON LINE THROUGH X WITH DIRECTION H
14
B9
SAVE OLD FUNCTION VALUE IN FK AND OLD DY IN DX
C0
STEP ARGUMENT X ALONG LINE WITH ESTIMATED STEPSIZE
C0
COMPUTE FUNCTION VALUE F AND GRADIENT G AT POINT X
SAVE FUNCTION VALUE IN FY
CALCULATE DIRECTIONAL DERIVATIVE DY
IS DY NEGATIVE
NO
YES
38
IS DY ZERO
YES
A3
NO
A4
3
20
17
HAS FUNCTION VALUE DECREASED
NO
20
A4
3
YES
18
INCREASE ESTIMATED STEPSIZE
IS STEPSIZE EXCEEDINGLY LARGE
YES
D7
19
SET ERROR CODE EQUAL 2
A3
3
NO
B9
14

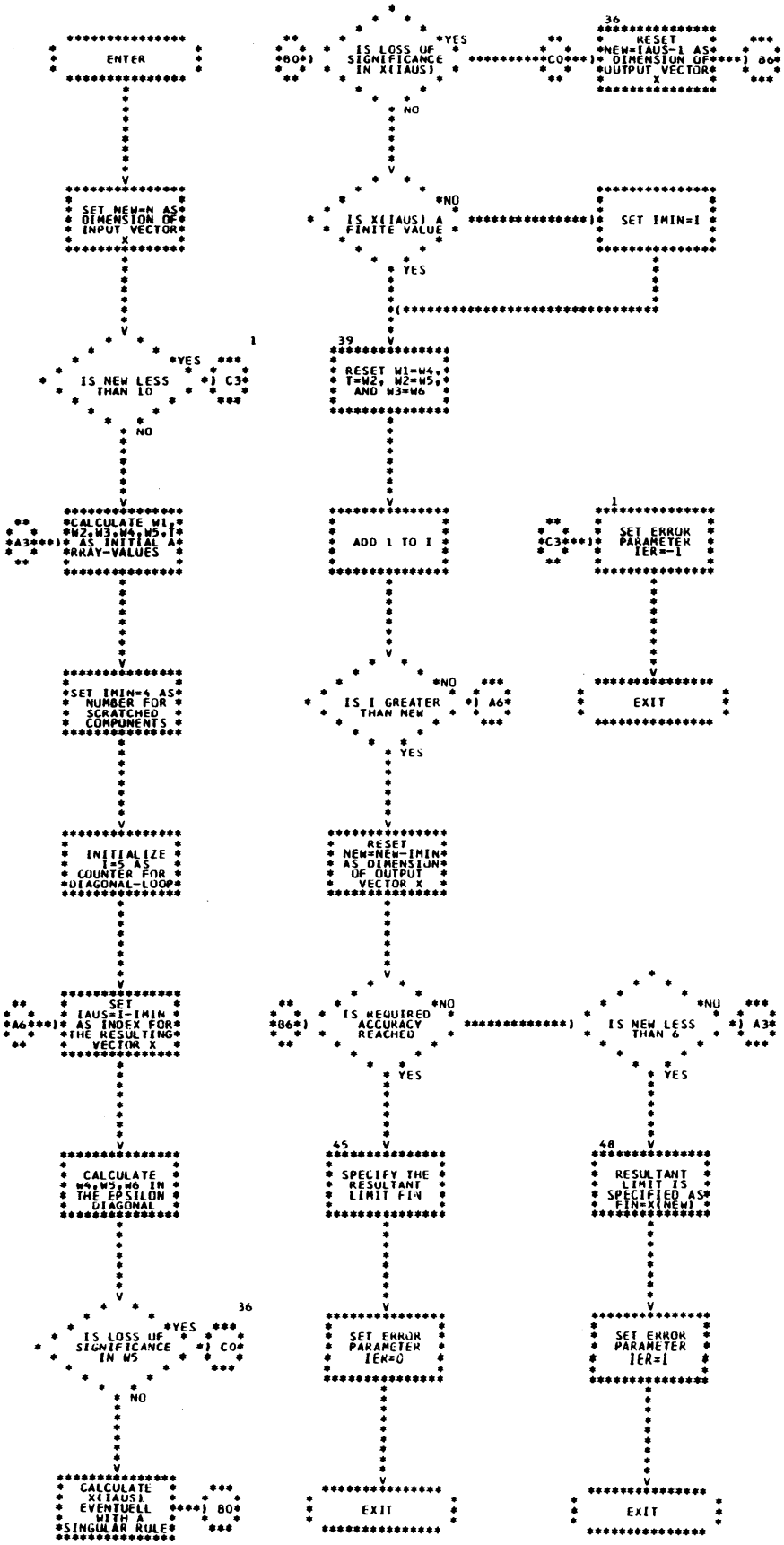

X, F, G ARE RESULTS OF LINEAR SEARCH
23
B0
RESTORE OLD VALUES
25
DID PRECEDING ITERATION FAIL TOO
YES
47
B1
C7
1
NO
26
SET ERROR CODE EQUAL -1
1
A3
1
32
IS F EQUAL FX
YES
NO
33
IS DX EQUAL DALFA
YES
38
A3
2
NO
34
REPLACE OLDER SEARCH POINT BY CURRENT MINIMUM
21
A5
IS F LESS THAN FX
YES
NO
B4
35
IS DALFA NEGATIVE
NO
B4
35
C0
YES

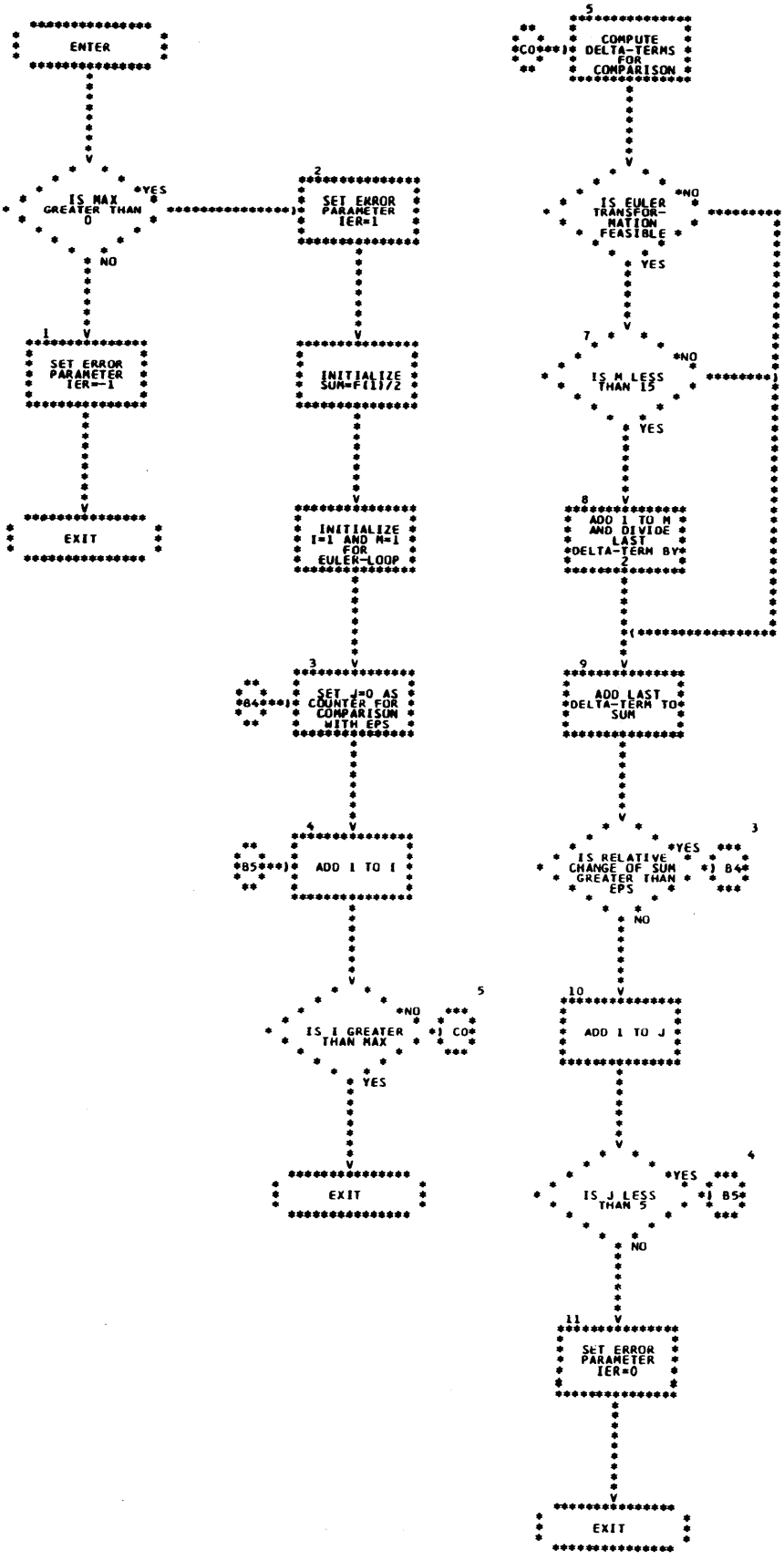

A3
EXIT

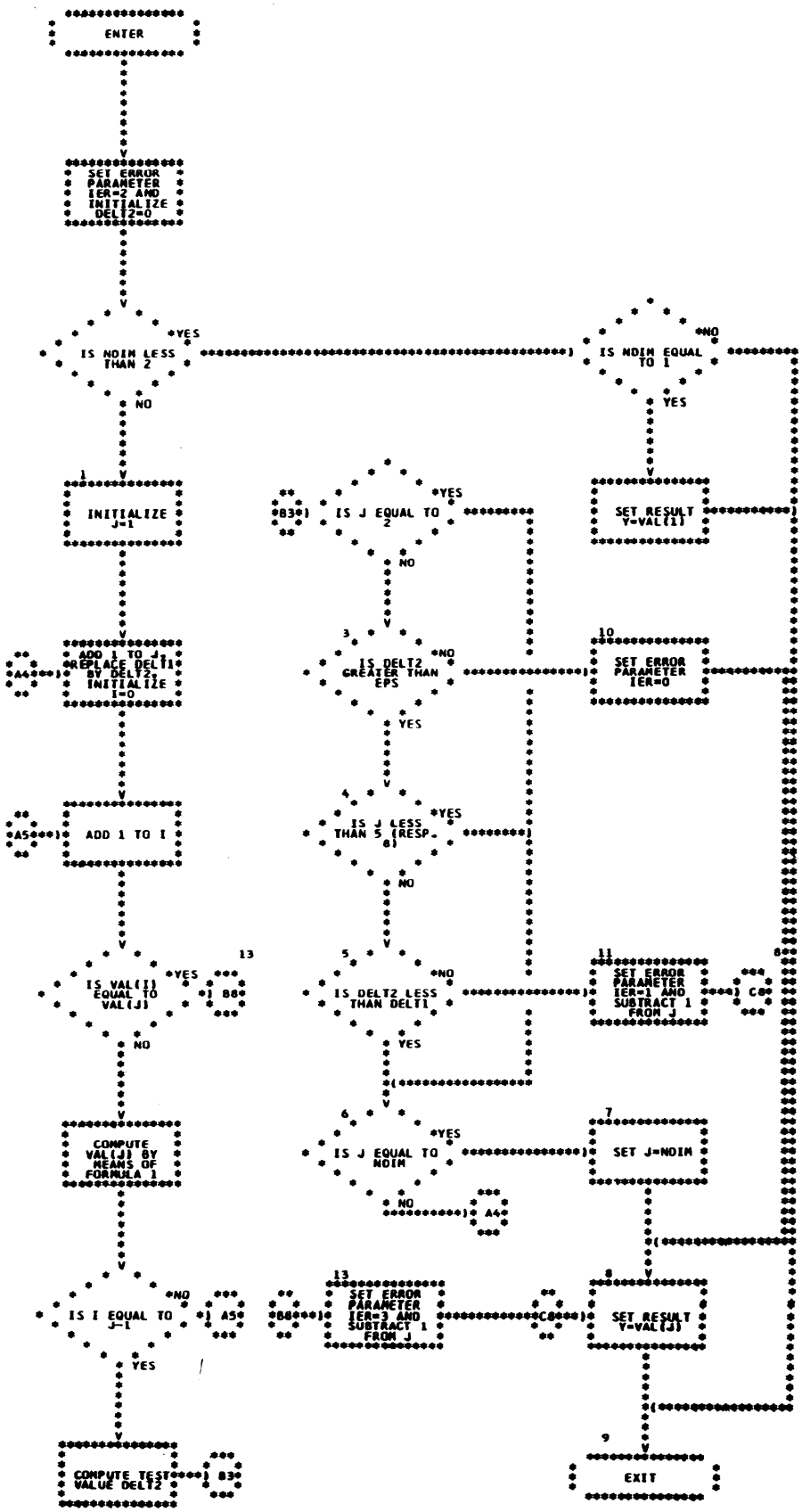

20
A4
INITIALIZE CUBICAL INTERPOLATION
21
A5
SET UP CUBICAL INTERPOLATION POLYNOMIAL
23
IS MINIMIZING ARGUMENT X COMPLEX
YES
B0
NO
27
CALCULATE MINIMIZING ARGUMENT X
COMPUTE FUNCTION VALUE F AND GRADIENT G AT POINT X
DOES F EXCEED FX OR FY
YES
NO
A3
2
38
35
IS F EQUAL FY
YES
B4
NO
30
CALCULATE DIRECTIONAL DERIVATIVE DALFA
C0
36
IS DY EQUAL DALFA
YES
38
A3
2
NO
37
REPLACE LAST SEARCH POINT BY CURRENT MINIMUM
20
A4

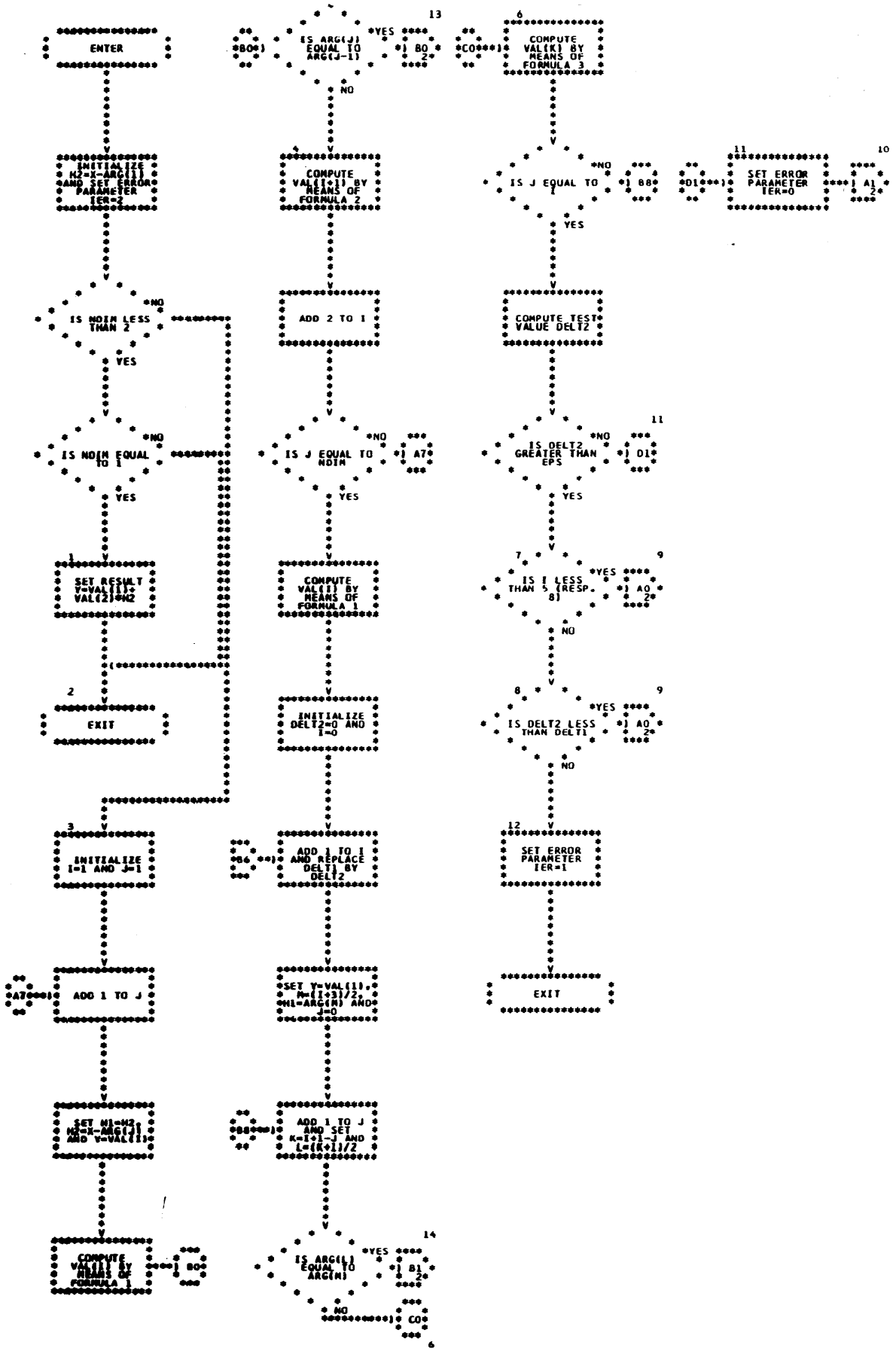

A3
ENTER PPRCN
B3
CALL PERM
C3
IS THERE AN ERROR
YES
NO
D3
IS IPAR NEGATIVE
NO
YES
E3
COMPUTE IP1(IP2(IP1 INVERSE))
E4
COMPUTE IP2(IP1)
F3
EXIT

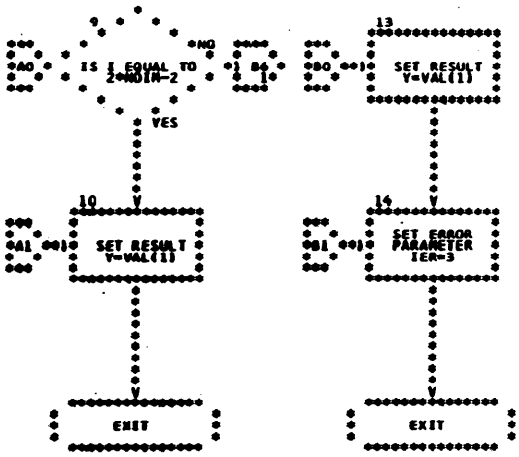

A3
ENTER PERM
B3
IS DIMENSION POSITIVE
NO
YES
C5
IER=-1
C3
TEST IPAR FOR PERM. VECTOR
NO
YES
D3
IS IP1 A PERMUTATION VECTOR
NO
YES
D4
CALCULATE PERMUTATION VECTOR IP2 EQUIVALENT TO IP1
E2
IER=1
E3
COMPUTE IP1 INVERSE
F3
IS IPAR NEGATIVE
YES
NO
G3
COMPUTE TRANSPOSITION VECTOR IP2 FOR PERMUTATION VECTOR IP1
G4
IER=0, NO ERROR
H3
IER=0, NO ERROR
J3
EXIT

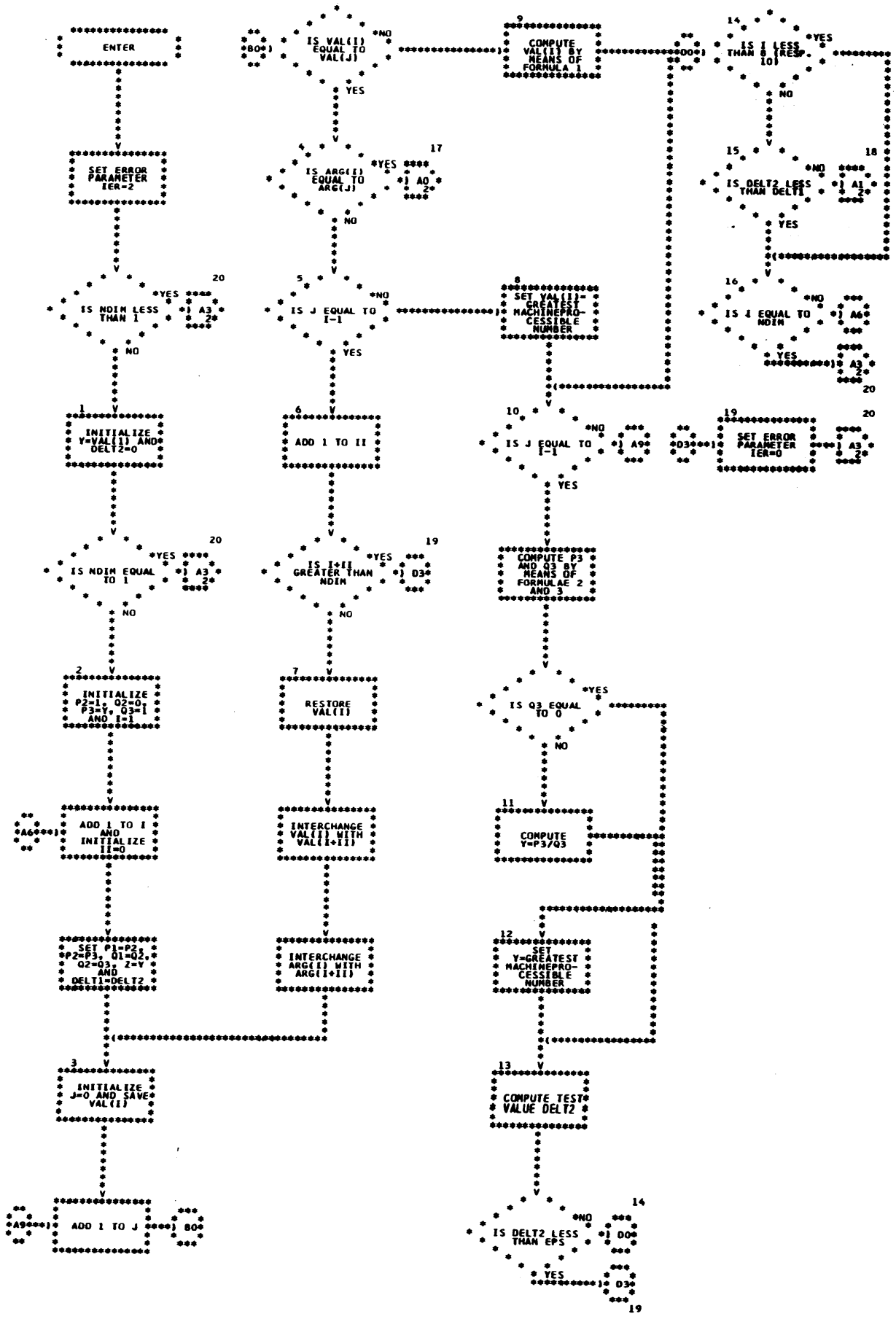

ENTER
SET NEW=N AS DIMENSION OF INPUT VECTOR X
IS NEW LESS THAN 10
YES
NO
1
C3
CALCULATE W1,W2,W3,W4,W5,T AS INITIAL ARRAY-VALUES
A3
SET IMIN=4 AS NUMBER FOR SCRATCHED COMPONENTS
INITIALIZE I=5 AS COUNTER FOR DIAGONAL-LOOP
SET IAUS=I-IMIN AS INDEX FOR THE RESULTING VECTOR X
CALCULATE W4,W5,W6 IN THE EPSILON DIAGONAL
IS LOSS OF SIGNIFICANCE IN W5
36
CO
CALCULATE X(IAUS) EVENTUELL WITH A SINGULAR RULE
B0
IS LOSS OF SIGNIFICANCE IN X(IAUS)
B0
RESET NEW=IAUS-1 AS DIMENSION OF OUTPUT VECTOR X
IS X(IAUS) A FINITE VALUE
SET IMIN=I
39
RESET W1=W4, T=W2, W2=W5, AND W3=W6
ADD 1 TO I
IS I GREATER THAN NEW
A6
RESET NEW=NEW-IMIN AS DIMENSION OF OUTPUT VECTOR X
IS REQUIRED ACCURACY REACHED
B6
45
SPECIFY THE RESULTANT LIMIT FIN
SET ERROR PARAMETER IER=0
EXIT
SET ERROR PARAMETER IER=-1
EXIT
IS NEW LESS THAN 6
A3
48
RESULTANT LIMIT IS SPECIFIED AS FIN=X(NEW)
SET ERROR PARAMETER IER=1
EXIT

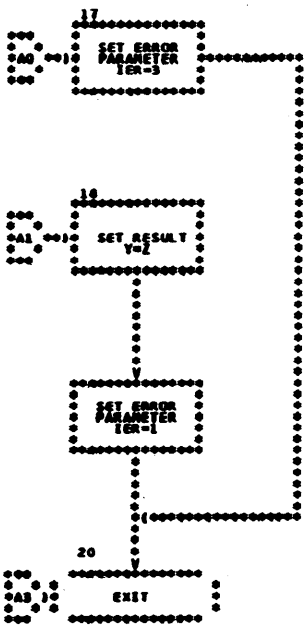

ENTER
IS MAX GREATER THAN 0
YES
NO
SET ERROR PARAMETER IER=-1
EXIT
SET ERROR PARAMETER IER=1
INITIALIZE SUM=F(1)/2
INITIALIZE I=1 AND M=1 FOR EULER-LOOP
SET J=0 AS COUNTER FOR COMPARISON WITH EPS
ADD 1 TO I
IS I GREATER THAN MAX
NO
YES
EXIT
COMPUTE DELTA-TERMS FOR COMPARISON
IS EULER TRANSFOR-MATION FEASIBLE
NO
YES
IS M LESS THAN 15
NO
YES
ADD 1 TO M AND DIVIDE LAST DELTA-TERM BY 2
ADD LAST DELTA-TERM TO SUM
IS RELATIVE CHANGE OF SUM GREATER THAN EPS
YES
NO
ADD 1 TO J
IS J LESS THAN 5
YES
NO
SET ERROR PARAMETER IER=0
EXIT
1
2
3
4
5
7
8
9
10
11
CO
B4
B5

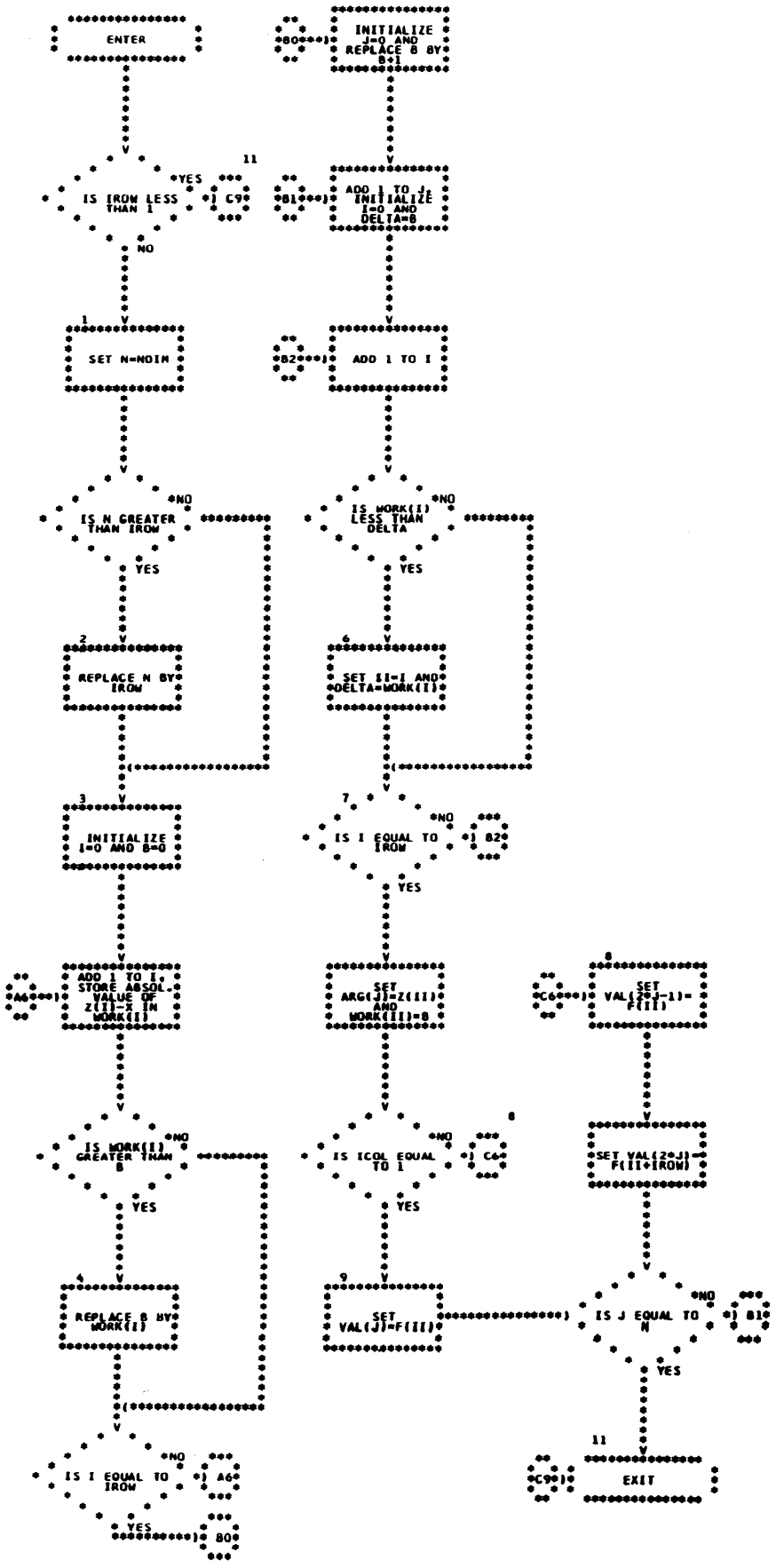

ENTER
SET ERROR PARAMETER IER=2 AND INITIALIZE DELT2=0
IS NDIM LESS THAN 2
YES
NO
IS NDIM EQUAL TO 1
NO
YES
1
INITIALIZE J=1
SET RESULT Y=VAL(1)
B3
IS J EQUAL TO 2
YES
NO
3
IS DELT2 GREATER THAN EPS
NO
YES
10
SET ERROR PARAMETER IER=0
A4
ADD 1 TO J, REPLACE DELT1 BY DELT2, INITIALIZE I=0
4
IS J LESS THAN 5 (RESP. 8)
YES
NO
A5
ADD 1 TO I
5
IS DELT2 LESS THAN DELT1
NO
YES
11
SET ERROR PARAMETER IER=1 AND SUBTRACT 1 FROM J
C
13
IS VAL(I) EQUAL TO VAL(J)
YES
NO
B8
6
IS J EQUAL TO NDIM
YES
NO
7
SET J=NDIM
A4
COMPUTE VAL(J) BY MEANS OF FORMULA 1
13
SET ERROR PARAMETER IER=3 AND SUBTRACT 1 FROM J
C
8
SET RESULT Y=VAL(J)
IS I EQUAL TO J-1
NO
YES
A5
B8
COMPUTE TEST VALUE DELT2
B3
9
EXIT

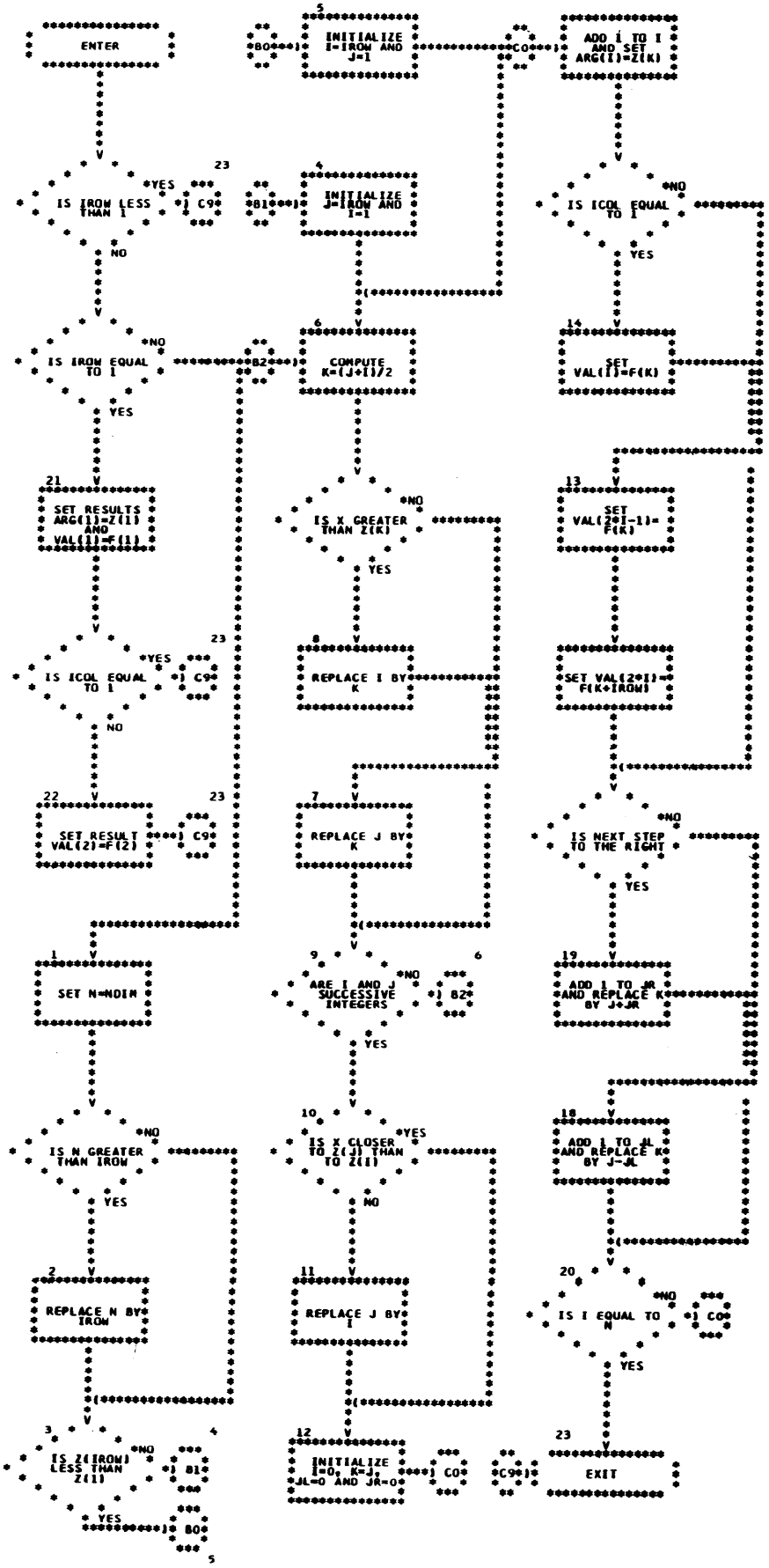

ENTER
INITIALIZE K2=X-ARG(1) AND SET ERROR PARAMETER IER=2
IS NDIM LESS THAN 2
*NO
YES
IS NDIM EQUAL TO 1
*NO
YES
1
SET RESULT Y=VAL(1)*VAL(2)*H2
2
EXIT
3
INITIALIZE I=1 AND J=1
ADD 1 TO J
SET H1=H2, H2=X-ARG(J+1) AND Y=VAL(I)*
COMPUTE VAL(I) BY MEANS OF FORMULA 1
*B0
13
IS ARG(J) EQUAL TO ARG(J-1)
*YES
*NO
*B0
4
COMPUTE VAL(I+1) BY MEANS OF FORMULA 2
ADD 2 TO I
IS J EQUAL TO NDIM
*NO
*A7
YES
INITIALIZE DELT2=0 AND I=0
COMPUTE VAL(I) BY MEANS OF FORMULA 1
ADD 1 TO I AND REPLACE DELT1 BY DELT2
*A6
SET Y=VAL(I)*VAL(J)*H1=ARG(N) AND J=0
ADD 1 TO J AND SET K=I+1-J AND L=(K+1)/2
14
IS ARG(L) EQUAL TO ARG(N)
*YES
*NO
*B1
*CO
6
COMPUTE VAL(K) BY MEANS OF FORMULA 3
*CO
IS J EQUAL TO I
*NO
*B8
YES
COMPUTE TEST VALUE DELT2
11
IS DELT2 GREATER THAN EPS
*NO
*D1
YES
7
IS I LESS THAN 5 (RESP. 8)
*YES
*A0
NO
8
IS DELT2 LESS THAN DELT1
*YES
*A0
NO
12
SET ERROR PARAMETER IER=1
EXIT
11
SET ERROR PARAMETER IER=0
10
*A1
*D1

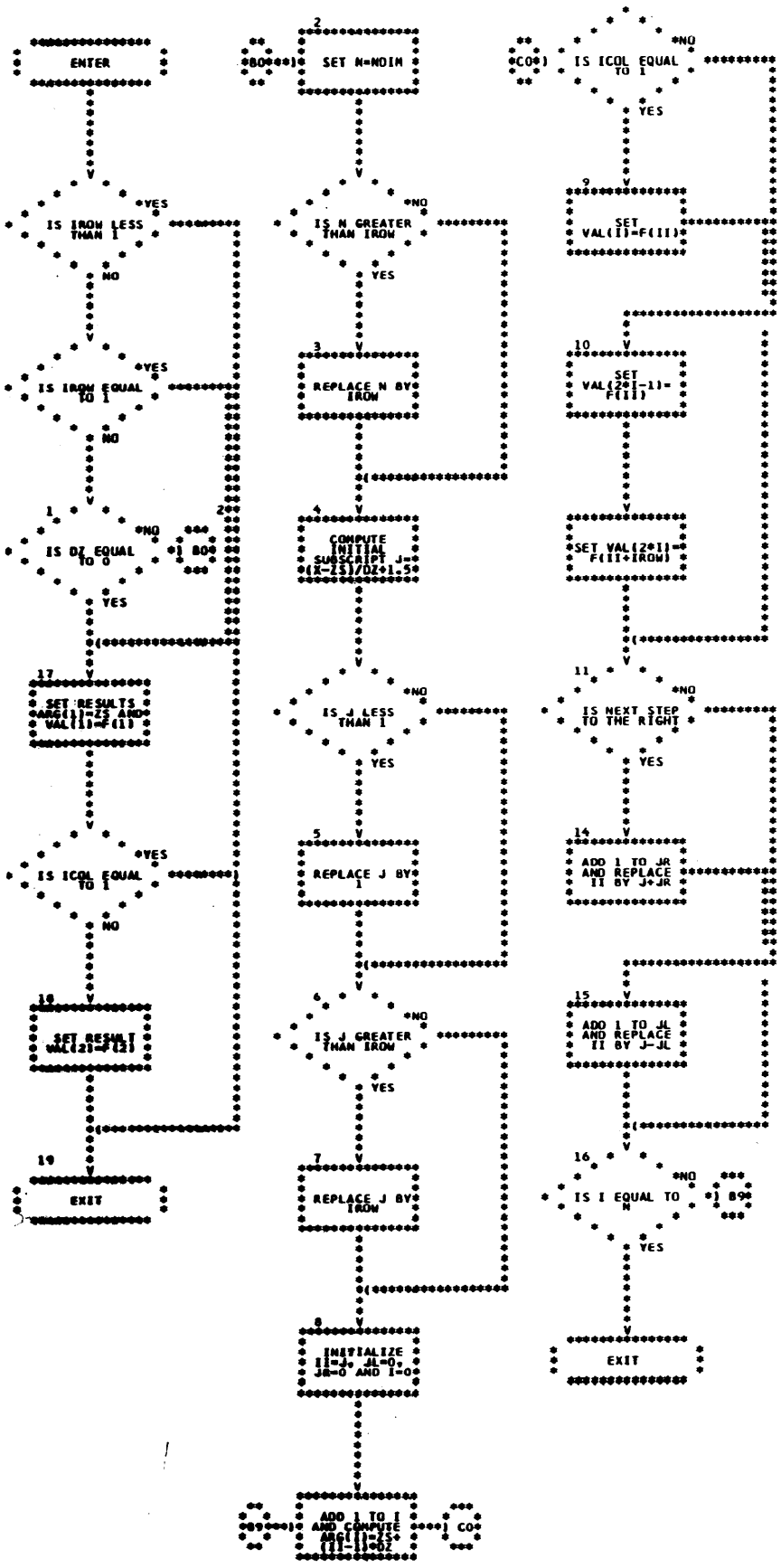

IS I EQUAL TO 2+NDIM-2
NO
YES
10
SET RESULT V=VAL(I)
EXIT
13
SET RESULT V=VAL(I)
14
SET ERROR PARAMETER IER=3
EXIT

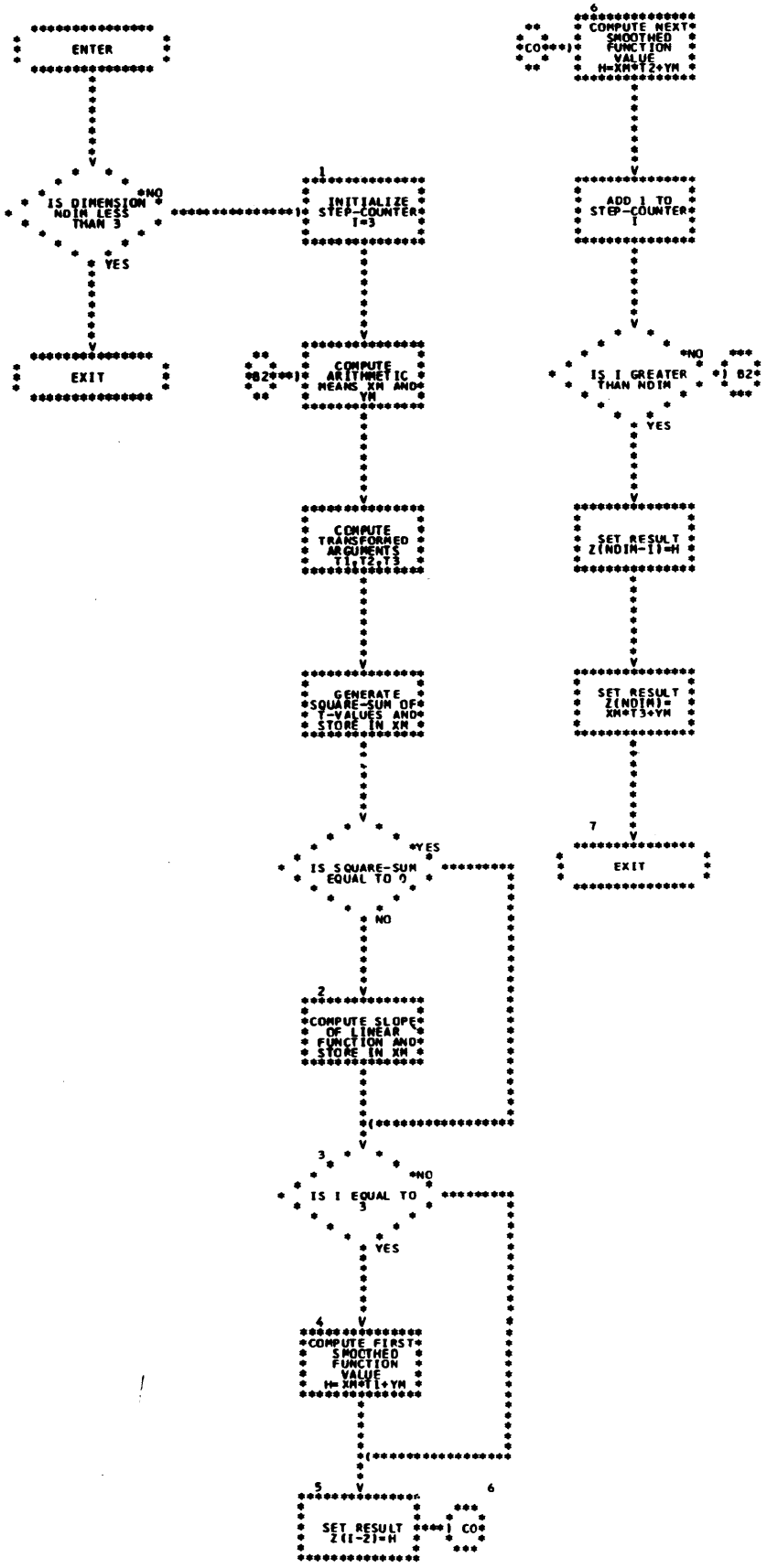

ENTER
SET ERROR PARAMETER IER=2
IS NDIM LESS THAN 1
YES
NO
A3
2
INITIALIZE Y=VAL(1) AND DELT2=0
IS NDIM EQUAL TO 1
INITIALIZE P2=1, Q2=0, P3=Y, Q3=1 AND I=1
ADD 1 TO I AND INITIALIZE II=0
SET P1=P2, P2=P3, Q1=Q2, Q2=Q3, Z=Y AND DELT1=DELT2
INITIALIZE J=0 AND SAVE VAL(I)
ADD 1 TO J
B0
IS VAL(I) EQUAL TO VAL(J)
IS ARG(I) EQUAL ARG(J)
A0
IS J EQUAL TO I-1
ADD 1 TO II
IS I+II GREATER THAN NDIM
D3
RESTORE VAL(I)
INTERCHANGE VAL(I) WITH VAL(I+II)
INTERCHANGE ARG(I) WITH ARG(I+II)
COMPUTE VAL(I) BY MEANS OF FORMULA 1
SET VAL(I)= GREATEST MACHINEPRO- CESSIBLE NUMBER
IS J EQUAL TO I-1
A9
COMPUTE P3 AND Q3 BY MEANS OF FORMULAE 2 AND 3
IS Q3 EQUAL TO 0
COMPUTE Y=P3/Q3
SET Y=GREATEST MACHINEPRO- CESSIBLE NUMBER
COMPUTE TEST VALUE DELT2
IS DELT2 LESS THAN EPS
D0
IS I LESS THAN 0 (RESP. 10)
IS DELT2 LESS THAN DELTI
A1
IS J EQUAL TO NDIM
A6
SET ERROR PARAMETER IER=0

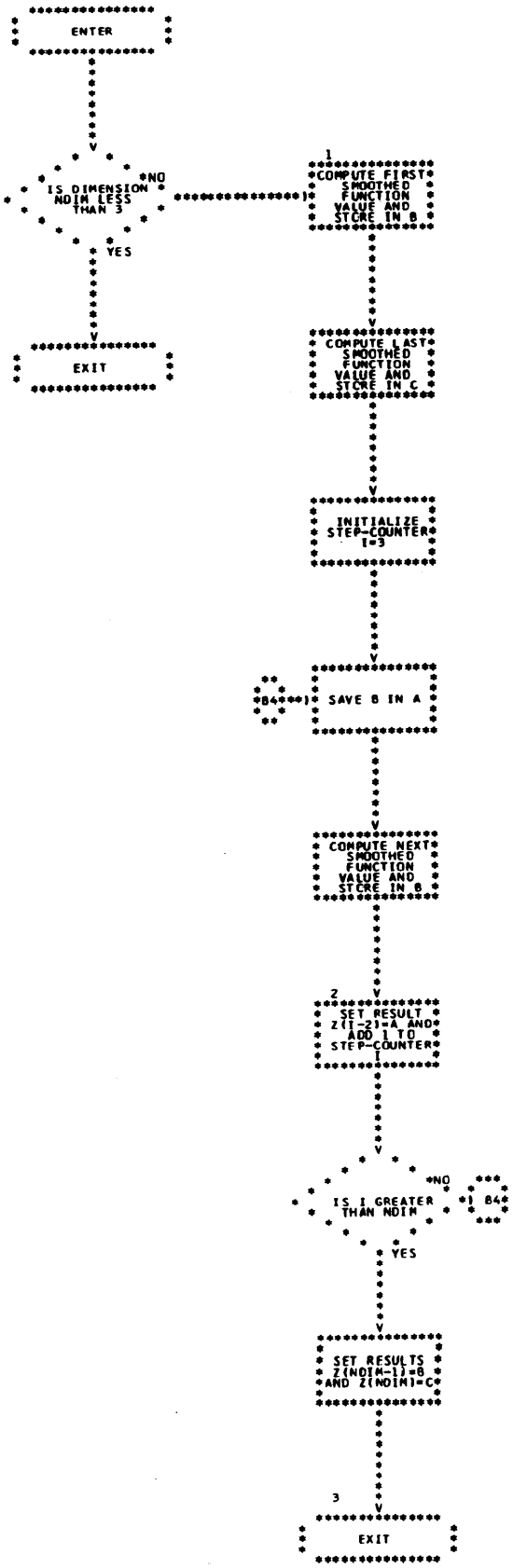

17
A0
SET ERROR PARAMETER IER=3
18
A1
SET RESULT V=Z
SET ERROR PARAMETER IER=1
20
A3
EXIT

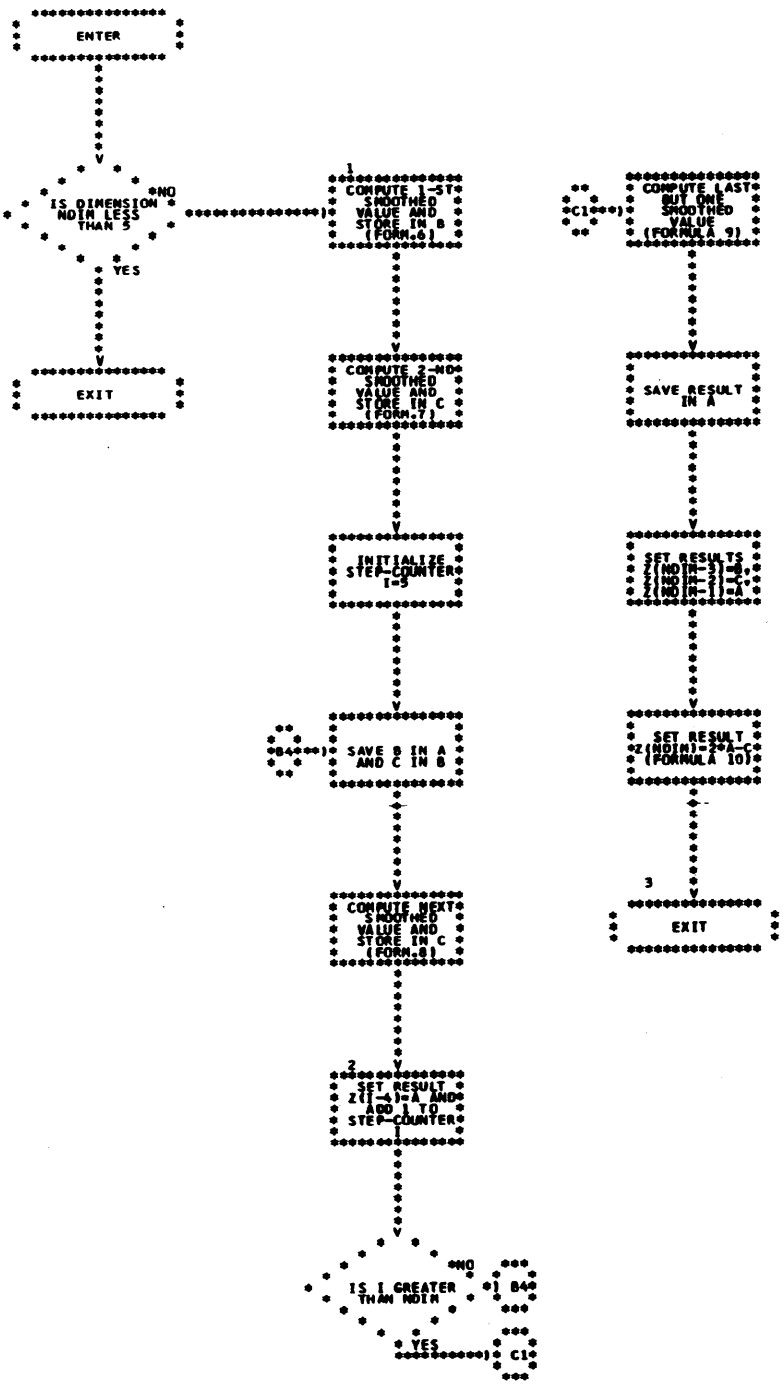

ENTER
IS IROW LESS THAN 1
YES
C9
NO
SET N=NDIM
IS N GREATER THAN IROW
NO
YES
REPLACE N BY IROW
INITIALIZE I=0 AND B=0
ADD 1 TO I, STORE ABSOL. VALUE OF Z(I)-X IN WORK(I)
A6
IS WORK(I) GREATER THAN B
NO
YES
REPLACE B BY WORK(I)
IS I EQUAL TO IROW
NO
YES
A6
80
INITIALIZE J=0 AND REPLACE B BY B+1
80
ADD 1 TO J, INITIALIZE I=0 AND DELTA=B
B1
ADD 1 TO I
B2
IS WORK(I) LESS THAN DELTA
NO
YES
SET II=I AND DELTA=WORK(I)
IS I EQUAL TO IROW
NO
B2
YES
SET ARG(J)=Z(III) AND WORK(III)=B
IS ICOL EQUAL TO 1
NO
C4
YES
SET VAL(J)=F(III)
SET VAL(2*J-1)=F(II)
C6
SET VAL(2*J)=F(III+IROW)
IS J EQUAL TO N
NO
B1
YES
EXIT
C9

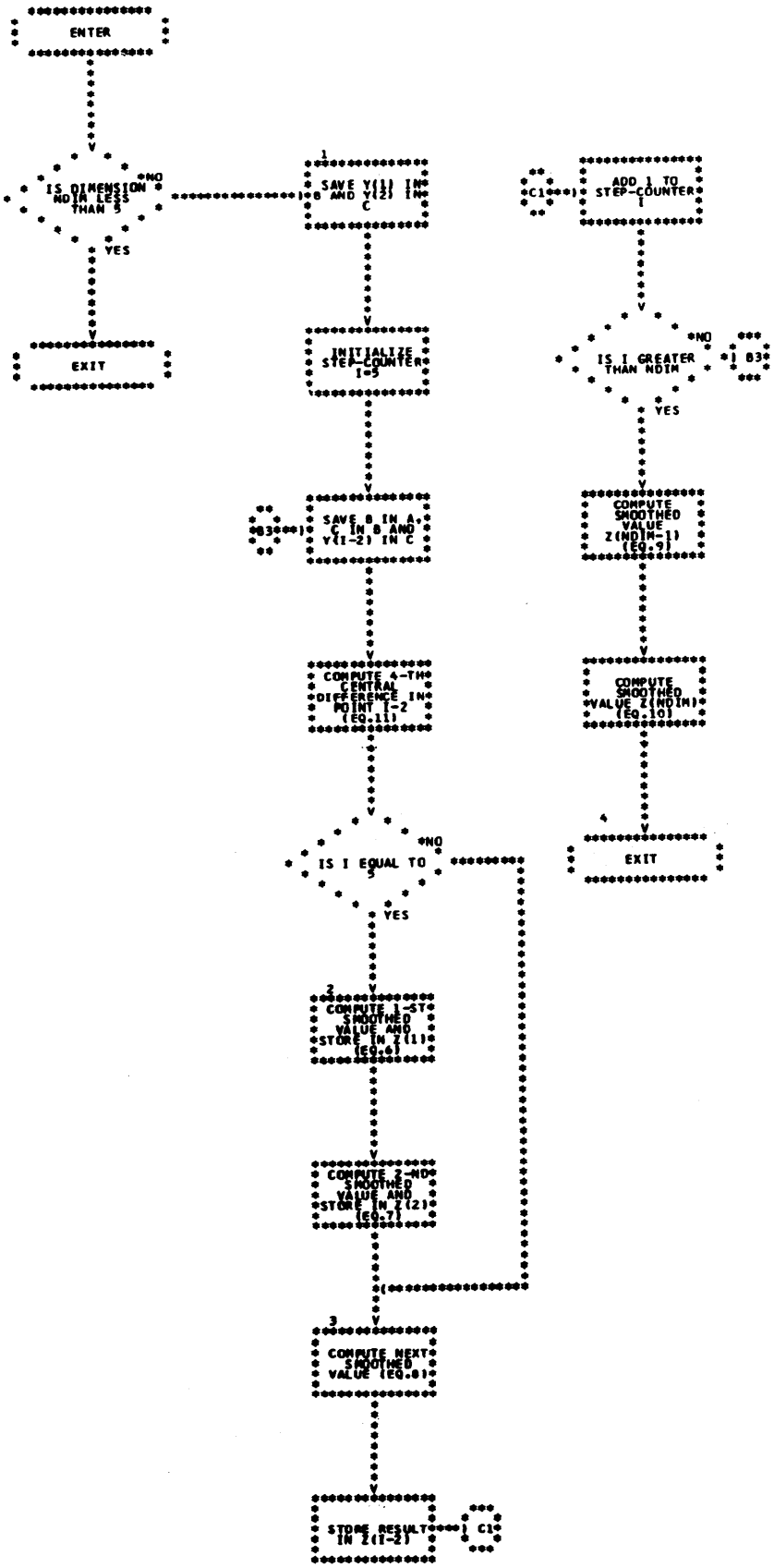

ENTER
INITIALIZE I=IROW AND J=1
ADD 1 TO I AND SET ARG(I)=Z(K)
IS IROW LESS THAN 1
YES
NO
INITIALIZE J=IROW AND I=1
IS ICOL EQUAL TO I
IS IROW EQUAL TO I
COMPUTE K=(J+I)/2
SET VAL(I)=F(K)
SET RESULTS ARG(1)=Z(1) AND VAL(1)=F(1)
IS X GREATER THAN Z(K)
SET VAL(2*I-1)= F(K)
IS ICOL EQUAL TO 1
REPLACE I BY K
SET VAL(2*I)= F(K+IROW)
SET RESULT VAL(2)=F(2)
REPLACE J BY K
IS NEXT STEP TO THE RIGHT
SET N=NDIM
ARE I AND J SUCCESSIVE INTEGERS
ADD 1 TO JR AND REPLACE K BY J+JR
IS N GREATER THAN IROW
IS X CLOSER TO Z(J) THAN TO Z(I)
ADD 1 TO JL AND REPLACE K BY J-JL
REPLACE N BY IROW
REPLACE J BY I
IS I EQUAL TO N
IS Z(IROW) LESS THAN Z(1)
INITIALIZE I=0, K=J, JL=0 AND JR=0
EXIT

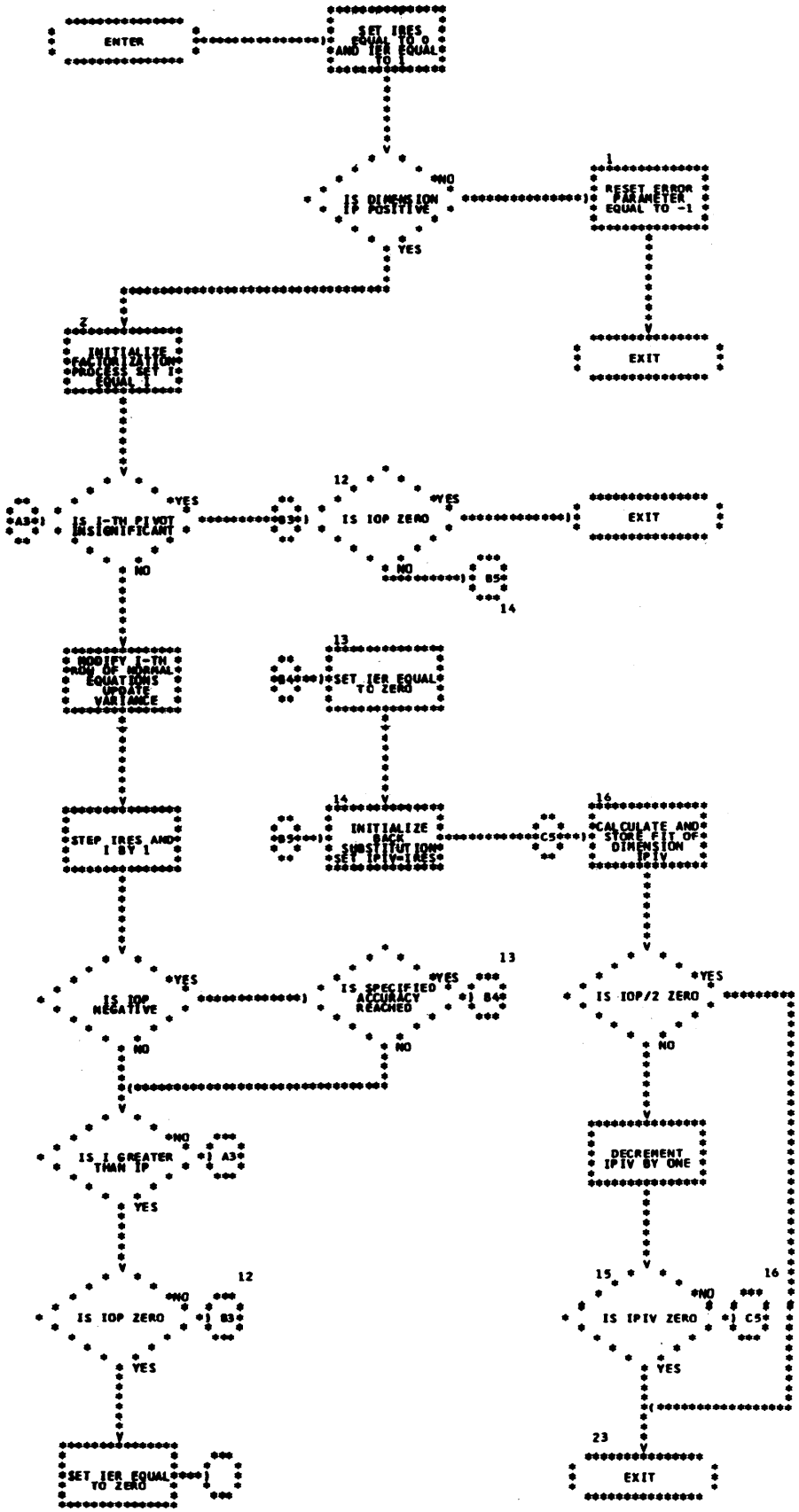

ENTER
IS IROW LESS THAN 1
YES
NO
IS IROW EQUAL TO 1
YES
NO
IS DZ EQUAL TO 0
YES
NO
SET RESULTS ARG(1)=ZS AND VAL(1)=F(1)
IS ICOL EQUAL TO 1
YES
NO
SET RESULT VAL(2)=F(2)
EXIT
SET N=NDIM
IS N GREATER THAN IROW
YES
NO
REPLACE N BY IROW
COMPUTE INITIAL SUBSCRIPT J=(X-ZS)/DZ+1.5
IS J LESS THAN 1
YES
NO
REPLACE J BY 1
IS J GREATER THAN IROW
YES
NO
REPLACE J BY IROW
INITIALIZE II=J, JL=0, JR=0 AND I=0
ADD 1 TO I AND COMPUTE ARG(I)=ZS+(II-1)*DZ
IS ICOL EQUAL TO 1
YES
NO
SET VAL(I)=F(II)
SET VAL(2*I-1)=F(II)
SET VAL(2*I)=F(II+IROW)
IS NEXT STEP TO THE RIGHT
YES
NO
ADD 1 TO JR AND REPLACE II BY J+JR
ADD 1 TO JL AND REPLACE II BY J-JL
IS I EQUAL TO N
YES
NO
EXIT

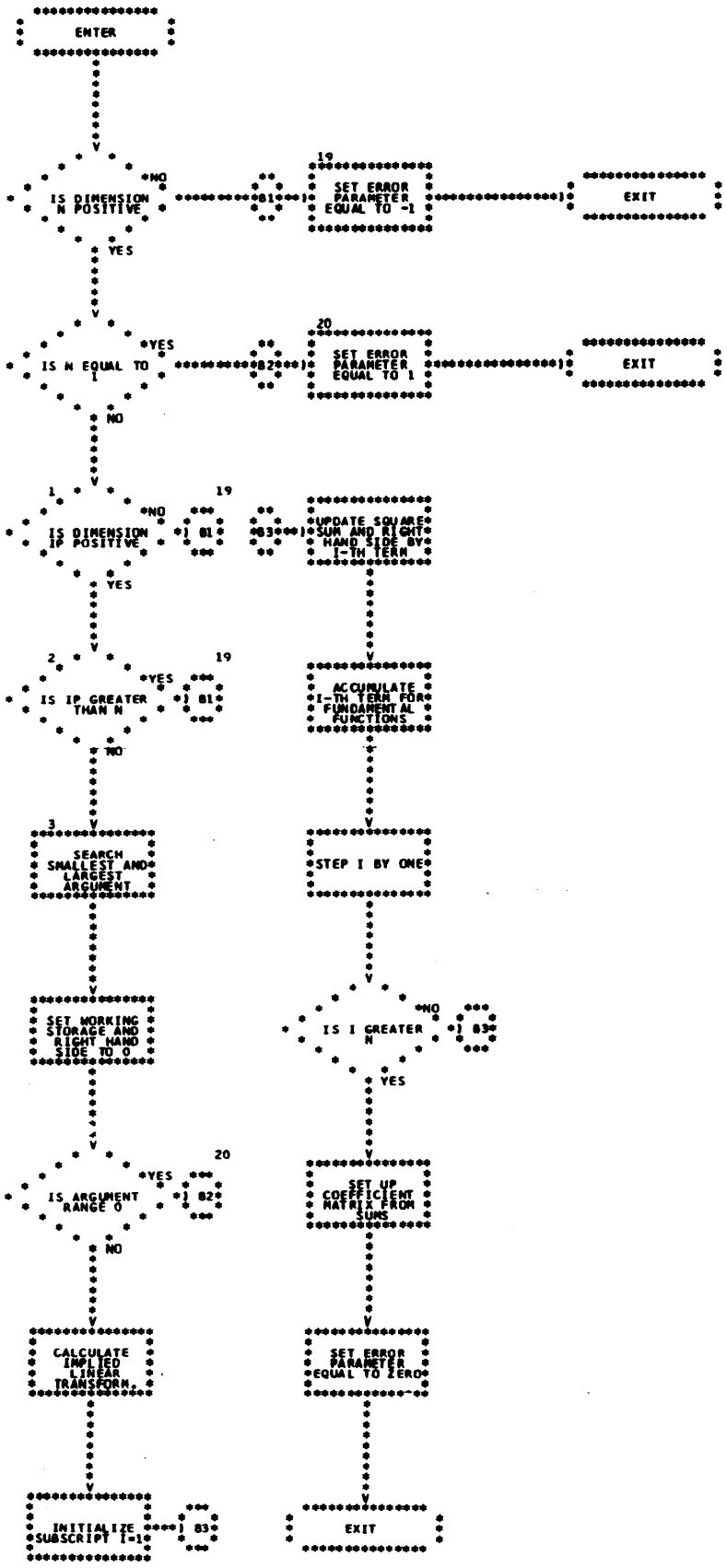

ENTER
IS DIMENSION NDIM LESS THAN 3
NO
YES
EXIT
1
INITIALIZE STEP-COUNTER I=3
B2
COMPUTE ARITHMETIC MEANS XM AND YM
COMPUTE TRANSFORMED ARGUMENTS T1,T2,T3
GENERATE SQUARE-SUM OF T-VALUES AND STORE IN XM
IS SQUARE-SUM EQUAL TO 0
YES
NO
2
COMPUTE SLOPE OF LINEAR FUNCTION AND STORE IN XM
3
IS I EQUAL TO 3
NO
YES
4
COMPUTE FIRST SMOOTHED FUNCTION VALUE H=XM*T1+YM
5
SET RESULT Z(I-2)=H
6
CO
6
COMPUTE NEXT SMOOTHED FUNCTION VALUE H=XM*T2+YM
CO
ADD 1 TO STEP-COUNTER I
IS I GREATER THAN NDIM
NO
B2
YES
SET RESULT Z(NDIM-1)=H
SET RESULT Z(NDIM)=XM*T3+YM
7
EXIT

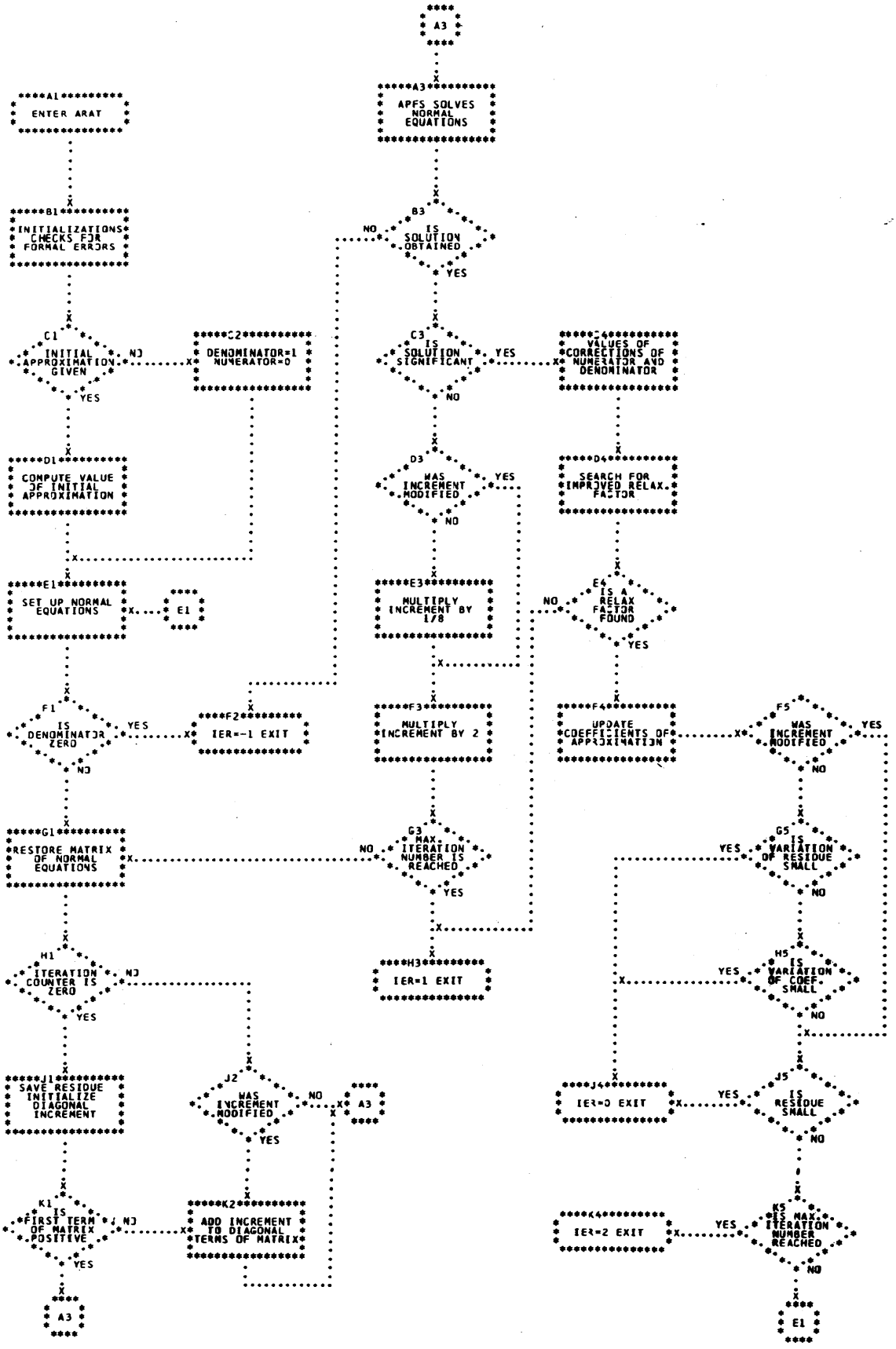

ENTER
IS DIMENSION NDIM LESS THAN 3
NO
YES
EXIT
1
COMPUTE FIRST SMOOTHED FUNCTION VALUE AND STORE IN B
COMPUTE LAST SMOOTHED FUNCTION VALUE AND STORE IN C
INITIALIZE STEP-COUNTER I=3
B4
SAVE B IN A
COMPUTE NEXT SMOOTHED FUNCTION VALUE AND STORE IN B
2
SET RESULT Z(I-2)=A AND ADD 1 TO STEP-COUNTER I
IS I GREATER THAN NDIM
NO
I B4
YES
SET RESULTS Z(NDIM-1)=B AND Z(NDIM)=C
3
EXIT

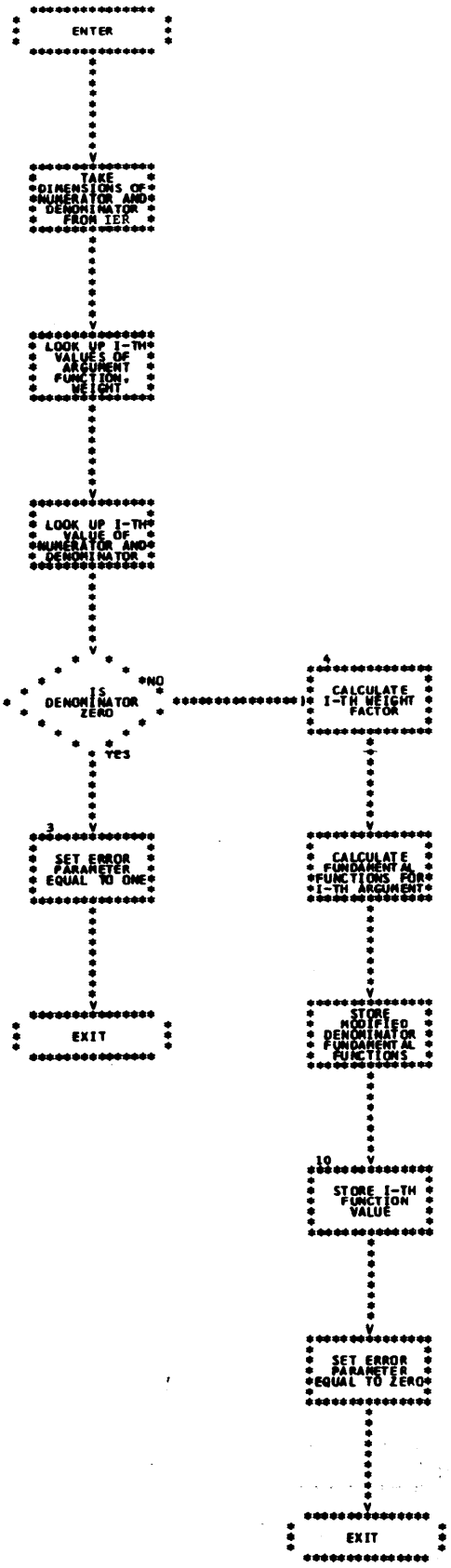

ENTER
IS DIMENSION NDIM LESS THAN 5
NO
YES
EXIT
1
COMPUTE 1-ST SMOOTHED VALUE AND STORE IN B (FORM.6)
COMPUTE 2-ND SMOOTHED VALUE AND STORE IN C (FORM.7)
INITIALIZE STEP-COUNTER I=5
B4
SAVE B IN A AND C IN B
COMPUTE NEXT SMOOTHED VALUE AND STORE IN C (FORM.8)
2
SET RESULT Z(I-4)=A AND ADD 1 TO STEP-COUNTER I
IS I GREATER THAN NDIM
NO
B4
YES
C1
C1
COMPUTE LAST BUT ONE SMOOTHED VALUE (FORMULA 9)
SAVE RESULT IN A
SET RESULTS Z(NDIM-3)=B, Z(NDIM-2)=C, Z(NDIM-1)=A
SET RESULT Z(NDIM)=2*A-C (FORMULA 10)
3
EXIT

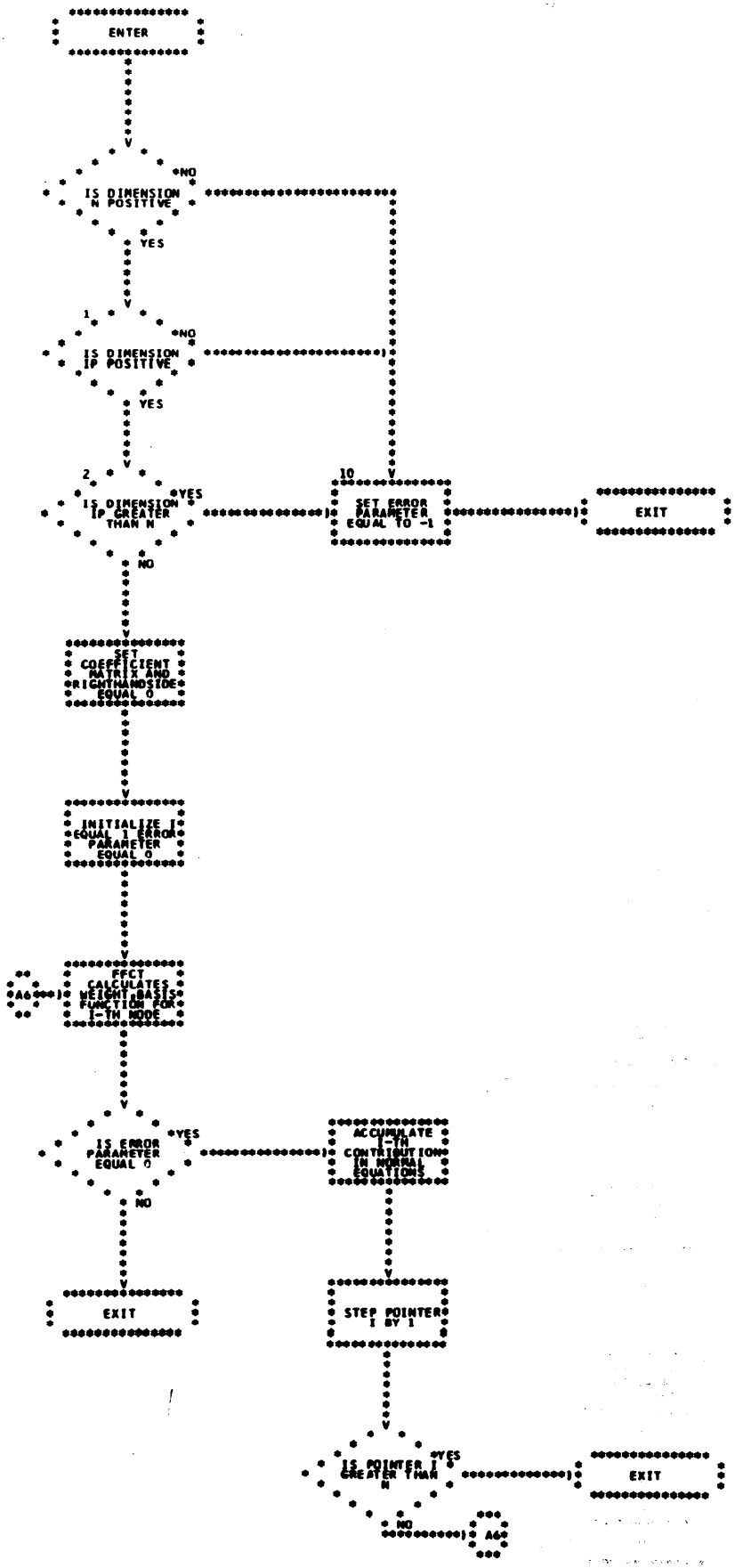

ENTER
IS DIMENSION NDIM LESS THAN 5
NO
YES
EXIT
1
SAVE Y(1) IN B AND Y(2) IN C
INITIALIZE STEP-COUNTER I=5
B3
SAVE B IN A, C IN B AND Y(I-2) IN C
COMPUTE 4-TH CENTRAL DIFFERENCE IN POINT I-2 (EQ.11)
IS I EQUAL TO 5
NO
YES
2
COMPUTE 1-ST SMOOTHED VALUE AND STORE IN Z(1) (EQ.6)
COMPUTE 2-ND SMOOTHED VALUE AND STORE IN Z(2) (EQ.7)
3
COMPUTE NEXT SMOOTHED VALUE (EQ.8)
STORE RESULT IN Z(I-2)
C1
ADD 1 TO STEP-COUNTER I
IS I GREATER THAN NDIM
NO
B3
YES
COMPUTE SMOOTHED VALUE Z(NDIM-1) (EQ.9)
COMPUTE SMOOTHED VALUE Z(NDIM) (EQ.10)
4
EXIT

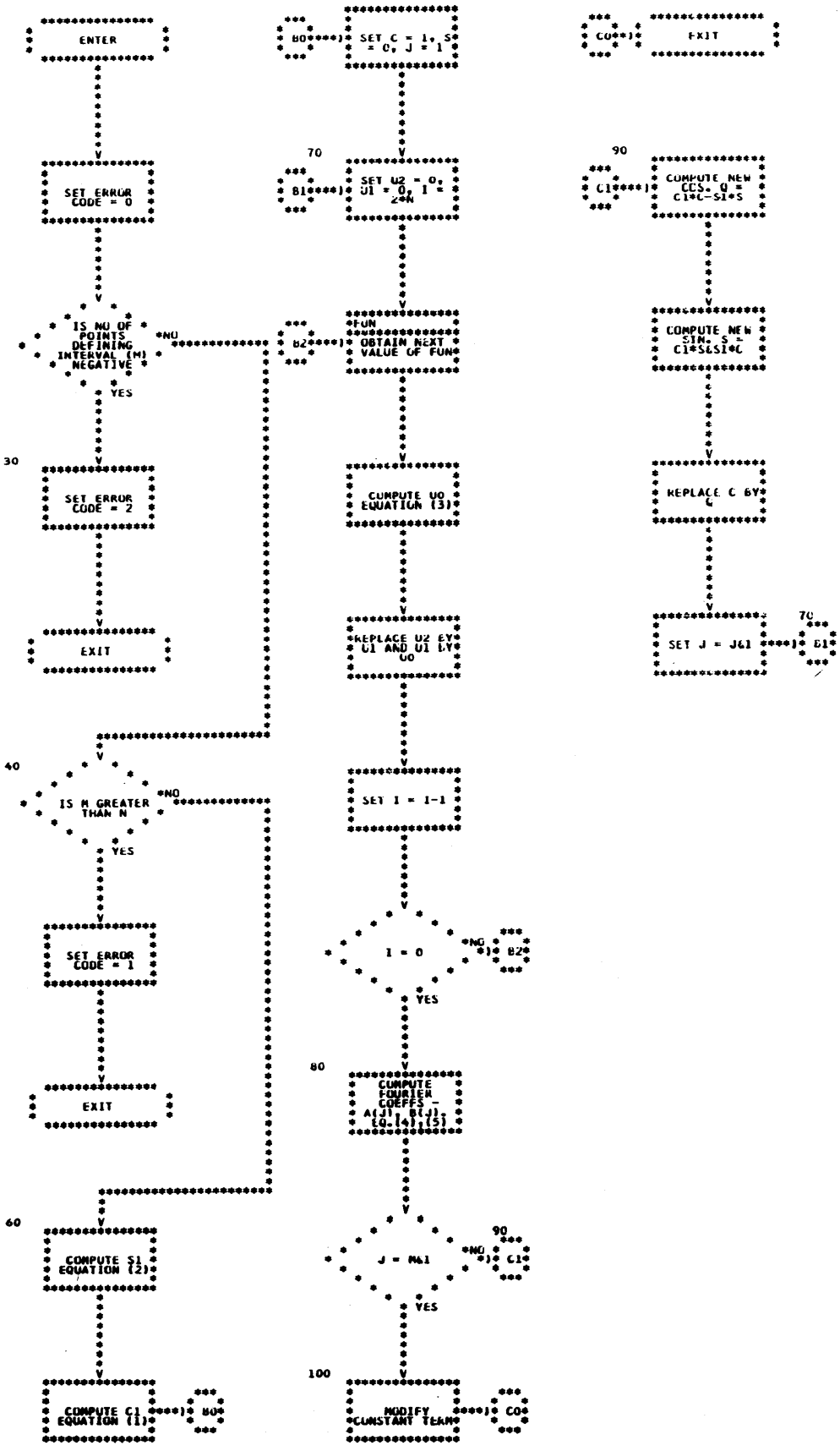

ENTER
SET IRES EQUAL TO 0 AND IER EQUAL TO 1
IS DIMENSION IP POSITIVE
NO
YES
1
RESET ERROR PARAMETER EQUAL TO -1
EXIT
2
INITIALIZE FACTORIZATION PROCESS SET I EQUAL 1
IS I-TH PIVOT INSIGNIFICANT
YES
A3
NO
12
IS IOP ZERO
YES
B3
NO
B5
14
EXIT
MODIFY I-TH ROW OF NORMAL EQUATIONS UPDATE VARIANCE
13
B4
SET IER EQUAL TO ZERO
STEP IRES AND I BY 1
14
INITIALIZE BACK SUBSTITUTION SET IPIV=IRES
B5
C5
16
CALCULATE AND STORE FIT OF DIMENSION IPIV
IS IOP NEGATIVE
YES
13
IS SPECIFIED ACCURACY REACHED
YES
B4
NO
NO
IS IOP/2 ZERO
YES
NO
IS I GREATER THAN IP
NO
A3
YES
DECREMENT IPIV BY ONE
12
IS IOP ZERO
NO
B3
YES
15
IS IPIV ZERO
NO
16
C5
YES
SET IER EQUAL TO ZERO
23
EXIT

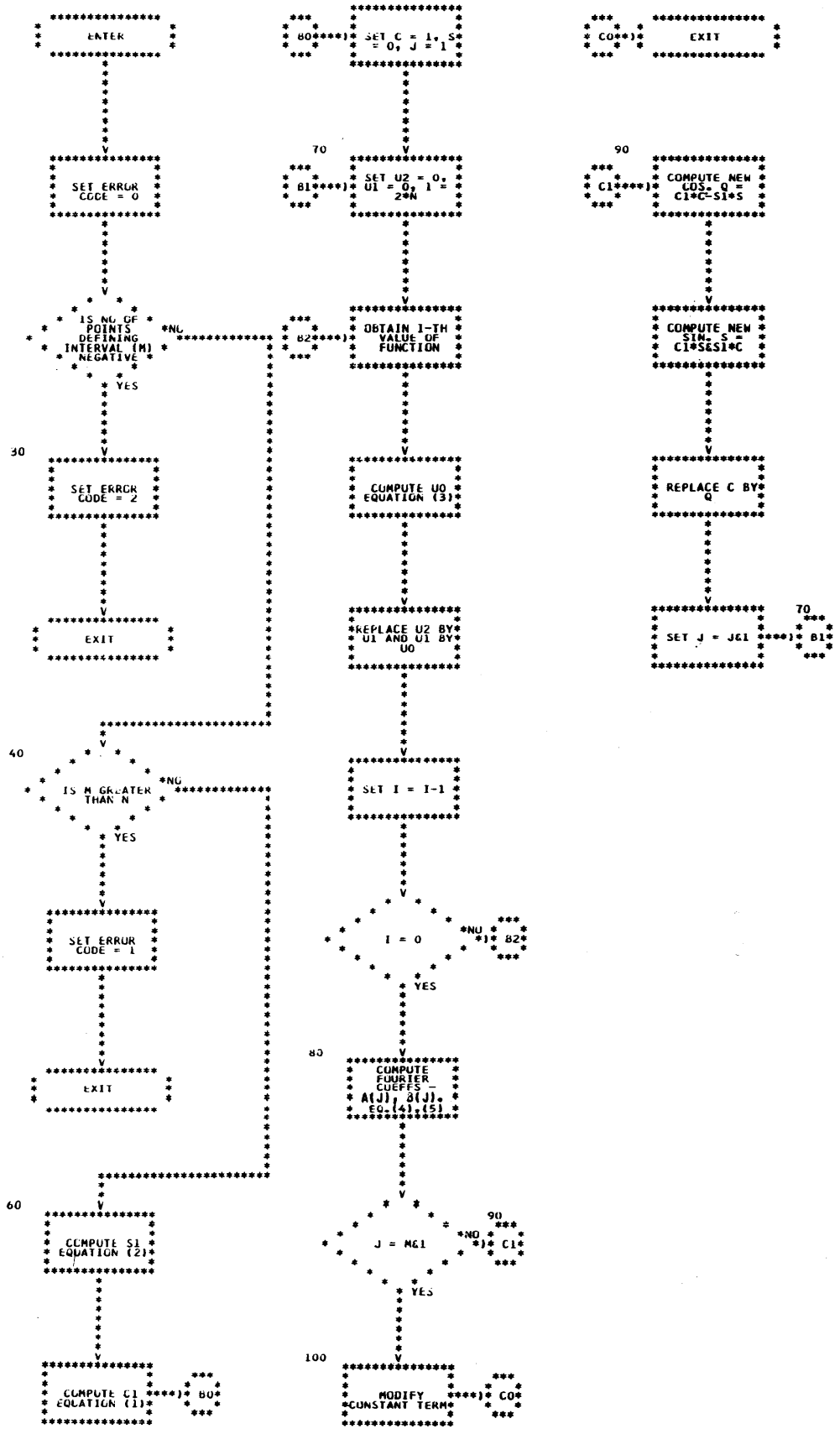

ENTER
IS DIMENSION N POSITIVE
NO
B1
19
SET ERROR PARAMETER EQUAL TO -1
EXIT
YES
IS N EQUAL TO 1
YES
B2
20
SET ERROR PARAMETER EQUAL TO 1
EXIT
NO
1
IS DIMENSION IP POSITIVE
NO
B1
19
B3
UPDATE SQUARE SUM AND RIGHT HAND SIDE BY I-TH TERM
YES
2
IS IP GREATER THAN N
YES
B1
19
ACCUMULATE I-TH TERM FOR FUNDAMENTAL FUNCTIONS
NO
3
SEARCH SMALLEST AND LARGEST ARGUMENT
STEP I BY ONE
SET WORKING STORAGE AND RIGHT HAND SIDE TO 0
IS I GREATER N
NO
B3
YES
IS ARGUMENT RANGE 0
YES
B2
20
SET UP COEFFICIENT MATRIX FROM SUMS
NO
CALCULATE IMPLIED LINEAR TRANSFORM.
SET ERROR PARAMETER EQUAL TO ZERO
INITIALIZE SUBSCRIPT I=1
B3
EXIT

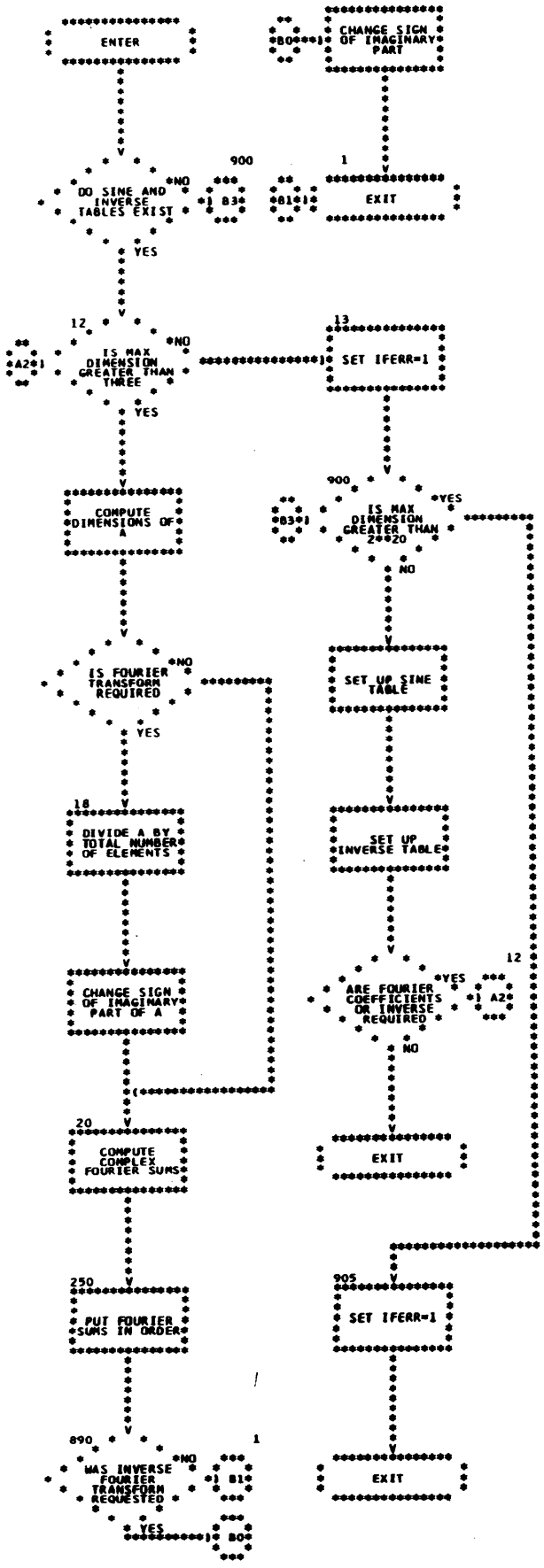

A1
ENTER ARAT
B1
INITIALIZATIONS CHECKS FOR FORMAL ERRORS
C1
INITIAL APPROXIMATION GIVEN
NO
YES
C2
DENOMINATOR=1 NUMERATOR=0
D1
COMPUTE VALUE OF INITIAL APPROXIMATION
E1
SET UP NORMAL EQUATIONS
E1
F1
IS DENOMINATOR ZERO
YES
NO
F2
IER=-1 EXIT
G1
RESTORE MATRIX OF NORMAL EQUATIONS
H1
ITERATION COUNTER IS ZERO
NO
YES
J1
SAVE RESIDUE INITIALIZE DIAGONAL INCREMENT
K1
IS FIRST TERM OF MATRIX POSITIVE
NO
YES
A3
J2
WAS INCREMENT MODIFIED
NO
YES
A3
K2
ADD INCREMENT TO DIAGONAL TERMS OF MATRIX
A3
A3
APFS SOLVES NORMAL EQUATIONS
B3
IS SOLUTION OBTAINED
NO
YES
C3
IS SOLUTION SIGNIFICANT
NO
YES
C4
VALUES OF CORRECTIONS OF NUMERATOR AND DENOMINATOR
D3
WAS INCREMENT MODIFIED
NO
YES
D4
SEARCH FOR IMPROVED RELAX. FACTOR
E3
MULTIPLY INCREMENT BY 1/8
E4
IS A RELAX FACTOR FOUND
NO
YES
F3
MULTIPLY INCREMENT BY 2
F4
UPDATE COEFFICIENTS OF APPROXIMATION
G3
MAX. ITERATION NUMBER IS REACHED
NO
YES
H3
IER=1 EXIT
J4
IER=0 EXIT
K4
IER=2 EXIT
F5
WAS INCREMENT MODIFIED
YES
NO
G5
IS VARIATION OF RESIDUE SMALL
YES
NO
H5
IS VARIATION OF COEF. SMALL
YES
NO
J5
IS RESIDUE SMALL
YES
NO
K5
IS MAX. ITERATION NUMBER REACHED
YES
NO
E1

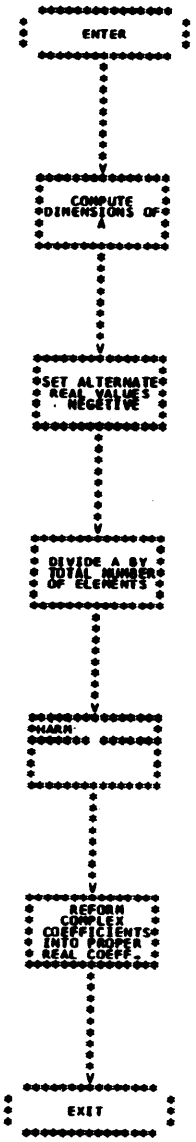

ENTER
TAKE DIMENSIONS OF NUMERATOR AND DENOMINATOR FROM IER
LOOK UP I-TH VALUES OF ARGUMENT FUNCTION. WEIGHT
LOOK UP I-TH VALUE OF NUMERATOR AND DENOMINATOR
IS DENOMINATOR ZERO
NO
YES
4
CALCULATE I-TH WEIGHT FACTOR
3
SET ERROR PARAMETER EQUAL TO ONE
CALCULATE FUNDAMENTAL FUNCTIONS FOR I-TH ARGUMENT
EXIT
STORE MODIFIED DENOMINATOR FUNDAMENTAL FUNCTIONS
10
STORE I-TH FUNCTION VALUE
SET ERROR PARAMETER EQUAL TO ZERO
EXIT

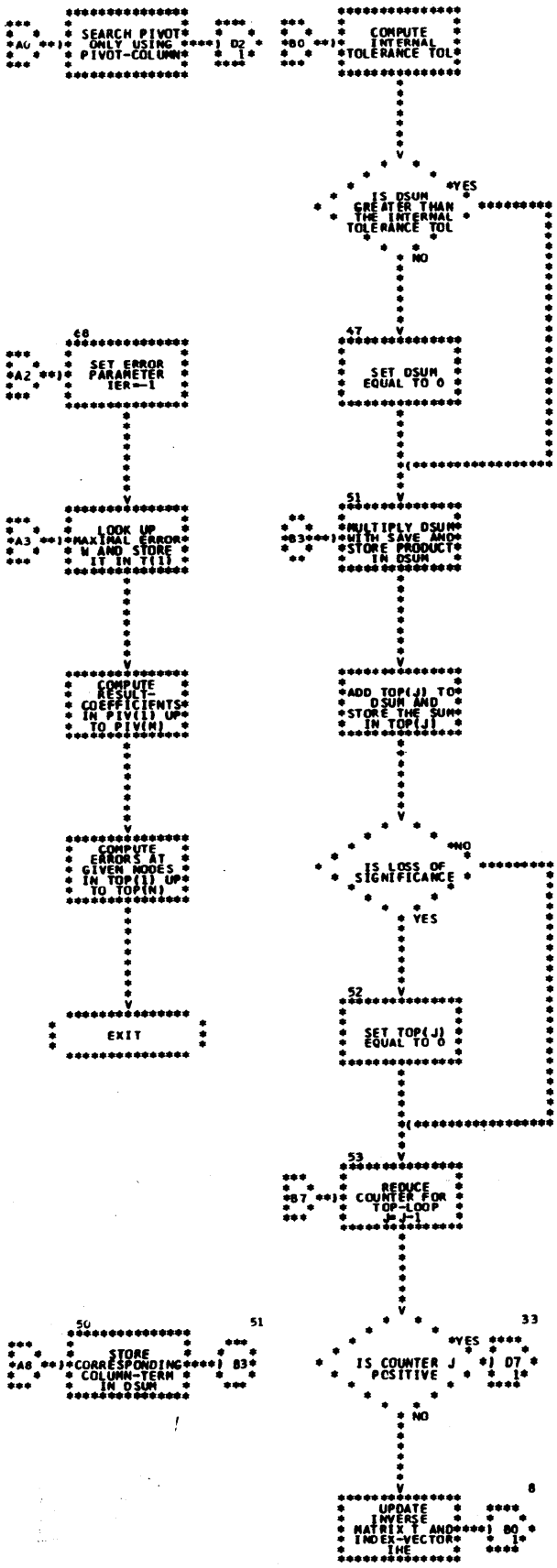

ENTER
IS DIMENSION N POSITIVE
NO
YES
1
IS DIMENSION IP POSITIVE
NO
YES
2
IS DIMENSION IP GREATER THAN N
YES
NO
10
SET ERROR PARAMETER EQUAL TO -1
EXIT
SET COEFFICIENT MATRIX AND RIGHTHANDSIDE EQUAL 0
INITIALIZE I EQUAL 1 ERROR PARAMETER EQUAL 0
A6
FPCT CALCULATES WEIGHT,BASIS FUNCTION FOR I-TH NODE
IS ERROR PARAMETER EQUAL 0
YES
NO
ACCUMULATE I-TH CONTRIBUTION IN NORMAL EQUATIONS
EXIT
STEP POINTER I BY 1
IS POINTER I GREATER THAN N
YES
NO
EXIT
A6

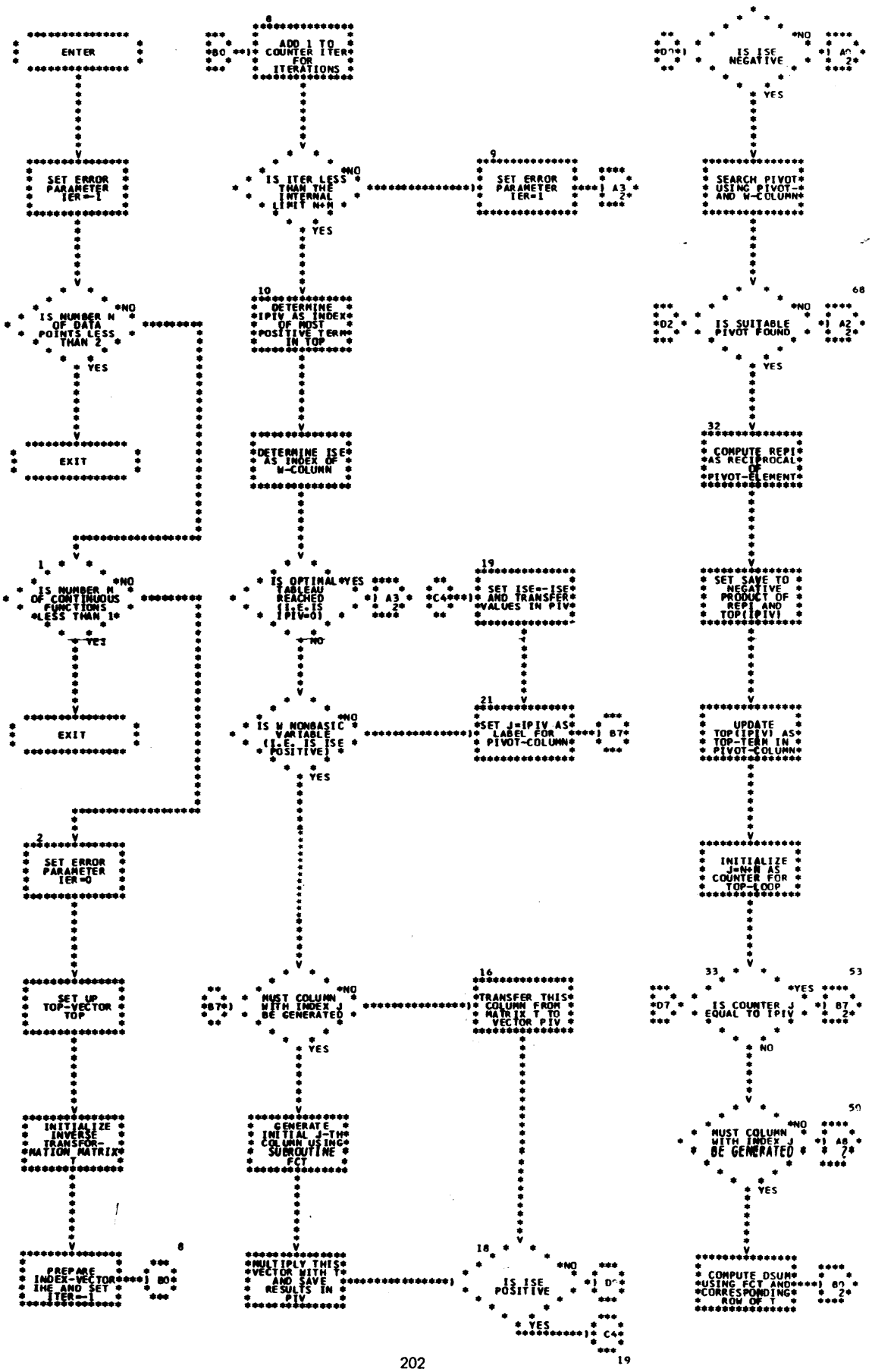

ENTER
SET ERROR CODE = 0
IS NU OF POINTS DEFINING INTERVAL (H) NEGATIVE
*NO
YES
30
SET ERROR CODE = 2
EXIT
40
IS M GREATER THAN N
*NO
YES
SET ERROR CODE = 1
EXIT
60
COMPUTE S1 EQUATION (2)
COMPUTE C1 EQUATION (1)
B0
SET C = 1, S = 0, J = 1
70
B1
SET U2 = 0, U1 = 0, I = N
B2
FUN OBTAIN NEXT VALUE OF FUN
COMPUTE U0 EQUATION (3)
REPLACE U2 BY U1 AND U1 BY U0
SET I = I-1
I = 0
*NO
YES
80
COMPUTE FOURIER COEFFS - A(J), B(J) EQ.(4),(5)
J = M&1
*NO
90
C1
YES
100
MODIFY *CONSTANT TERM*
C0
EXIT
90
C1
COMPUTE NEW CCS. C = C1*C-S1*S
COMPUTE NEW SIN. S = C1*S&S1*C
REPLACE C BY C
SET J = J&1
70
B1

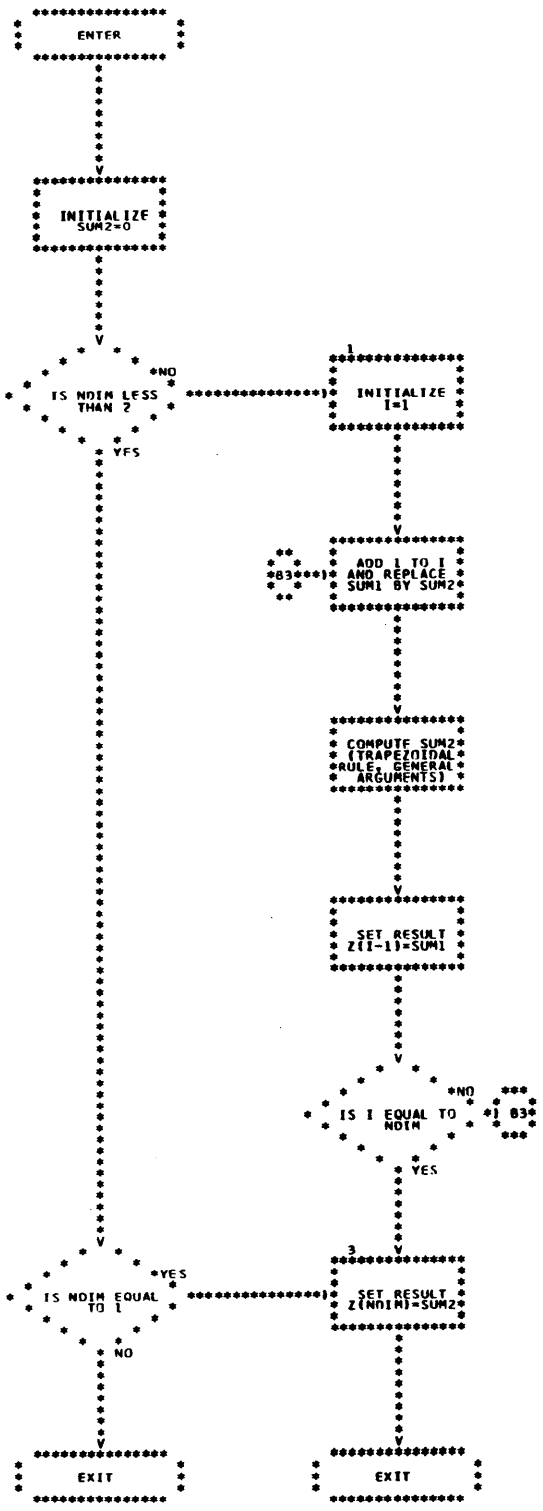

ENTER
SET ERROR CODE = 0
IS NO OF POINTS DEFINING INTERVAL (M) NEGATIVE
NO
YES
30
SET ERROR CODE = 2
EXIT
40
IS M GREATER THAN N
NO
YES
SET ERROR CODE = 1
EXIT
60
COMPUTE S1 EQUATION (2)
COMPUTE C1 EQUATION (1)
80
SET C = 1, S = 0, J = 1
70
SET U2 = 0, U1 = 0, I = 2*N
82
OBTAIN I-TH VALUE OF FUNCTION
COMPUTE U0 EQUATION (3)
REPLACE U2 BY U1 AND U1 BY U0
SET I = I-1
I = 0
NO
82
YES
85
COMPUTE FOURIER COEFFS - A(J), B(J), EQ.(4),(5)
90
J = M&1
NO
C1
YES
100
MODIFY CONSTANT TERM
CO
EXIT
90
C1
COMPUTE NEW COS. Q = C1*C-S1*S
COMPUTE NEW SIN. S = C1*S&S1*C
REPLACE C BY Q
70
SET J = J&1
B1

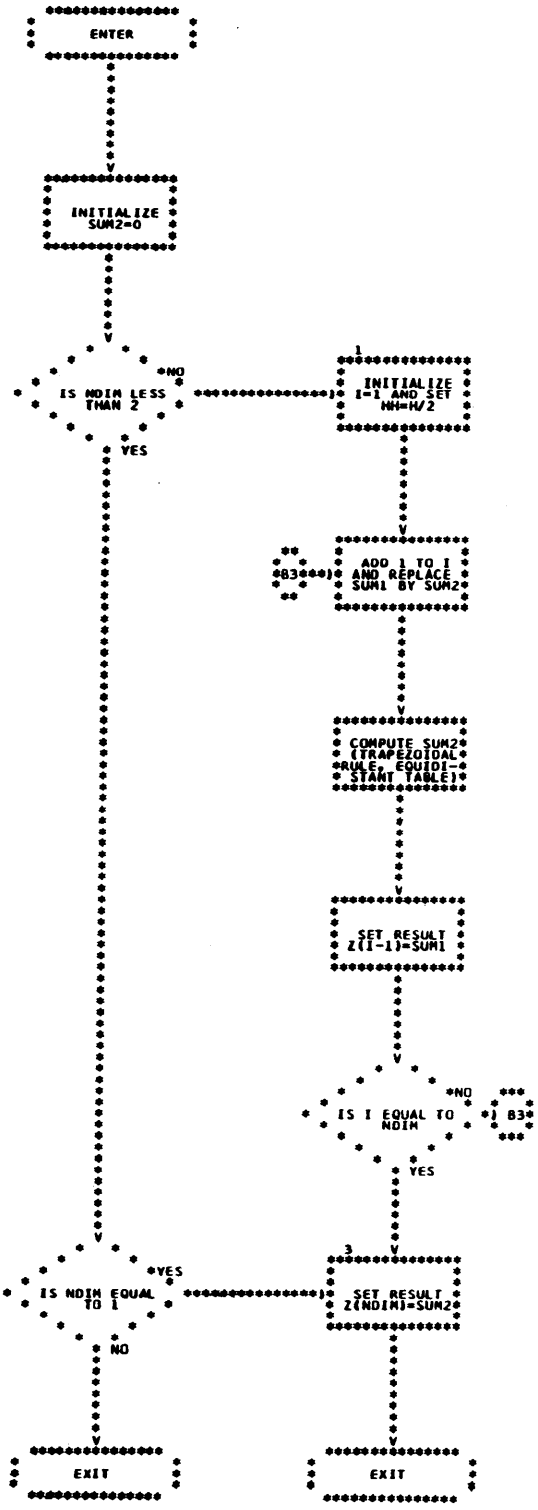

ENTER
DO SINE AND INVERSE TABLES EXIST
NO
YES
900
B3
IS MAX DIMENSION GREATER THAN THREE
A2
NO
YES
12
13
SET IFERR=1
COMPUTE DIMENSIONS OF A
IS FOURIER TRANSFORM REQUIRED
NO
YES
18
DIVIDE A BY TOTAL NUMBER OF ELEMENTS
CHANGE SIGN OF IMAGINARY PART OF A
20
COMPUTE COMPLEX FOURIER SUMS
250
PUT FOURIER SUMS IN ORDER
890
WAS INVERSE FOURIER TRANSFORM REQUESTED
NO
YES
1
B1
B0
CHANGE SIGN OF IMAGINARY PART
B0
B1
EXIT
900
B3
IS MAX DIMENSION GREATER THAN 2**20
YES
NO
SET UP SINE TABLE
SET UP INVERSE TABLE
ARE FOURIER COEFFICIENTS OR INVERSE REQUIRED
YES
NO
12
A2
EXIT
905
SET IFERR=1
EXIT

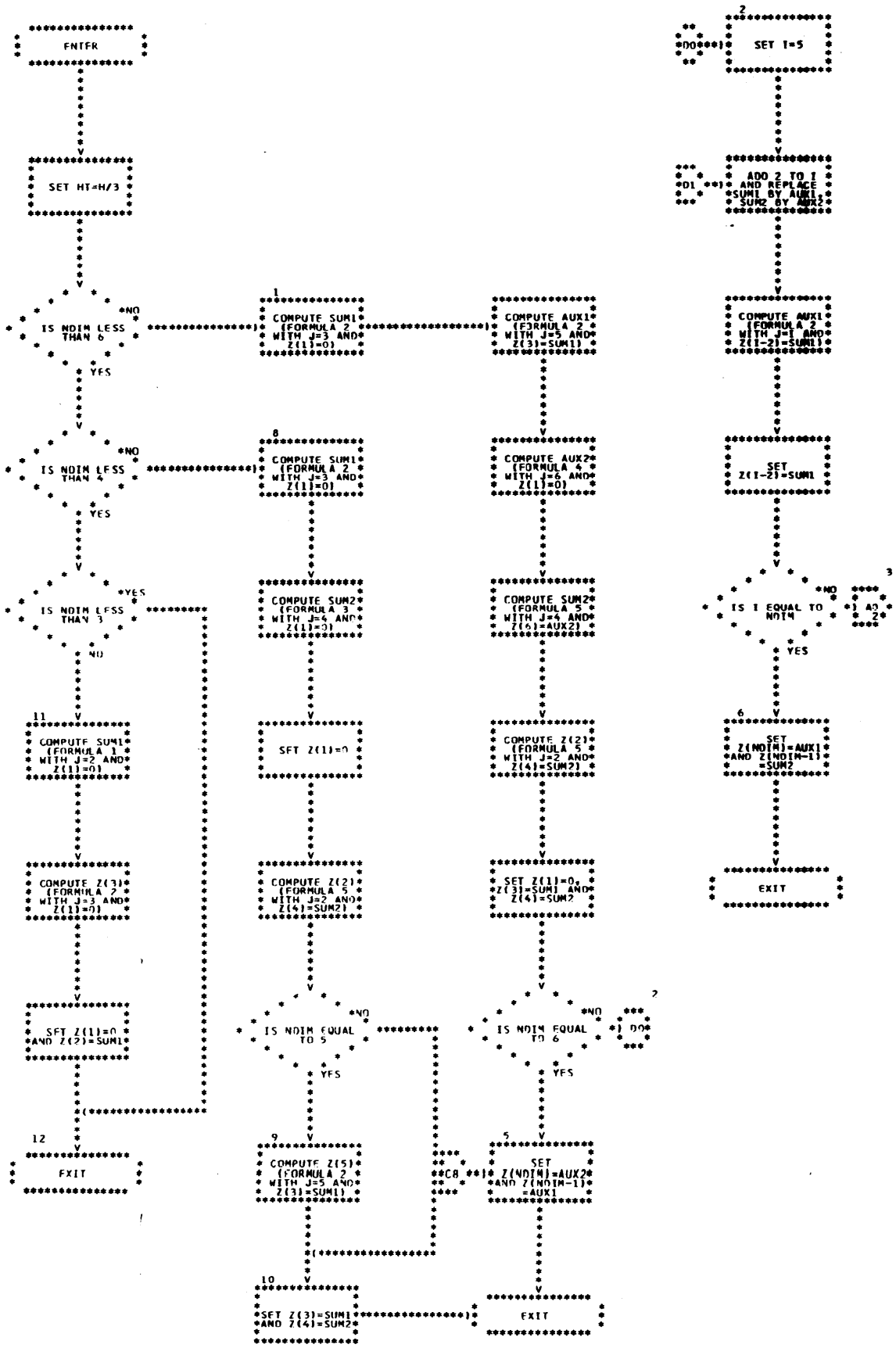

ENTER
COMPUTE DIMENSIONS OF A
SET ALTERNATE REAL VALUES NEGETIVE
DIVIDE A BY TOTAL NUMBER OF ELEMENTS
RHARM
REFORM COMPLEX COEFFICIENTS INTO PROPER REAL COEFF.
EXIT

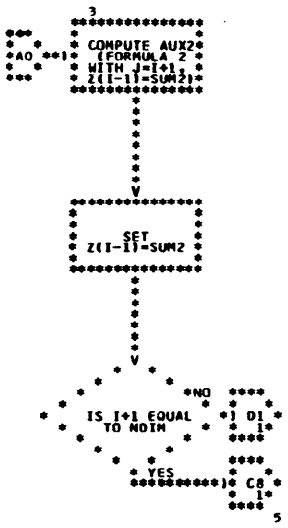

SEARCH PIVOT ONLY USING PIVOT-COLUMN
A0
D2
1
COMPUTE INTERNAL TOLERANCE TOL
B0
IS DSUM GREATER THAN THE INTERNAL TOLERANCE TOL
YES
NO
68
SET ERROR PARAMETER IER=-1
A2
47
SET DSUM EQUAL TO 0
LOOK UP MAXIMAL ERROR W AND STORE IT IN T(1)
A3
51
MULTIPLY DSUM WITH SAVE AND STORE PRODUCT IN DSUM
B3
COMPUTE RESULT-COEFFICIENTS IN PIV(1) UP TO PIV(N)
ADD TOP(J) TO DSUM AND STORE THE SUM IN TOP(J)
COMPUTE ERRORS AT GIVEN NODES IN TOP(1) UP TO TOP(N)
IS LOSS OF SIGNIFICANCE
NO
YES
EXIT
52
SET TOP(J) EQUAL TO 0
53
REDUCE COUNTER FOR TOP-LOOP J=J-1
B7
50
51
STORE CORRESPONDING COLUMN-TERM IN DSUM
A8
B3
33
IS COUNTER J POSITIVE
YES
D7
1
NO
8
UPDATE INVERSE MATRIX T AND INDEX-VECTOR THE
B0
1

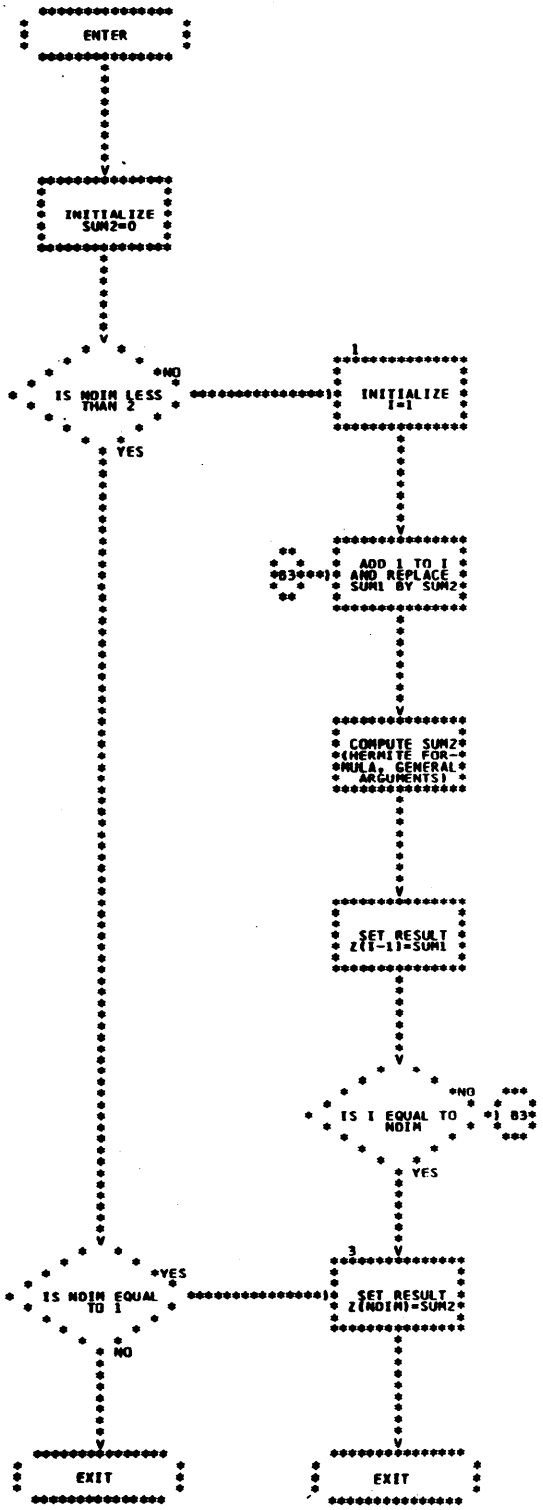

ENTER
SET ERROR PARAMETER IER=1
IS NUMBER N OF DATA POINTS LESS THAN 2
NO
YES
EXIT
IS NUMBER M OF CONTINUOUS FUNCTIONS LESS THAN 1
NO
YES
EXIT
SET ERROR PARAMETER IER=0
SET UP TOP-VECTOR TOP
INITIALIZE INVERSE TRANSFOR-MATION MATRIX T
PREPARE INDEX-VECTOR IHE AND SET ITER=1
8
ADD 1 TO COUNTER ITER FOR ITERATIONS
IS ITER LESS THAN THE INTERNAL LIMIT N+M
NO
YES
9
SET ERROR PARAMETER IER=1
A3
2
10
DETERMINE IPIV AS INDEX OF MOST POSITIVE TERM IN TOP
DETERMINE ISE AS INDEX OF W-COLUMN
IS OPTIMAL TABLEAU REACHED (I.E.IS IPIV=0)
YES
A3
2
C4
NO
IS W NONBASIC VARIABLE (I.E. IS ISE POSITIVE)
NO
YES
MUST COLUMN WITH INDEX J BE GENERATED
NO
YES
B7
GENERATE INITIAL J-TH COLUMN USING SUBROUTINE FCT
MULTIPLY THIS VECTOR WITH T AND SAVE RESULTS IN PIV
19
SET ISE=-ISE AND TRANSFER VALUES IN PIV
21
SET J=IPIV AS LABEL FOR PIVOT-COLUMN
B7
16
TRANSFER THIS COLUMN FROM MATRIX T TO VECTOR PIV
18
IS ISE POSITIVE
NO
D
YES
C4
19
IS ISE NEGATIVE
NO
A2
2
YES
SEARCH PIVOT USING PIVOT-AND W-COLUMN
68
IS SUITABLE PIVOT FOUND
NO
A2
2
YES
32
COMPUTE REPI AS RECIPROCAL OF PIVOT-ELEMENT
SET SAVE TO NEGATIVE PRODUCT OF REPI AND TOP(IPIV)
UPDATE TOP(IPIV) AS TOP-TERM IN PIVOT-COLUMN
INITIALIZE J=N+M AS COUNTER FOR TOP-LOOP
33
53
IS COUNTER J EQUAL TO IPIV
YES
B7
2
NO
50
MUST COLUMN WITH INDEX J BE GENERATED
NO
A8
2
YES
COMPUTE DSUM USING FCT AND CORRESPONDING ROW OF T
B8
2

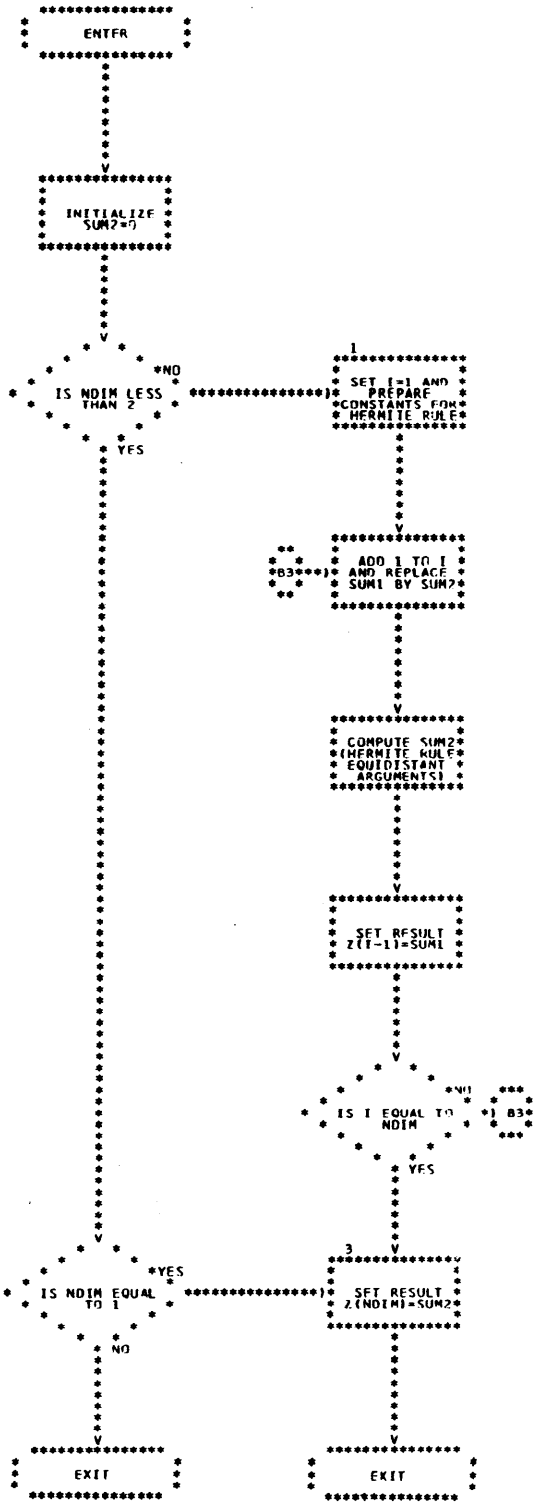

ENTER
INITIALIZE SUM2=0
IS NDIM LESS THAN 2
NO
YES
1
INITIALIZE I=1
ADD 1 TO I AND REPLACE SUM1 BY SUM2
83
COMPUTE SUM2 (TRAPEZOIDAL RULE, GENERAL ARGUMENTS)
SET RESULT Z(I-1)=SUM1
IS I EQUAL TO NDIM
NO
83
YES
IS NDIM EQUAL TO 1
YES
NO
3
SET RESULT Z(NDIM)=SUM2
EXIT
EXIT

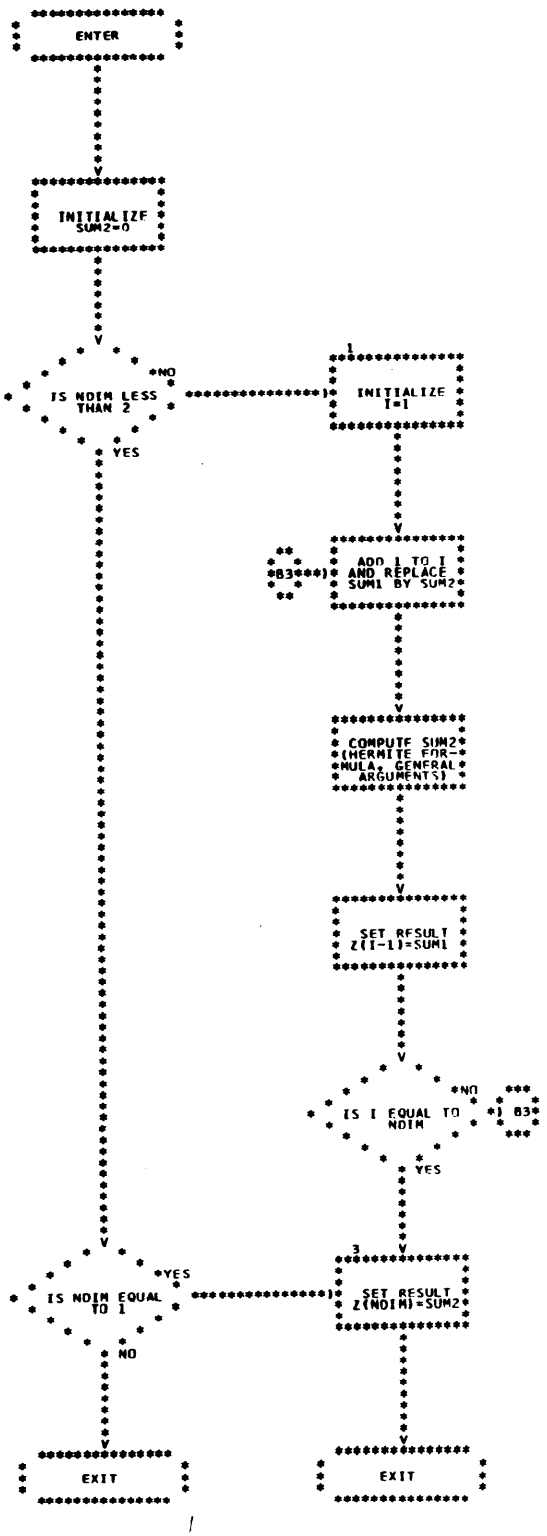

ENTER
INITIALIZE SUM2=0
IS NDIM LESS THAN 2
*NO
YES
1
INITIALIZE I=1 AND SET HH=H/2
ADD 1 TO I AND REPLACE SUM1 BY SUM2
B3
COMPUTE SUM2 (TRAPEZOIDAL RULE, EQUIDI-STANT TABLE)
SET RESULT Z(I-1)=SUM1
IS I EQUAL TO NDIM
*NO
B3
YES
3
IS NDIM EQUAL TO 1
*YES
SET RESULT Z(NDIM)=SUM2
NO
EXIT
EXIT

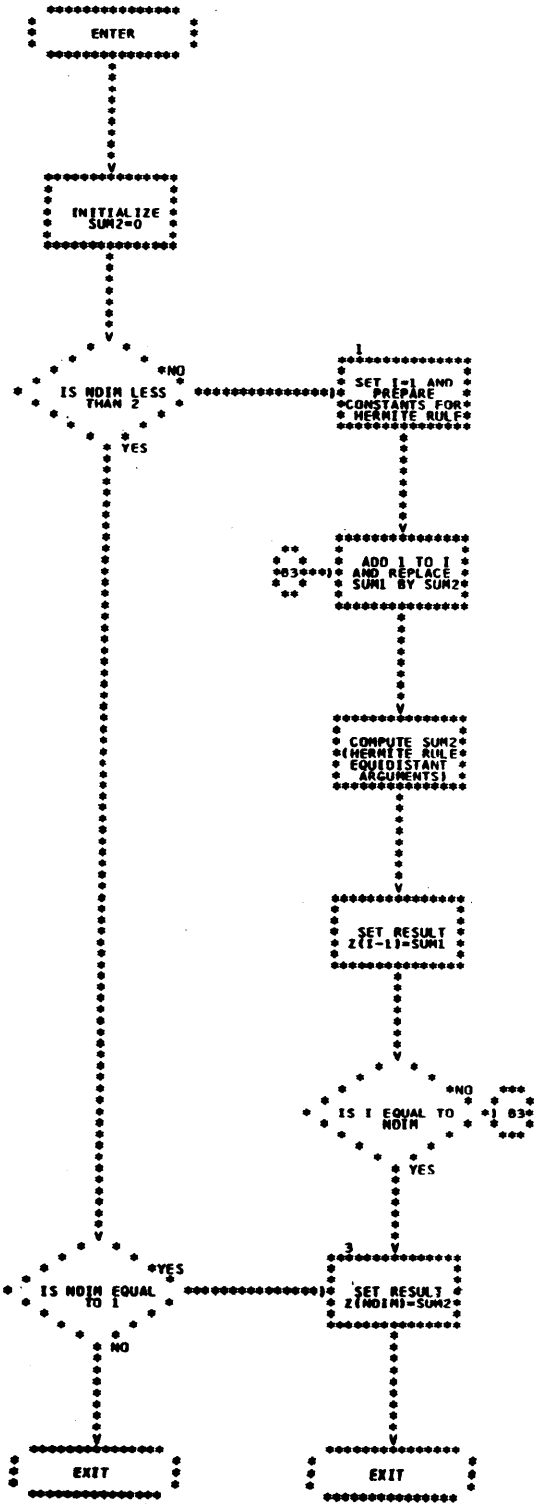

ENTER
SET HT=H/3
IS NDIM LESS THAN 6
NO
YES
IS NDIM LESS THAN 4
NO
YES
IS NDIM LESS THAN 3
YES
NO
COMPUTE SUM1 (FORMULA 1 WITH J=2 AND Z(1)=0)
COMPUTE Z(3) (FORMULA 2 WITH J=3 AND Z(1)=0)
SET Z(1)=0 AND Z(2)=SUM1
EXIT
COMPUTE SUM1 (FORMULA 2 WITH J=3 AND Z(1)=0)
COMPUTE SUM1 (FORMULA 2 WITH J=3 AND Z(1)=0)
COMPUTE SUM2 (FORMULA 3 WITH J=4 AND Z(1)=0)
SET Z(1)=0
COMPUTE Z(2) (FORMULA 5 WITH J=2 AND Z(4)=SUM2)
IS NDIM EQUAL TO 5
YES
COMPUTE Z(5) (FORMULA 2 WITH J=5 AND Z(3)=SUM1)
SET Z(3)=SUM1 AND Z(4)=SUM2
EXIT
COMPUTE AUX1 (FORMULA 2 WITH J=5 AND Z(3)=SUM1)
COMPUTE AUX2 (FORMULA 4 WITH J=6 AND Z(1)=0)
COMPUTE SUM2 (FORMULA 5 WITH J=4 AND Z(6)=AUX2)
COMPUTE Z(2) (FORMULA 5 WITH J=2 AND Z(4)=SUM2)
SET Z(1)=0, Z(3)=SUM1 AND Z(4)=SUM2
IS NDIM EQUAL TO 6
NO
DO
YES
SET Z(NDIM)=AUX2 AND Z(NDIM-1)=AUX1
EXIT
SET I=5
DO
ADD 2 TO I AND REPLACE SUM1 BY AUX1, SUM2 BY AUX2
DO
COMPUTE AUX1 (FORMULA 2 WITH J=I AND Z(I-2)=SUM1)
SET Z(I-2)=SUM1
IS I EQUAL TO NDIM
NO
A3
YES
SET Z(NDIM)=AUX1 AND Z(NDIM-1)=SUM2
EXIT

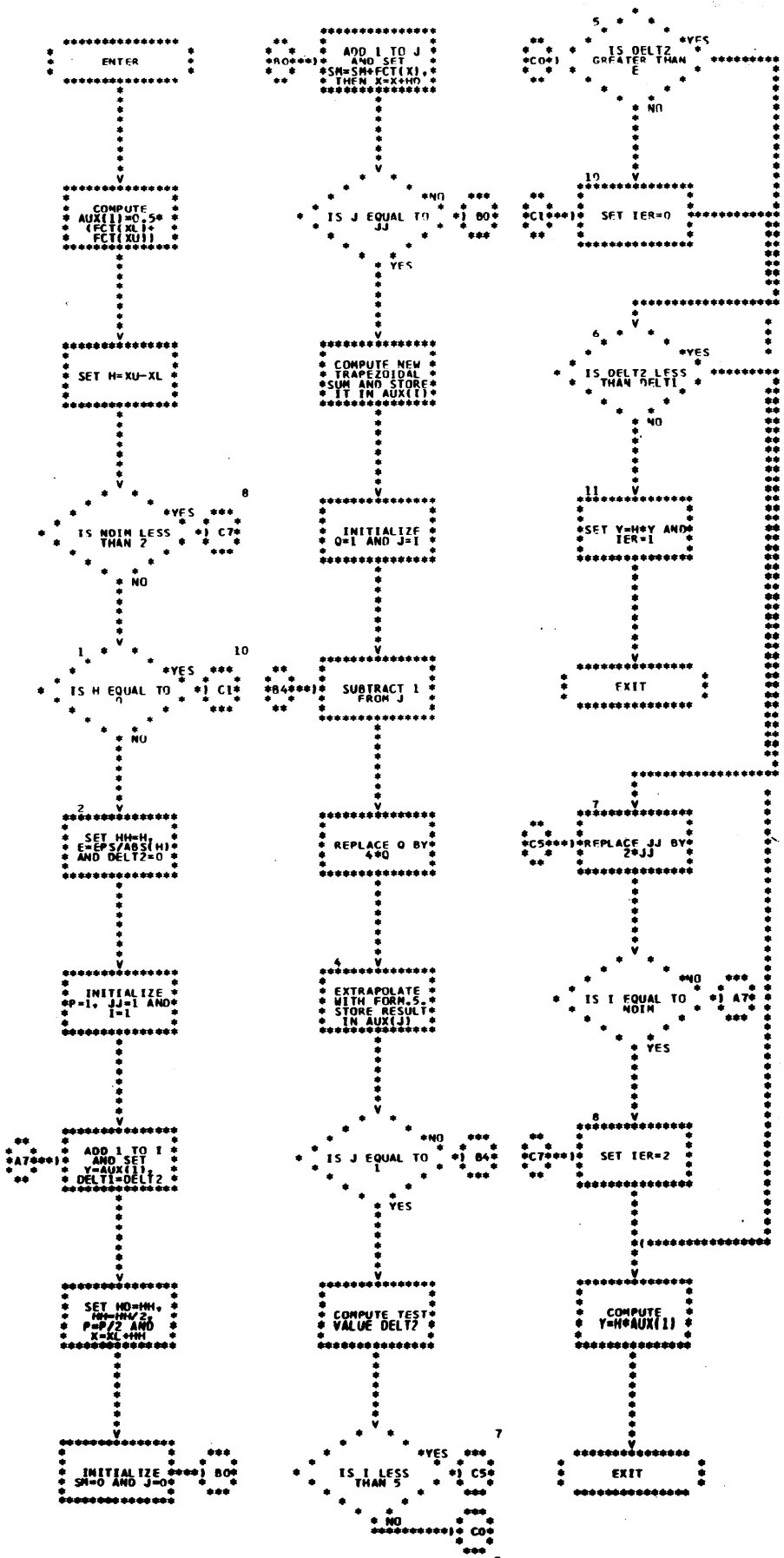

3
COMPUTE AUX2
(FORMULA 2
WITH J=I+1
Z(I-1)=SUM2)
AO
SET
Z(I-1)=SUM2
NO
IS I+1 EQUAL
TO NDIM
DI
I
YES
C8
I
5

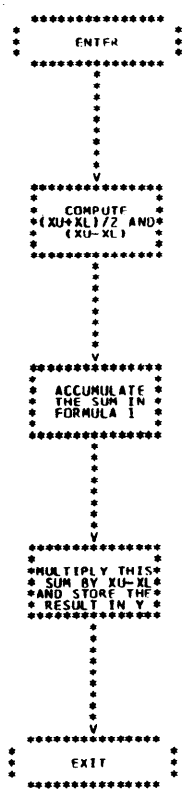

ENTER
INITIALIZE SUM2=0
IS NDIM LESS THAN 2
NO
YES
1
INITIALIZE I=1
B3
ADD 1 TO I AND REPLACE SUM1 BY SUM2
COMPUTE SUM2 (HERMITE FORMULA, GENERAL ARGUMENTS)
SET RESULT Z(I-1)=SUM1
IS I EQUAL TO NDIM
NO
YES
B3
IS NDIM EQUAL TO 1
YES
NO
3
SET RESULT Z(NDIM)=SUM2
EXIT
EXIT


ENTER
INITIALIZE SUM2=0
IS NDIM LESS THAN 2
*NO
YES
1
SET I=1 AND PREPARE CONSTANTS FOR HERMITE RULE
ADD 1 TO I AND REPLACE SUM1 BY SUM2
*B3
COMPUTE SUM2 (HERMITE RULE EQUIDISTANT ARGUMENTS)
SET RESULT Z(I-1)=SUM1
IS I EQUAL TO NDIM
*NO
*B3
YES
3
IS NDIM EQUAL TO 1
*YES
NO
SET RESULT Z(NDIM)=SUM2
EXIT
EXIT

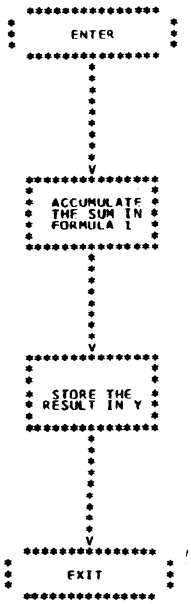

ENTER
INITIALIZE SUM2=0
IS NDIM LESS THAN 2
*NO
YES
1
INITIALIZE I=1
ADD 1 TO I AND REPLACE SUM1 BY SUM2
*B3***
COMPUTE SUM2 (HERMITE FORMULA, GENERAL ARGUMENTS)
SET RESULT Z(I-1)=SUM1
IS I EQUAL TO NDIM
*NO
1 B3
YES
3
IS NDIM EQUAL TO 1
*YES
NO
SET RESULT Z(NDIM)=SUM2
EXIT
EXIT


ENTER
INITIALIZE SUM2=0
IS NDIM LESS THAN 2
*NO
YES
1
SET I=1 AND PREPARE CONSTANTS FOR HERMITE RULE
ADD 1 TO I AND REPLACE SUM1 BY SUM2
*83
COMPUTE SUM2 (HERMITE RULE EQUIDISTANT ARGUMENTS)
SET RESULT Z(I-1)=SUM1
IS I EQUAL TO NDIM
*NO
*1 83
YES
3
IS NDIM EQUAL TO 1
*YES
NO
SET RESULT Z(NDIM)=SUM2
EXIT
EXIT

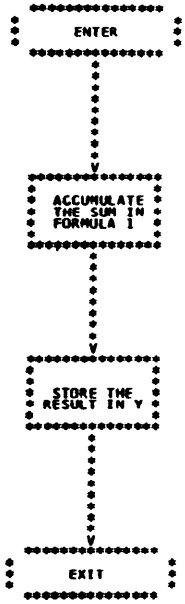

ENTER
COMPUTE AUX(I)=0.5* (FCT(XL)+ FCT(XU))
SET H=XU-XL
IS NOIM LESS THAN 2
YES
NO
C7
8
IS H EQUAL TO 0
YES
NO
C1
10
SET HH=H, E=EPS/ABS(H) AND DELT2=0
INITIALIZE P=1, JJ=1 AND I=1
ADD 1 TO I AND SET Y=AUX(I), DELT1=DELT2
SET HH=HH, H=HH/2, P=P/2 AND X=XL+HH
INITIALIZE SM=0 AND J=0
B0
ADD 1 TO J AND SET SM=SM+FCT(X), THEN X=X+HH
IS J EQUAL TO JJ
NO
YES
B0
COMPUTE NEW TRAPEZOIDAL SUM AND STORE IT IN AUX(I)
INITIALIZE Q=1 AND J=1
SUBTRACT 1 FROM J
B4
REPLACE Q BY 4*Q
EXTRAPOLATE WITH FORM.5, STORE RESULT IN AUX(J)
IS J EQUAL TO 1
NO
YES
B4
COMPUTE TEST VALUE DELT2
IS I LESS THAN 5
YES
NO
C5
C0
5
IS DELT2 GREATER THAN E
YES
NO
C0
5
SET IER=0
C1
10
IS DELT2 LESS THAN DELT1
YES
NO
6
SET Y=H*Y AND IER=1
11
EXIT
REPLACE JJ BY 2*JJ
C5
7
IS I EQUAL TO NOIM
NO
YES
A7
SET IER=2
C7
8
COMPUTE Y=H*AUX(I)
EXIT



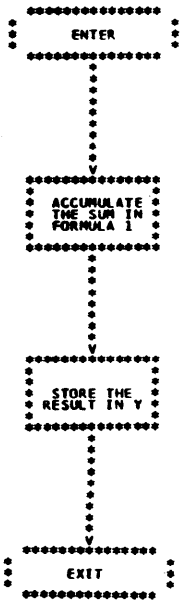
SUBROUTINES QL2,QL3,...,QL10 AND DQL4,DQL8,...,DQL32

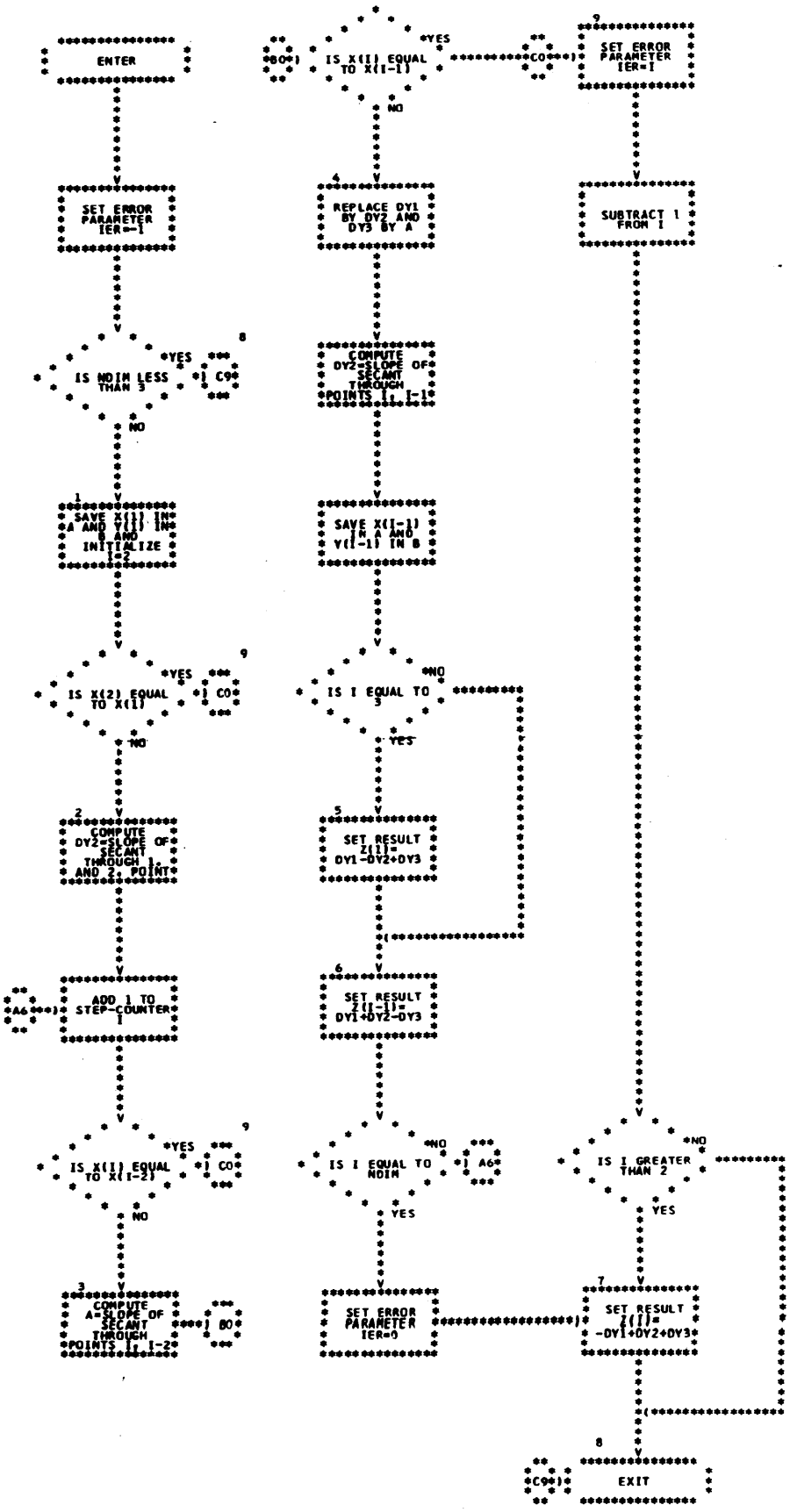


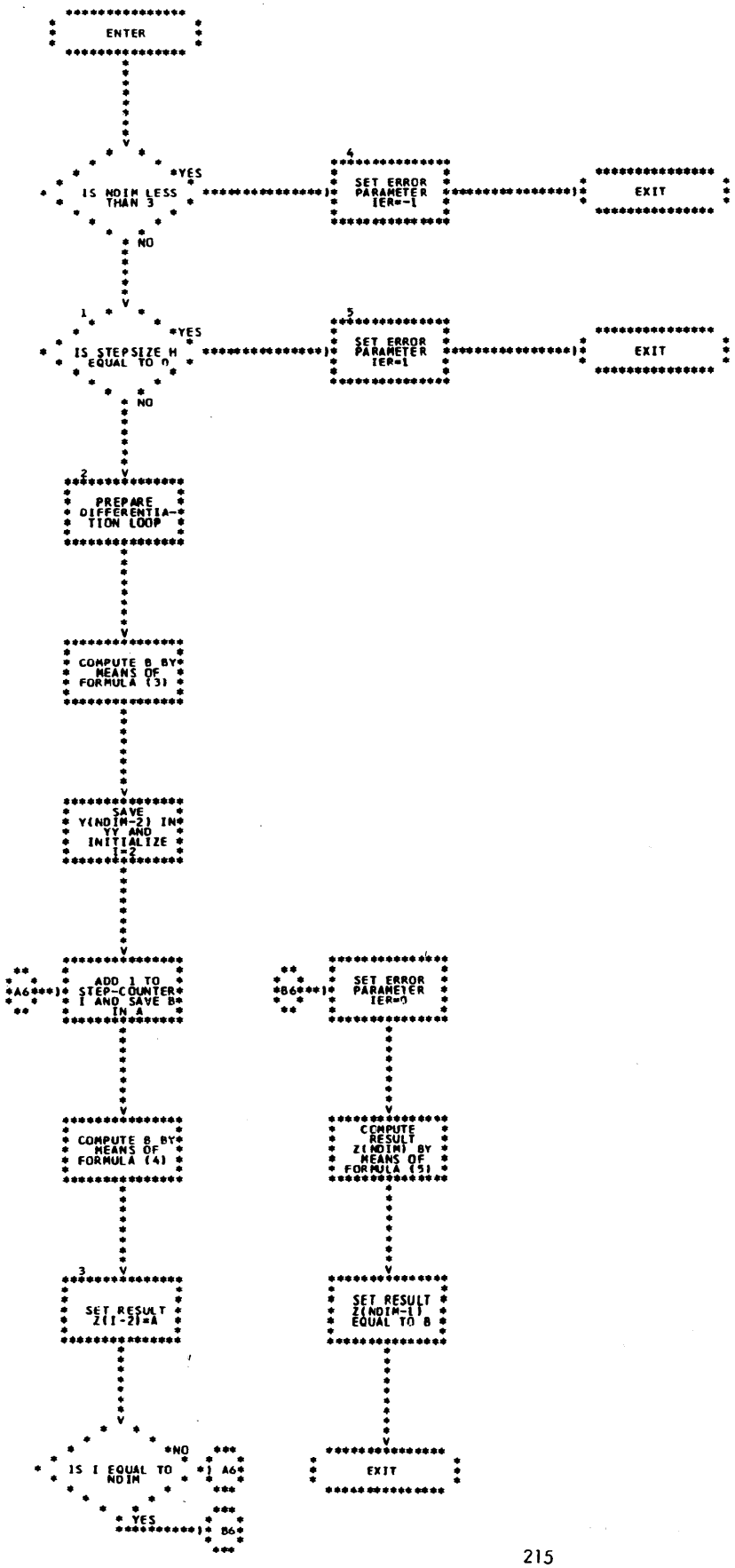
SUBROUTINES QH2,QH3,....QH10 AND DQH8,DQH16,...,DQH64

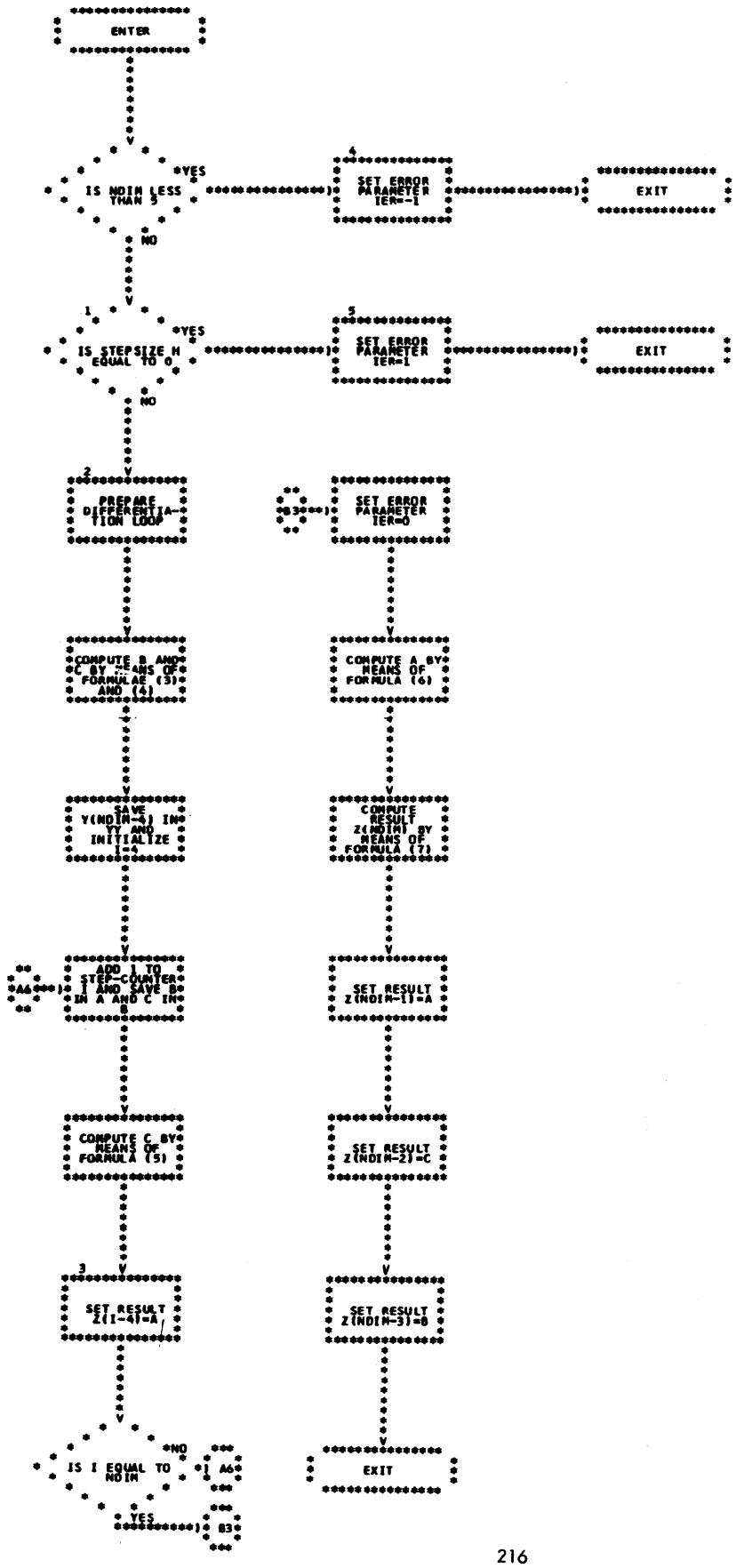


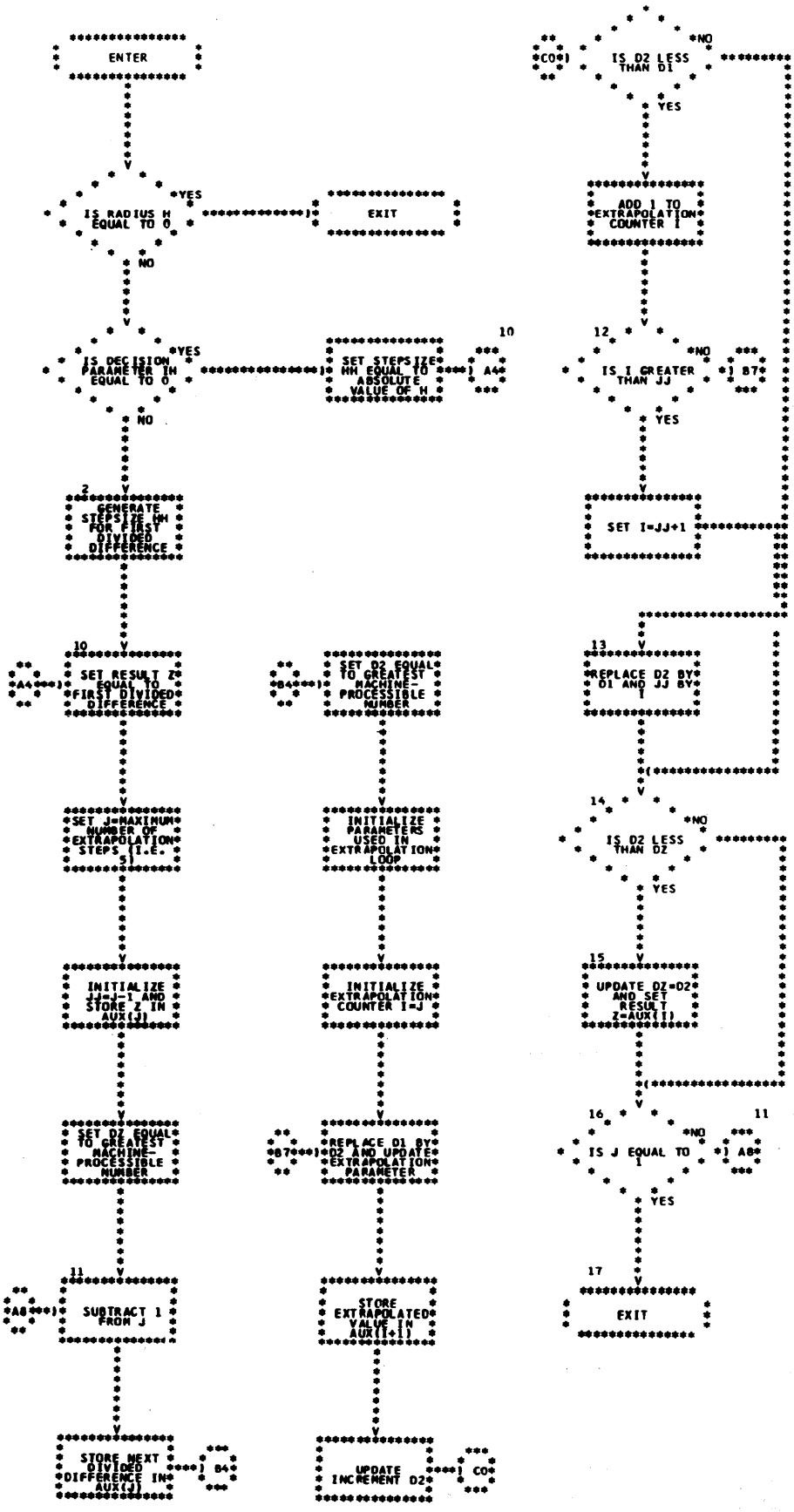
SUBROUTINES QA2,QA3,....QA10 AND DQA4,DQA8,....DQA32

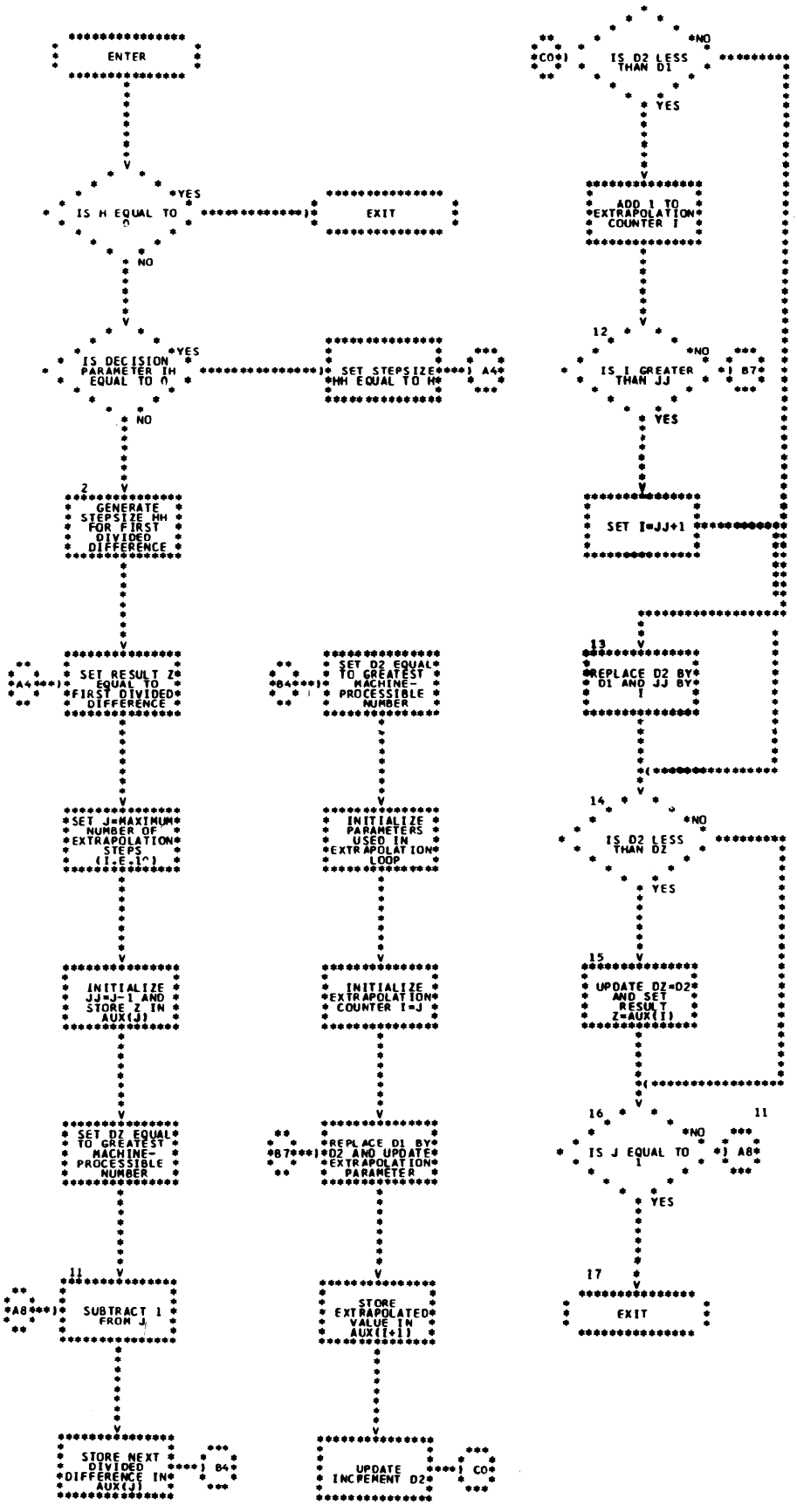


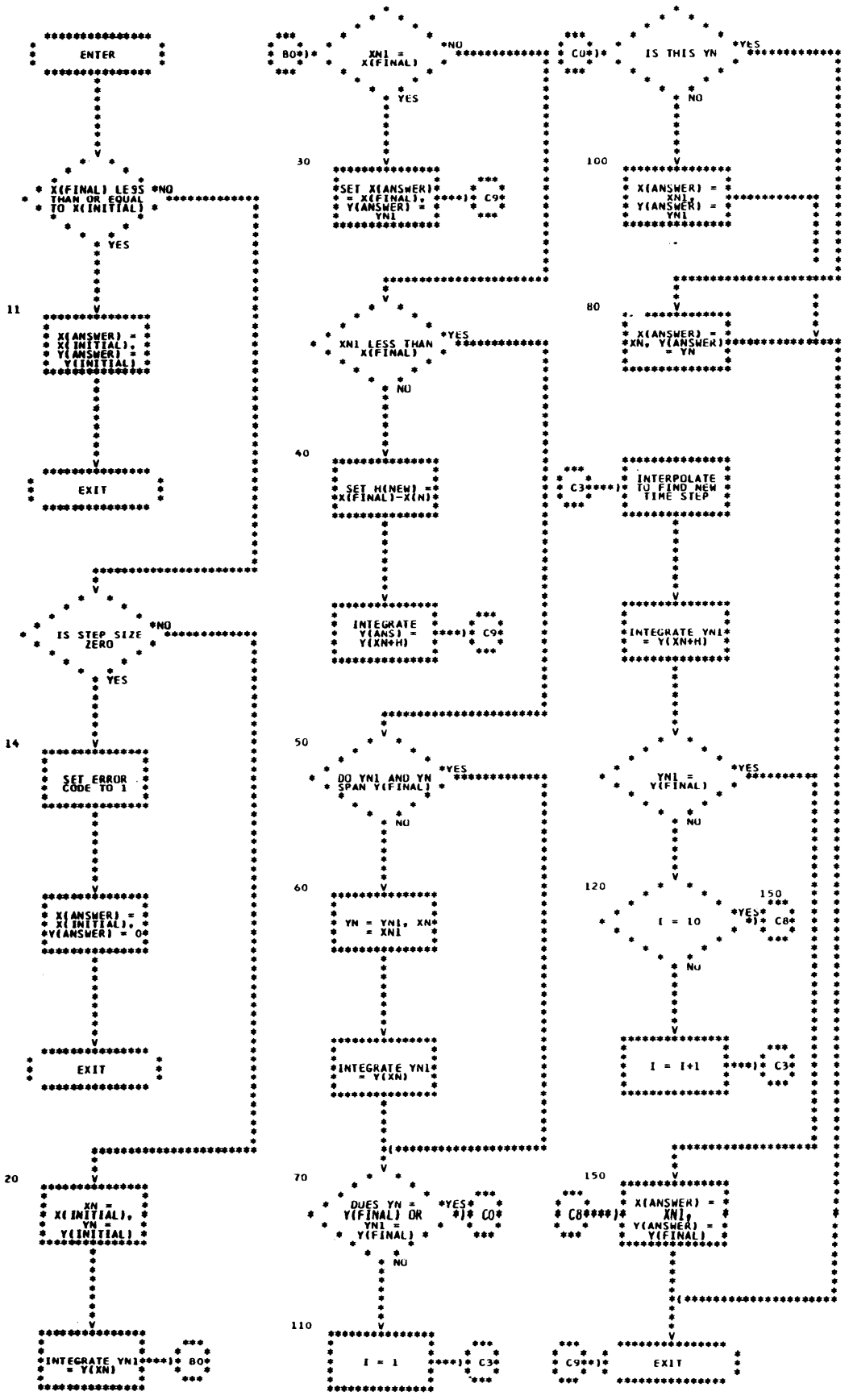

ENTER
SET ERROR PARAMETER IER=-1
IS NOIN LESS THAN 3
YES
NO
C9
SAVE X(1) IN A AND Y(1) IN B AND INITIALIZE I=2
IS X(2) EQUAL TO X(1)
YES
NO
C0
COMPUTE DY2=SLOPE OF SECANT THROUGH 1, AND 2, POINT
ADD 1 TO STEP-COUNTER I
A6
IS X(I) EQUAL TO X(I-2)
YES
NO
C0
COMPUTE A=SLOPE OF SECANT THROUGH POINTS I, I-2
B0
IS X(I) EQUAL TO X(I-1)
YES
NO
C0
REPLACE DY1 BY DY2 AND DY3 BY A
COMPUTE DY2=SLOPE OF SECANT THROUGH POINTS I, I-1
SAVE X(I-1) IN A AND Y(I-1) IN B
IS I EQUAL TO 3
NO
YES
SET RESULT Z(1)= DY1-DY2+DY3
SET RESULT Z(I-1)= DY1+DY2-DY3
IS I EQUAL TO NOIN
NO
A6
YES
SET ERROR PARAMETER IER=0
B0
SET ERROR PARAMETER IER=I
SUBTRACT 1 FROM I
IS I GREATER THAN 2
NO
YES
SET RESULT Z(I)= -DY1+DY2+DY3
C9
EXIT

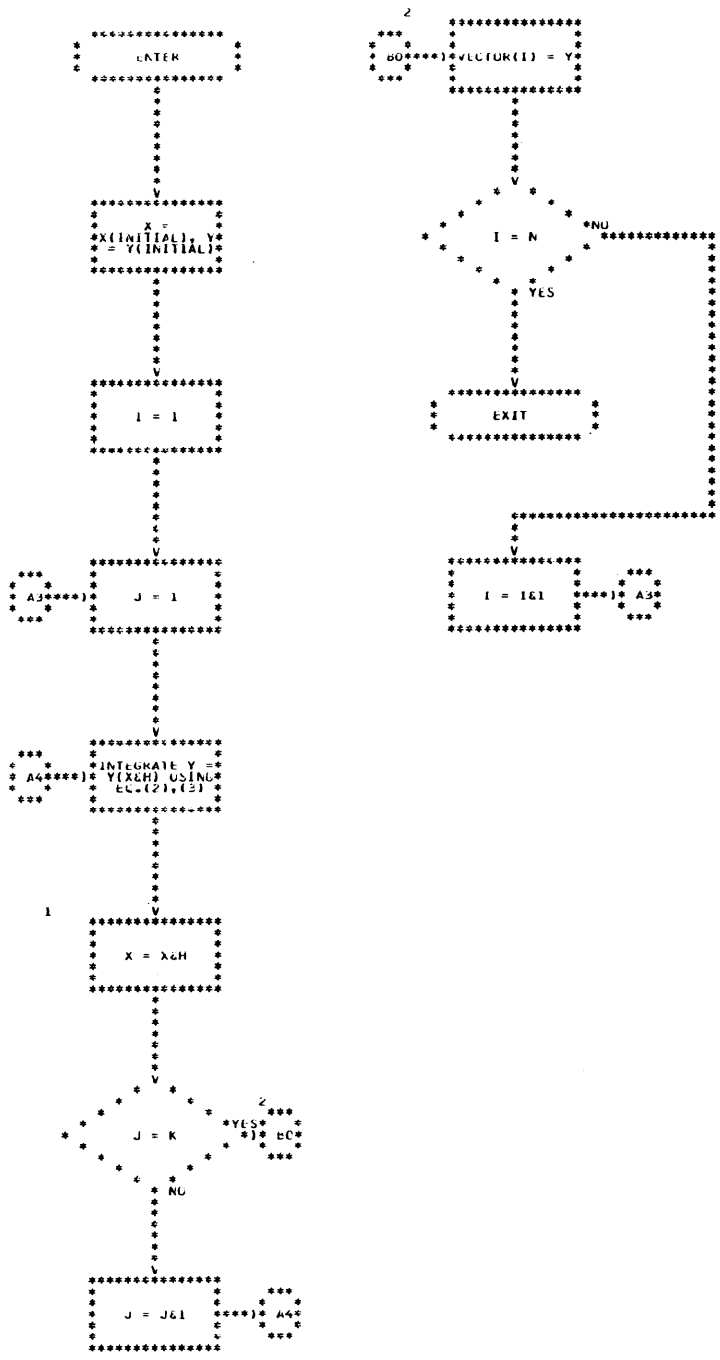

ENTER
IS NDIM LESS THAN 3
YES
NO
4
SET ERROR PARAMETER IER=-1
EXIT
1
IS STEPSIZE H EQUAL TO 0
YES
NO
5
SET ERROR PARAMETER IER=1
EXIT
2
PREPARE DIFFERENTIA-TION LOOP
COMPUTE B BY MEANS OF FORMULA (3)
SAVE Y(NDIM-2) IN YY AND INITIALIZE I=2
ADD 1 TO STEP-COUNTER I AND SAVE B IN A
A6
SET ERROR PARAMETER IER=0
B6
COMPUTE B BY MEANS OF FORMULA (4)
COMPUTE RESULT Z(NDIM) BY MEANS OF FORMULA (5)
3
SET RESULT Z(I-2)=A
SET RESULT Z(NDIM-1) EQUAL TO B
IS I EQUAL TO NDIM
NO
A6
YES
B6
EXIT

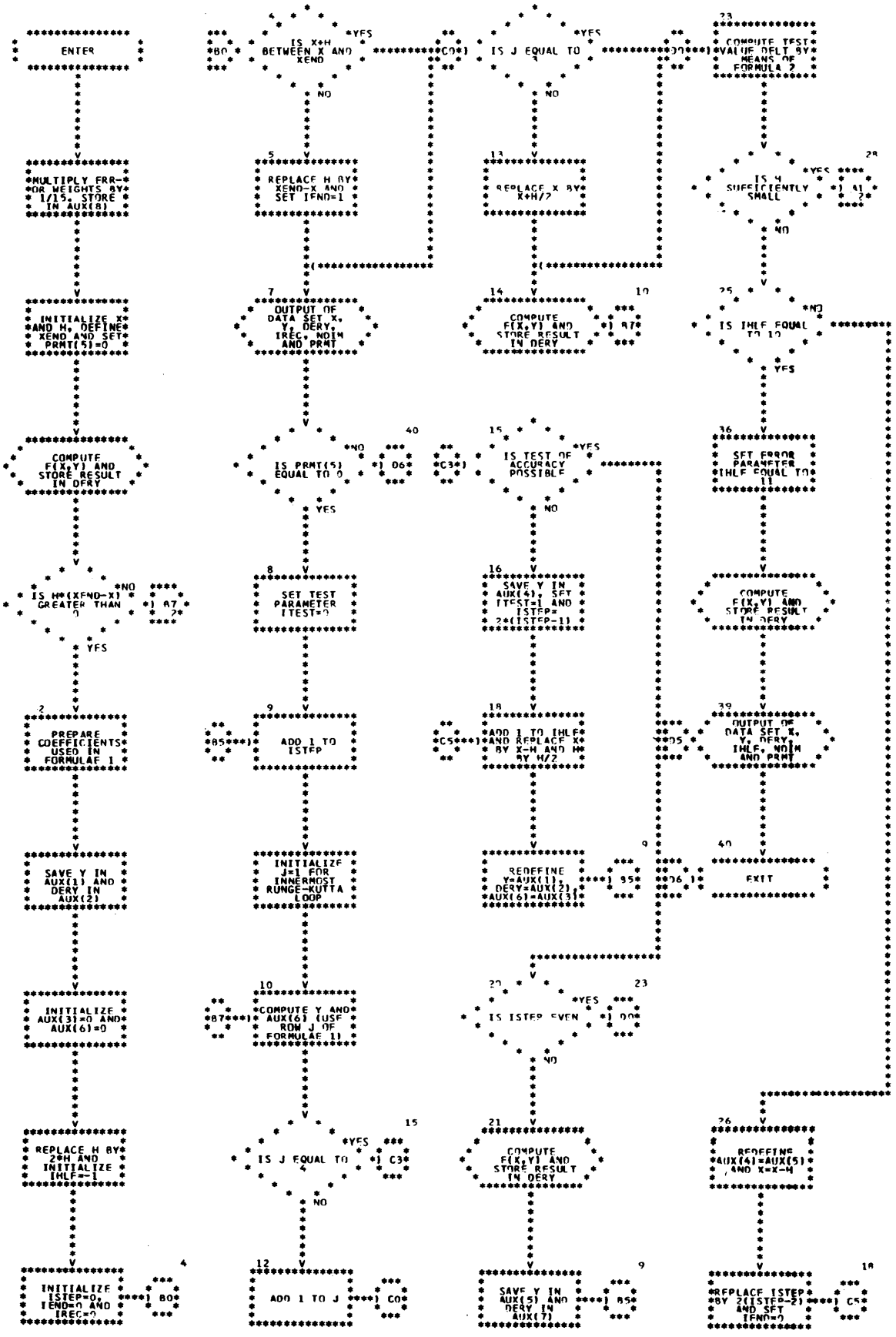

ENTER
IS NDIM LESS THAN 5
YES
NO
SET ERROR PARAMETER IER=-1
EXIT
IS STEPSIZE H EQUAL TO 0
YES
NO
SET ERROR PARAMETER IER=1
EXIT
PREPARE DIFFERENTIA-TION LOOP
SET ERROR PARAMETER IER=0
COMPUTE B AND C BY MEANS OF FORMULAE (3) AND (4)
COMPUTE A BY MEANS OF FORMULA (6)
SAVE Y(NDIM-4) IN YY AND INITIALIZE I=4
COMPUTE RESULT Z(NDIM) BY MEANS OF FORMULA (7)
ADD 1 TO STEP-COUNTER I AND SAVE B IN A AND C IN B
SET RESULT Z(NDIM-1)=A
COMPUTE C BY MEANS OF FORMULA (5)
SET RESULT Z(NDIM-2)=C
SET RESULT Z(I-4)=A
SET RESULT Z(NDIM-3)=B
IS I EQUAL TO NDIM
NO
YES
EXIT
A6
B3

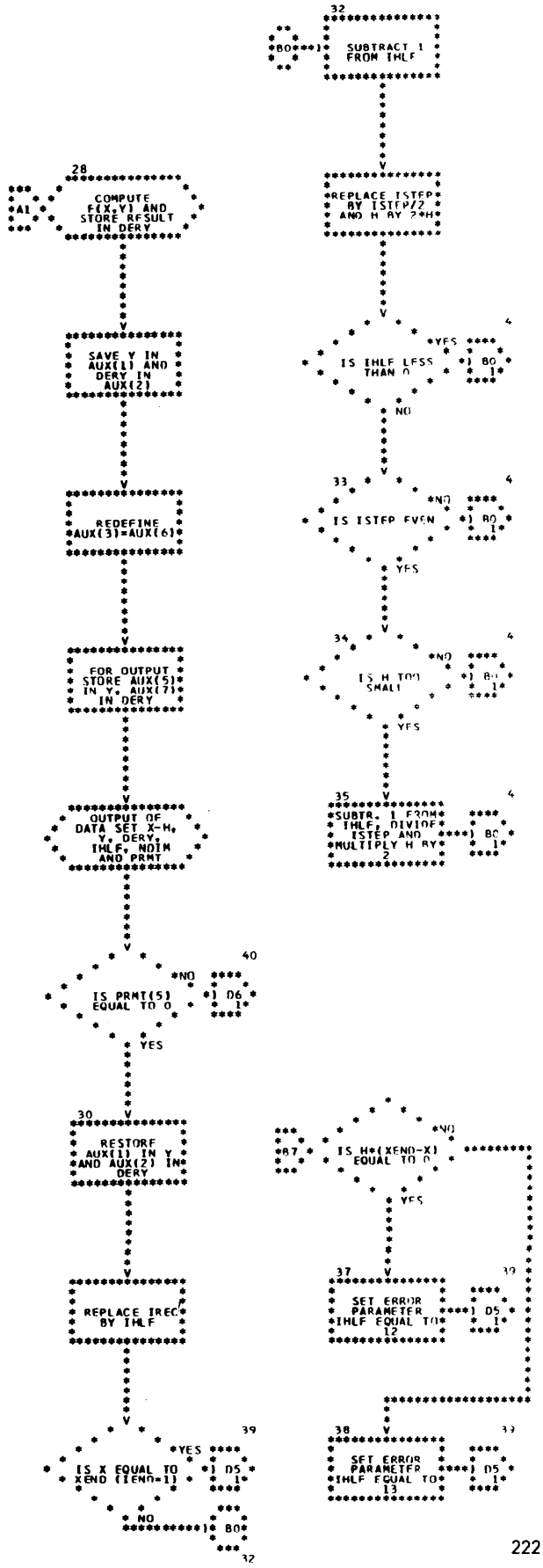

ENTER
IS RADIUS H EQUAL TO 0
YES
EXIT
NO
IS DECISION PARAMETER IN EQUAL TO 0
YES
SET STEPSIZE HH EQUAL TO ABSOLUTE VALUE OF H
10
A4
NO
2
GENERATE STEPSIZE HH FOR FIRST DIVIDED DIFFERENCE
10
A4
SET RESULT Z= EQUAL TO FIRST DIVIDED DIFFERENCE
B4
SET D2 EQUAL TO GREATEST MACHINE-PROCESSIBLE NUMBER
SET J=MAXIMUM NUMBER OF EXTRAPOLATION STEPS (I.E. 5)
INITIALIZE PARAMETERS USED IN EXTRAPOLATION LOOP
INITIALIZE JJ=J-1 AND STORE Z IN AUX(J)
INITIALIZE EXTRAPOLATION COUNTER I=J
SET DZ EQUAL TO GREATEST MACHINE-PROCESSIBLE NUMBER
B7
REPLACE D1 BY D2 AND UPDATE EXTRAPOLATION PARAMETER
11
A8
SUBTRACT 1 FROM J
STORE EXTRAPOLATED VALUE IN AUX(I+1)
STORE NEXT DIVIDED DIFFERENCE IN AUX(J)
B4
UPDATE INCREMENT D2
C0
C0
IS D2 LESS THAN D1
NO
YES
ADD 1 TO EXTRAPOLATION COUNTER I
12
IS I GREATER THAN JJ
NO
B7
YES
SET I=JJ+1
13
REPLACE D2 BY D1 AND JJ BY I
14
IS D2 LESS THAN DZ
NO
YES
15
UPDATE DZ=D2 AND SET RESULT Z=AUX(I)
16
IS J EQUAL TO 1
NO
11
A8
YES
17
EXIT

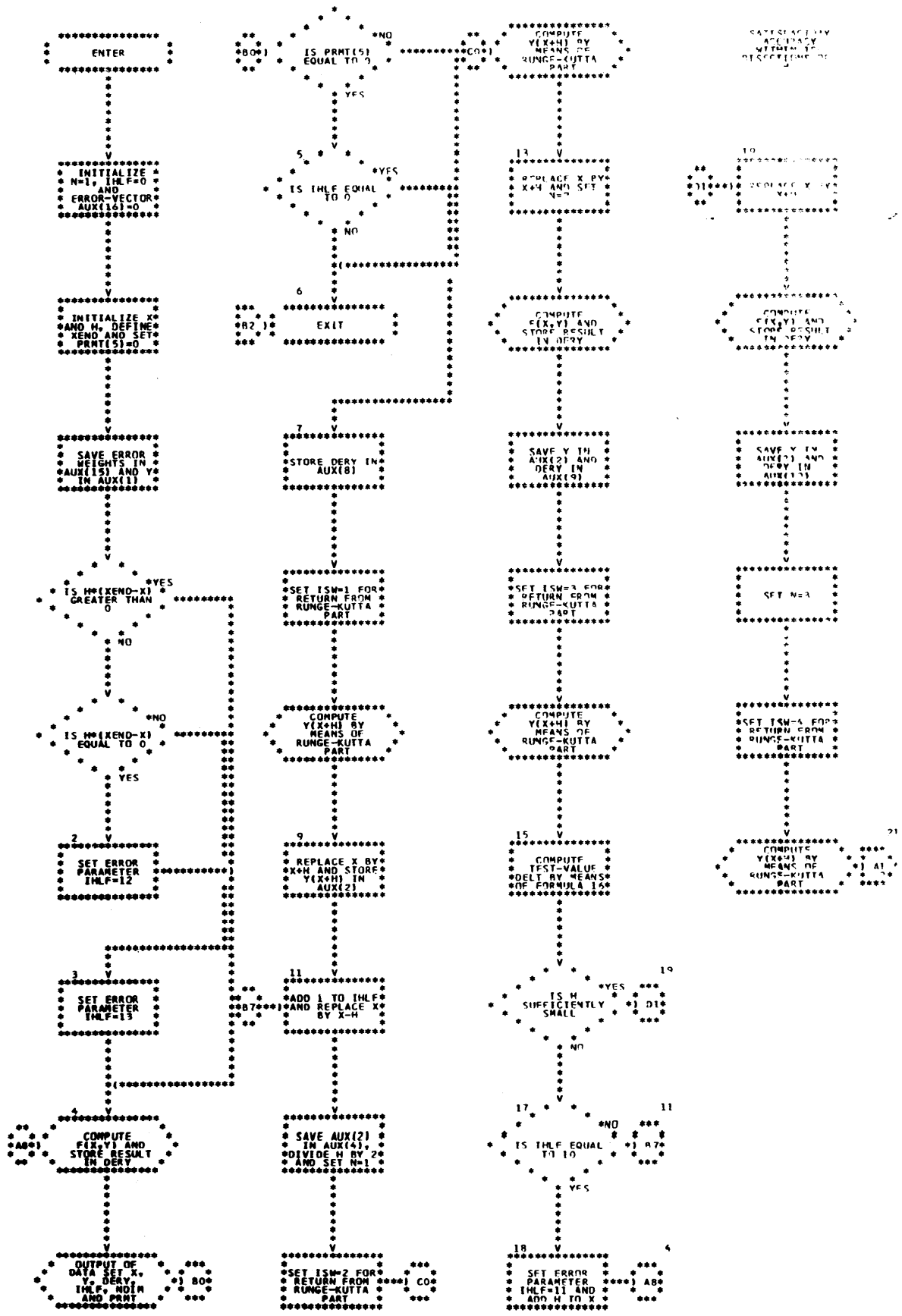

ENTER
IS H EQUAL TO 0
YES
EXIT
NO
IS DECISION PARAMETER IH EQUAL TO 0
YES
SET STEPSIZE HH EQUAL TO H
A4
NO
2
GENERATE STEPSIZE HH FOR FIRST DIVIDED DIFFERENCE
SET RESULT Z= EQUAL TO FIRST DIVIDED DIFFERENCE
A4
SET D2 EQUAL TO GREATEST MACHINE-PROCESSIBLE NUMBER
B4
SET J=MAXIMUM NUMBER OF EXTRAPOLATION STEPS (I.E.,I^)
INITIALIZE JJ=J-1 AND STORE Z IN AUX(J)
INITIALIZE PARAMETERS USED IN EXTRAPOLATION LOOP
INITIALIZE EXTRAPOLATION COUNTER I=J
SET D2 EQUAL TO GREATEST MACHINE-PROCESSIBLE NUMBER
B7
REPLACE D1 BY D2 AND UPDATE EXTRAPOLATION PARAMETER
11
SUBTRACT 1 FROM J
A8
STORE NEXT DIVIDED DIFFERENCE IN AUX(J)
B4
STORE EXTRAPOLATED VALUE IN AUX(I+1)
UPDATE INCREMENT D2
C0
C0
IS D2 LESS THAN D1
NO
YES
ADD 1 TO EXTRAPOLATION COUNTER I
12
IS I GREATER THAN JJ
NO
B7
YES
SET I=JJ+1
13
REPLACE D2 BY D1 AND JJ BY I
14
IS D2 LESS THAN D2
NO
YES
15
UPDATE D2=D2 AND SET RESULT Z=AUX(I)
16
IS J EQUAL TO 1
NO
A8
11
YES
17
EXIT

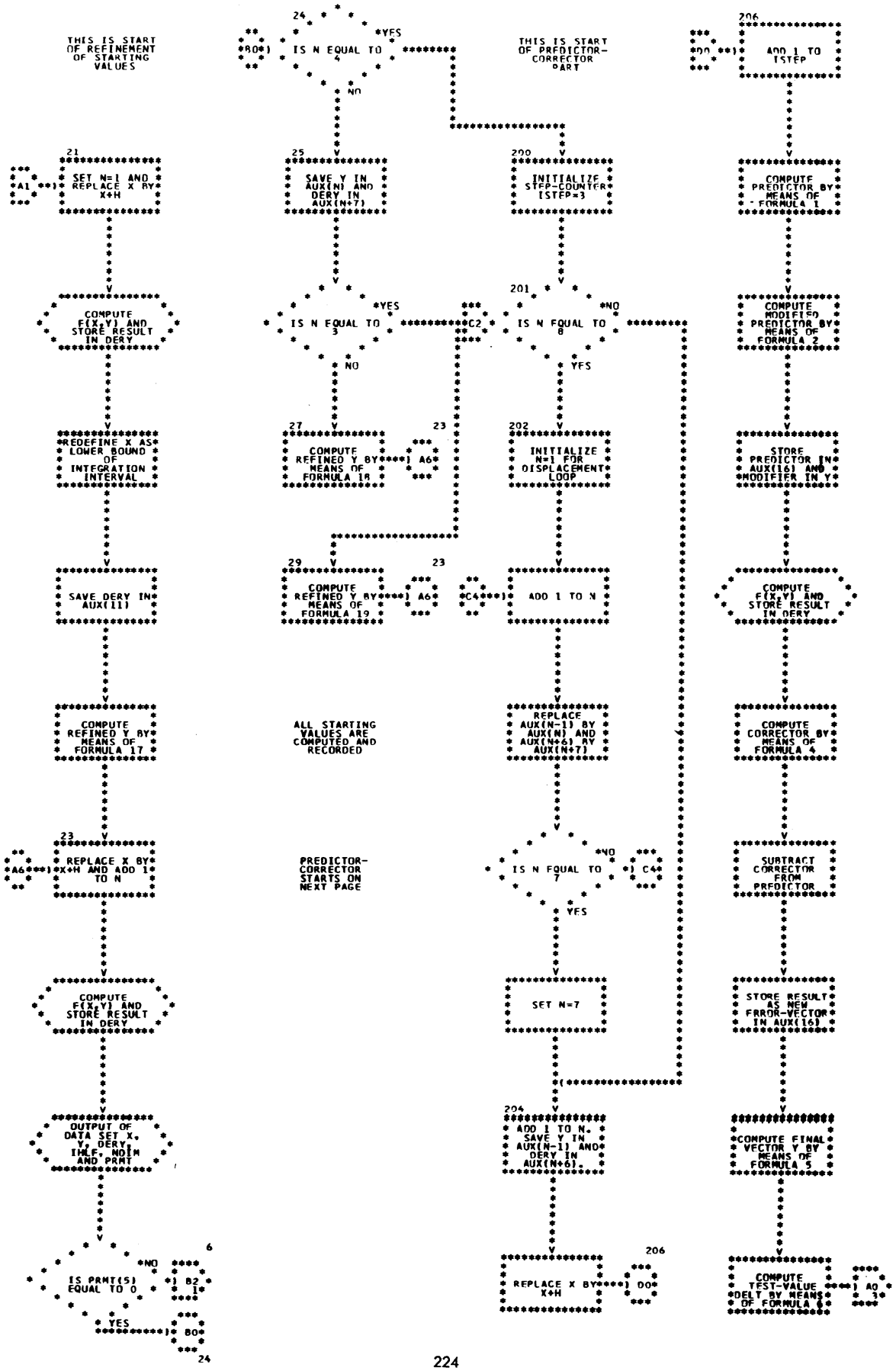

ENTER
X(FINAL) LESS THAN OR EQUAL TO X(INITIAL)
NO
YES
11
X(ANSWER) = X(INITIAL), Y(ANSWER) = Y(INITIAL)
EXIT
IS STEP SIZE ZERO
NO
YES
14
SET ERROR CODE TO 1
X(ANSWER) = X(INITIAL), Y(ANSWER) = 0*
EXIT
20
XN = X(INITIAL), YN = Y(INITIAL)
INTEGRATE YN1 = Y(XN)
B0*
XN1 = X(FINAL)
NO
YES
30
SET X(ANSWER) = X(FINAL), Y(ANSWER) = YN1
C9*
XN1 LESS THAN X(FINAL)
YES
NO
40
SET H(NEW) = X(FINAL)-X(N)
INTEGRATE Y(ANS) = Y(XN+H)
C9*
50
DO YN1 AND YN SPAN Y(FINAL)
YES
NO
60
YN = YN1, XN = XN1
INTEGRATE YN1 = Y(XN)
70
DOES YN = Y(FINAL) OR YN1 = Y(FINAL)
YES*
8
C0*
NO
110
I = 1
C3*
IS THIS YN
YES
NO
100
X(ANSWER) = XN1, Y(ANSWER) = YN1
80
X(ANSWER) = XN, Y(ANSWER) = YN
C3*
INTERPOLATE TO FIND NEW TIME STEP
INTEGRATE YN1 = Y(XN+H)
YN1 = Y(FINAL)
YES*
8
NO
120
I = 10
150
YES*
8
C8*
NO
I = I+1
C3*
150
C8*
X(ANSWER) = XN1, Y(ANSWER) = Y(FINAL)
C9*
EXIT

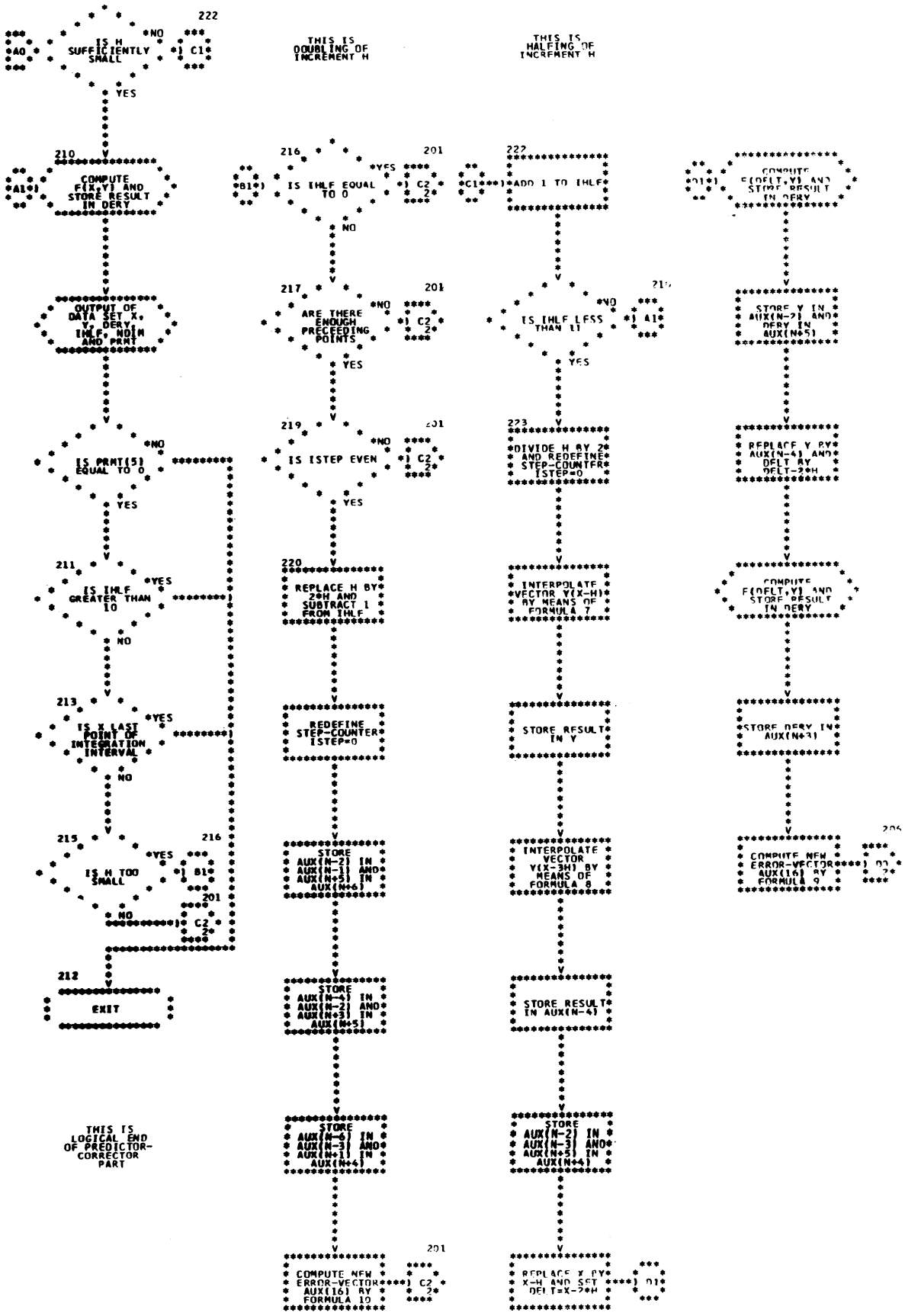

ENTER
X = X(INITIAL), Y = Y(INITIAL)
I = 1
J = 1
A3
INTEGRATE Y = Y(X&H) USING EC.(2),(3)
A4
X = X&H
J = K
YES
NO
B0
J = J&1
A4
2
B0
VECTOR(I) = Y
I = N
NO
YES
EXIT
I = I&1
A3

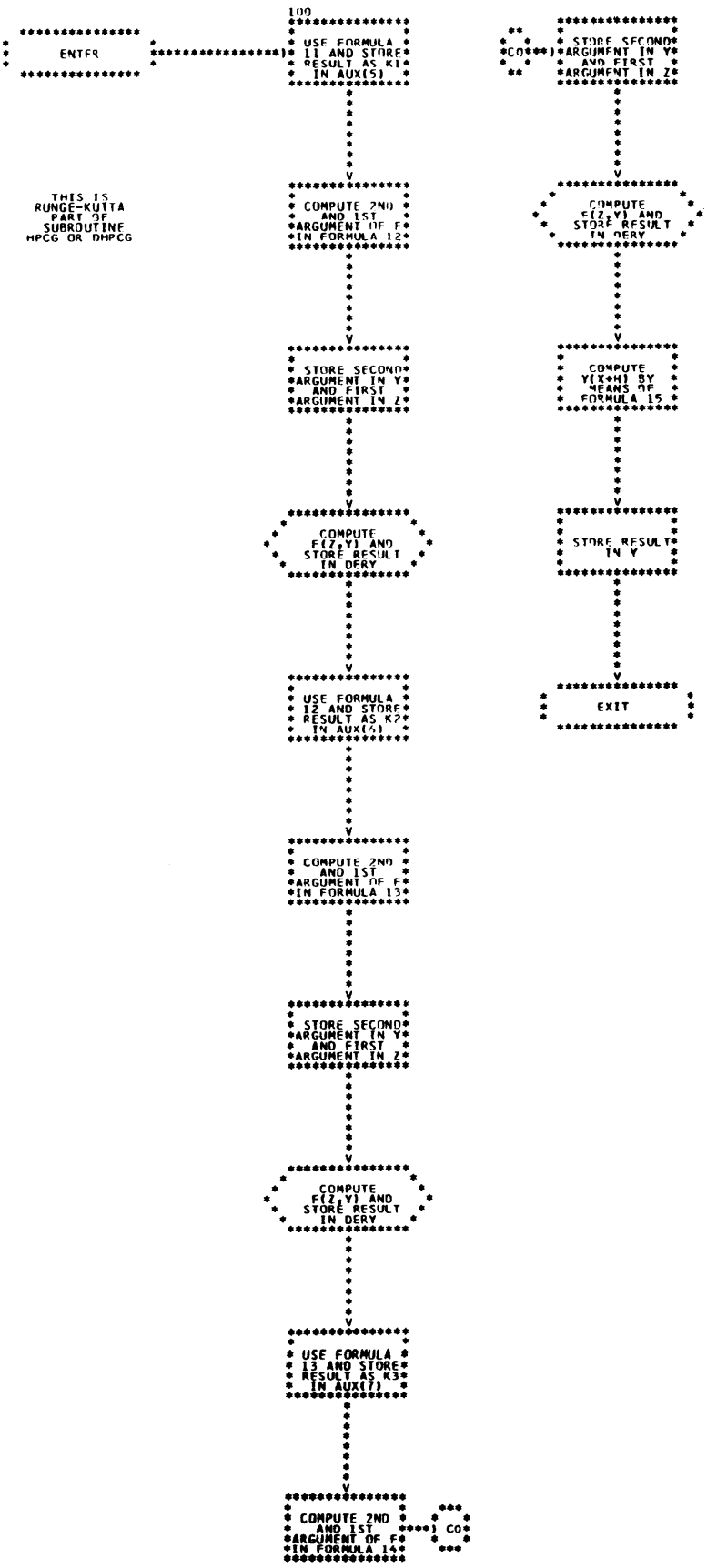

ENTER
MULTIPLY FRR-OR WEIGHTS BY 1/15, STORE IN AUX(8)
INITIALIZE X AND H, DEFINE XEND AND SET PRMT(5)=0
COMPUTE F(X,Y) AND STORE RESULT IN DERY
IS H*(XEND-X) GREATER THAN 0
YES
NO
PREPARE COEFFICIENTS USED IN FORMULAE 1
SAVE Y IN AUX(1) AND DERY IN AUX(2)
INITIALIZE AUX(3)=0 AND AUX(6)=0
REPLACE H BY 2*H AND INITIALIZE IHLF=-1
INITIALIZE ISTEP=0, IEND=0 AND IREC=0
IS X+H BETWEEN X AND XEND
REPLACE H BY XEND-X AND SET IEND=1
OUTPUT OF DATA SET X, Y, DERY, IREC, NDIM AND PRMT
IS PRMT(5) EQUAL TO 0
SET TEST PARAMETER ITEST=0
ADD 1 TO ISTEP
INITIALIZE J=1 FOR INNERMOST RUNGE-KUTTA LOOP
COMPUTE Y AND AUX(6) (USE ROW J OF FORMULAE 1)
IS J EQUAL TO 4
ADD 1 TO J
IS J EQUAL TO 3
REPLACE X BY X+H/2
COMPUTE F(X,Y) AND STORE RESULT IN DERY
IS TEST OF ACCURACY POSSIBLE
SAVE Y IN AUX(4), SET ITEST=1 AND ISTEP= 2*(ISTEP-1)
ADD 1 TO IHLF AND REPLACE X BY X-H AND H BY H/2
REDEFINE Y=AUX(1), DERY=AUX(2) AUX(6)=AUX(3)
IS ISTEP EVEN
COMPUTE F(X,Y) AND STORE RESULT IN DERY
SAVE Y IN AUX(5) AND DERY IN AUX(7)
COMPUTE TEST VALUE DELT BY MEANS OF FORMULA 2
IS H SUFFICIENTLY SMALL
IS IHLF EQUAL TO 10
SET ERROR PARAMETER IHLF EQUAL TO 11
COMPUTE F(X,Y) AND STORE RESULT IN DERY
OUTPUT OF DATA SET X, Y, DERY, IHLF, NDIM AND PRMT
EXIT
REDEFINE AUX(4)=AUX(5) AND X=X-H
REPLACE ISTEP BY 2*(ISTEP-2) AND SET IEND=2
YES
NO

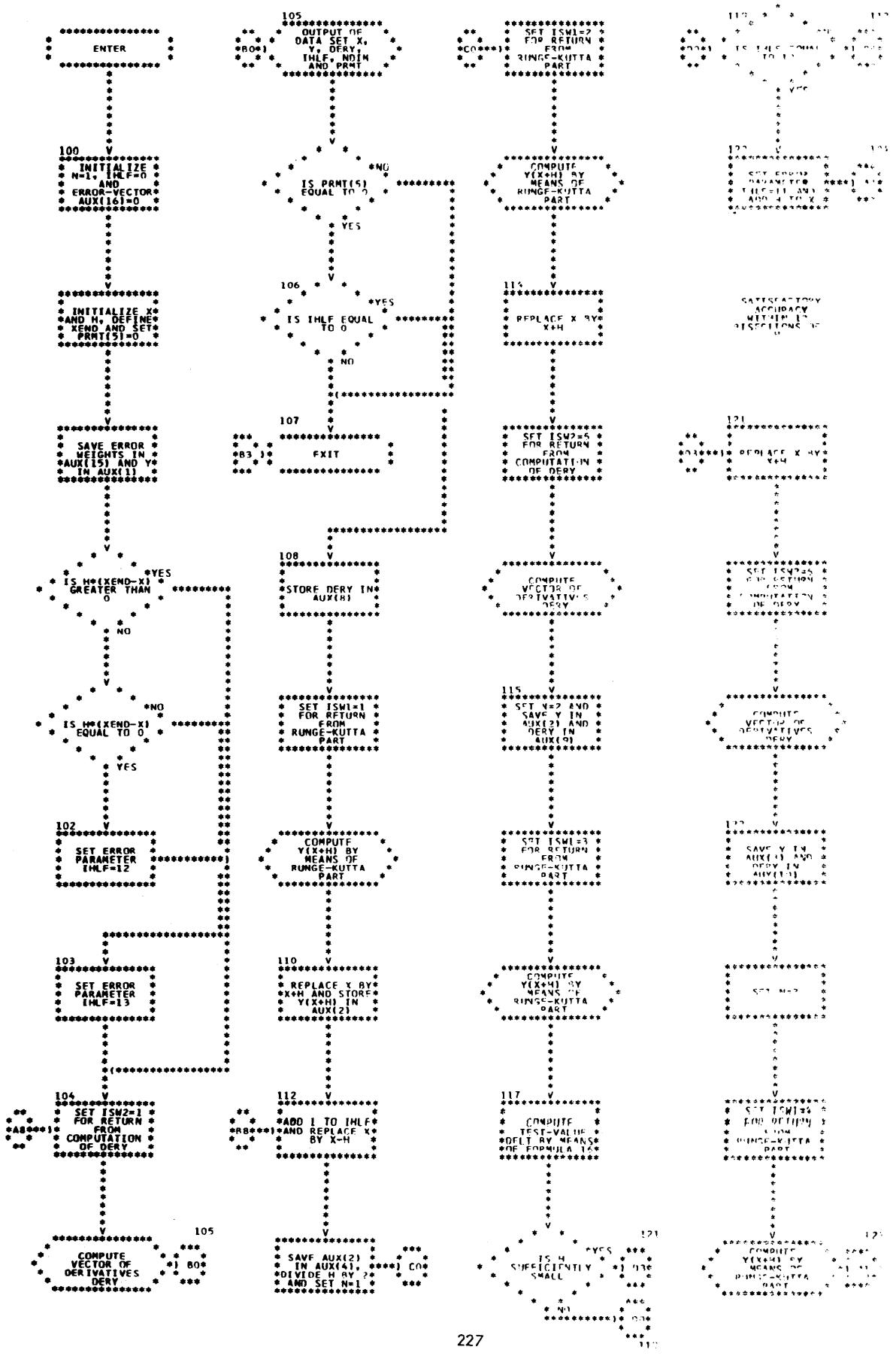

32
SUBTRACT 1 FROM IHLF
B0
28
COMPUTE F(X,Y) AND STORE RESULT IN DERY
A1
REPLACE ISTEP BY ISTEP/2 AND H BY 2*H
SAVE Y IN AUX(1) AND DERY IN AUX(2)
IS IHLF LESS THAN 0
YES
NO
4
B0
1
REDEFINE AUX(3)=AUX(6)
33
IS ISTEP EVEN
FOR OUTPUT STORE AUX(5) IN Y, AUX(7) IN DERY
34
IS H TOO SMALL
OUTPUT OF DATA SET X-H, Y, DERY, IHLF, NDIM AND PRMT
35
SUBTR. 1 FROM IHLF, DIVIDE ISTEP AND MULTIPLY H BY 2
IS PRMT(5) EQUAL TO 0
40
D6
30
RESTORE AUX(1) IN Y AND AUX(2) IN DERY
REPLACE IREC BY IHLF
IS X EQUAL TO XEND (IEND=1)
39
D5
B7
IS H*(XEND-X) EQUAL TO 0
37
SET ERROR PARAMETER IHLF EQUAL TO 12
38
SET ERROR PARAMETER IHLF EQUAL TO 13

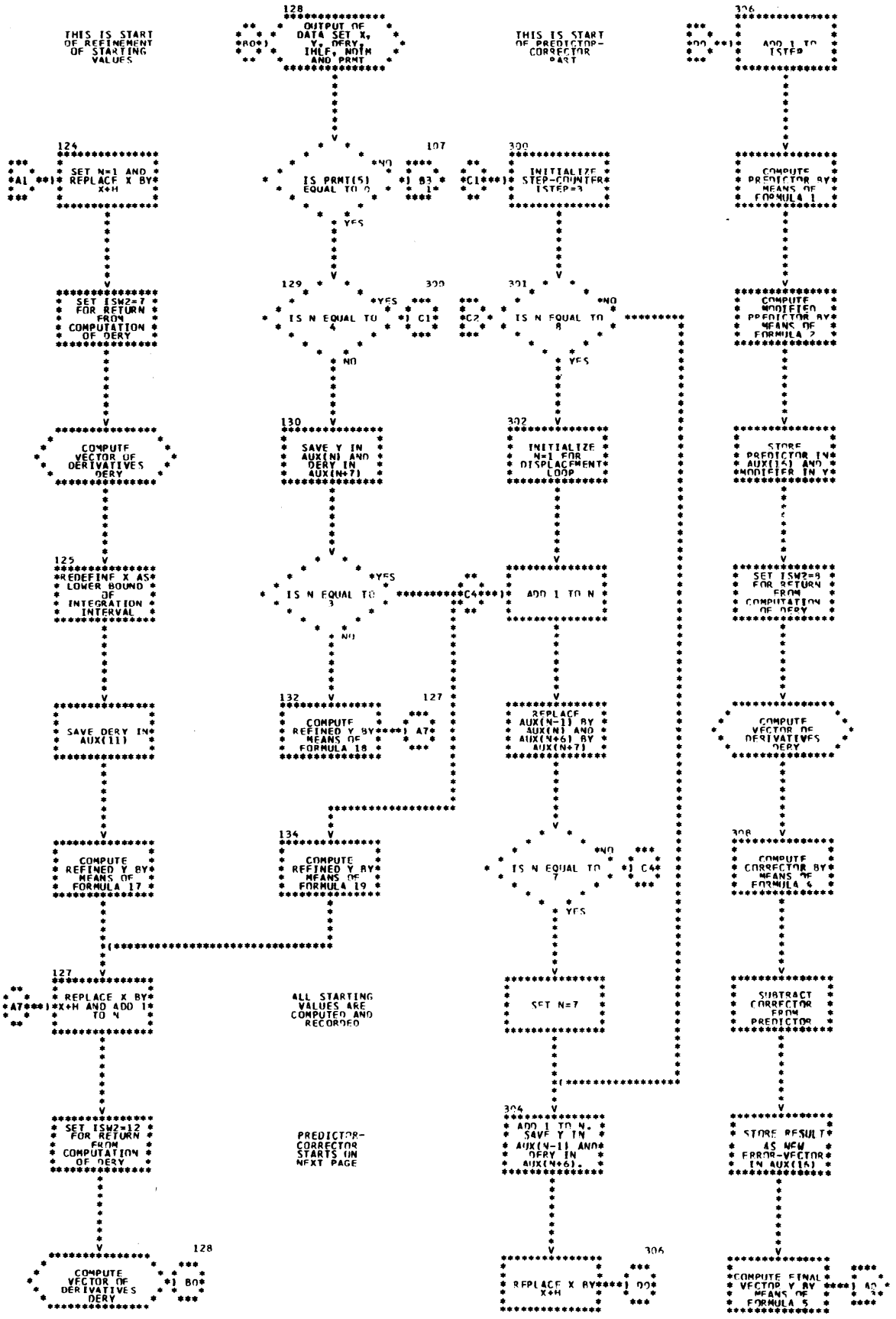

ENTER
INITIALIZE N=1, IHLF=0 AND ERROR-VECTOR AUX(16)=0
INITIALIZE X AND H, DEFINE XEND AND SET PRNT(5)=0
SAVE ERROR WEIGHTS IN AUX(15) AND Y IN AUX(1)
IS H*(XEND-X) GREATER THAN 0
IS H*(XEND-X) EQUAL TO 0
SET ERROR PARAMETER IHLF=12
SET ERROR PARAMETER IHLF=13
COMPUTE F(X,Y) AND STORE RESULT IN DERY
OUTPUT OF DATA SET X, Y, DERY, IHLF, NDIM AND PRNT
IS PRNT(5) EQUAL TO 0
IS IHLF EQUAL TO 0
EXIT
STORE DERY IN AUX(8)
SET ISW=1 FOR RETURN FROM RUNGE-KUTTA PART
COMPUTE Y(X+H) BY MEANS OF RUNGE-KUTTA PART
REPLACE X BY X+H AND STORE Y(X+H) IN AUX(2)
ADD 1 TO IHLF AND REPLACE X BY X-H
SAVE AUX(2) IN AUX(4), DIVIDE H BY 2 AND SET N=1
SET ISW=2 FOR RETURN FROM RUNGE-KUTTA PART
COMPUTE Y(X+H) BY MEANS OF RUNGE-KUTTA PART
REPLACE X BY X+H AND SET N=2
SAVE Y IN AUX(2) AND DERY IN AUX(9)
SET ISW=3 FOR RETURN FROM RUNGE-KUTTA PART
COMPUTE Y(X+H) BY MEANS OF RUNGE-KUTTA PART
COMPUTE TEST-VALUE DELT BY MEANS OF FORMULA 16
IS H SUFFICIENTLY SMALL
IS IHLF EQUAL TO 10
SET ERROR PARAMETER IHLF=11 AND ADD H TO X
SATISFACTORY ACCURACY WITHOUT TO RESTORE OLD
REPLACE X BY X+H
COMPUTE F(X,Y) AND STORE RESULT IN DERY
SAVE Y IN AUX(2) AND DERY IN AUX(11)
SET N=3
SET ISW=4 FOR RETURN FROM RUNGE-KUTTA PART
COMPUTE Y(X+H) BY MEANS OF RUNGE-KUTTA PART
YES
NO

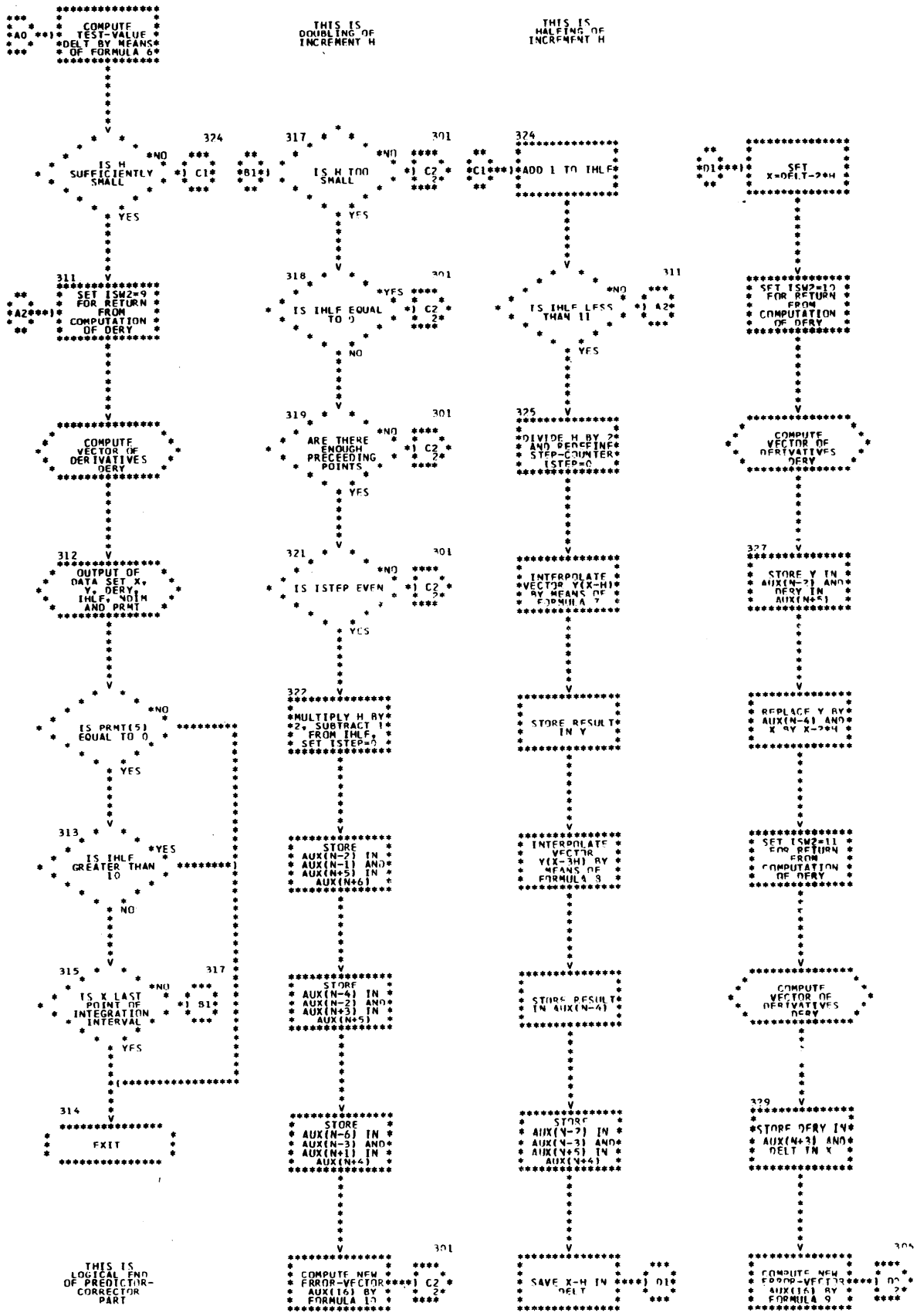

THIS IS START OF REFINEMENT OF STARTING VALUES
21
SET N=1 AND REPLACE X BY X+H
A1
COMPUTE F(X,Y) AND STORE RESULT IN DERY
REDEFINE X AS LOWER BOUND OF INTEGRATION INTERVAL
SAVE DERY IN AUX(11)
COMPUTE REFINED Y BY MEANS OF FORMULA 17
23
REPLACE X BY X+H AND ADD 1 TO N
A6
COMPUTE F(X,Y) AND STORE RESULT IN DERY
OUTPUT OF DATA SET X, Y, DERY, IHLF, NOIM AND PRMT
IS PRMT(5) EQUAL TO 0
NO
B2
I
YES
B0
6
24
IS N EQUAL TO 4
YES
B0
NO
25
SAVE Y IN AUX(N) AND DERY IN AUX(N+7)
IS N EQUAL TO 3
YES
C2
NO
27
COMPUTE REFINED Y BY MEANS OF FORMULA 18
23
A6
29
COMPUTE REFINED Y BY MEANS OF FORMULA 19
23
A6
ALL STARTING VALUES ARE COMPUTED AND RECORDED
PREDICTOR-CORRECTOR STARTS ON NEXT PAGE
THIS IS START OF PREDICTOR-CORRECTOR PART
200
INITIALIZE STEP-COUNTER ISTEP=3
201
IS N EQUAL TO 8
NO
YES
202
INITIALIZE N=1 FOR DISPLACEMENT LOOP
C2
ADD 1 TO N
REPLACE AUX(N-1) BY AUX(N) AND AUX(N+6) BY AUX(N+7)
IS N EQUAL TO 7
NO
C4
YES
SET N=7
204
ADD 1 TO N, SAVE Y IN AUX(N-1) AND DERY IN AUX(N+6).
206
REPLACE X BY X+H
D0
206
ADD 1 TO ISTEP
COMPUTE PREDICTOR BY MEANS OF FORMULA 1
COMPUTE MODIFIER PREDICTOR BY MEANS OF FORMULA 2
STORE PREDICTOR IN AUX(16) AND MODIFIER IN Y
COMPUTE F(X,Y) AND STORE RESULT IN DERY
COMPUTE CORRECTOR BY MEANS OF FORMULA 4
SUBTRACT CORRECTOR FROM PREDICTOR
STORE RESULT AS NEW ERROR-VECTOR IN AUX(16)
COMPUTE FINAL VECTOR Y BY MEANS OF FORMULA 5
COMPUTE TEST-VALUE DELT BY MEANS OF FORMULA 6
A0
3

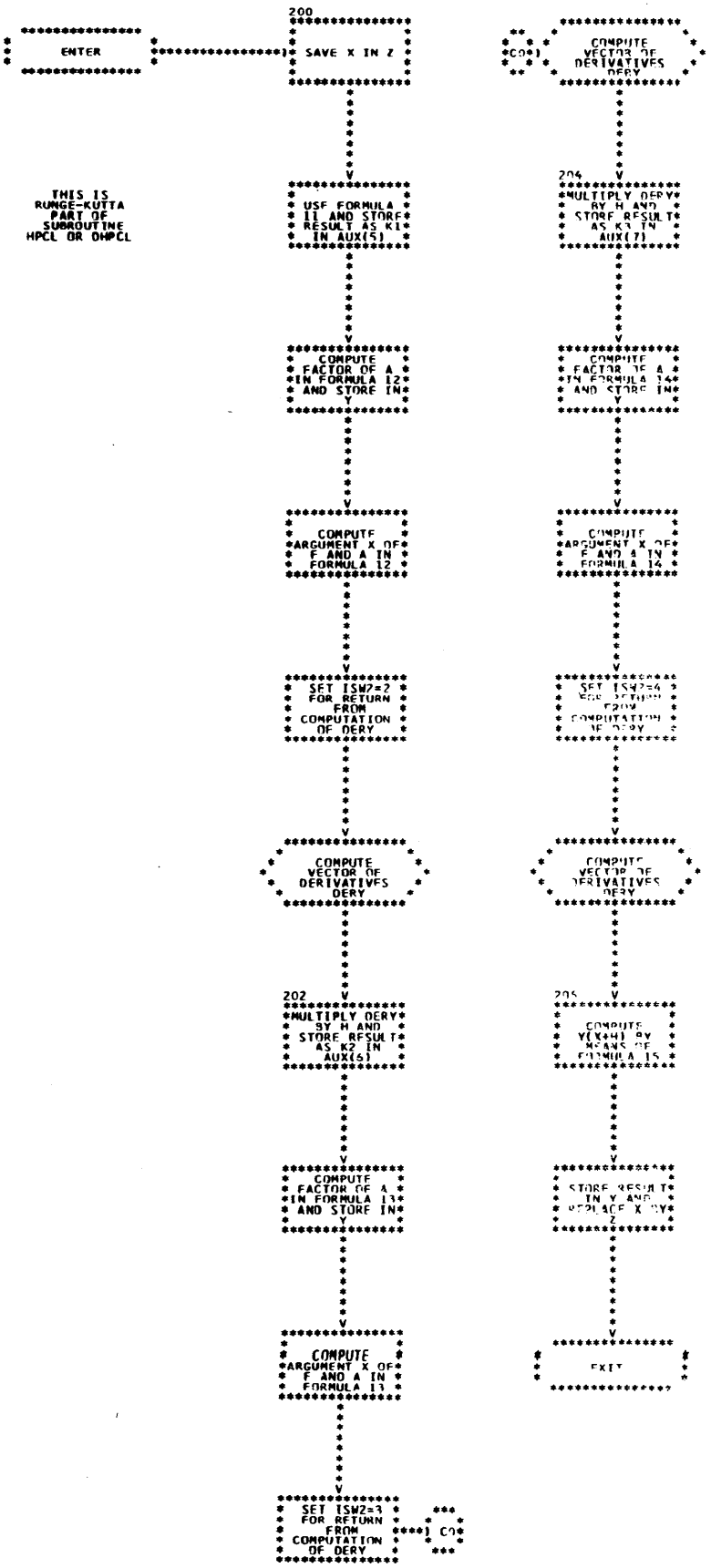

IS H SUFFICIENTLY SMALL
NO
C1
222
AO
YES
210
COMPUTE F(X,Y) AND STORE RESULT IN DERY
A1
OUTPUT OF DATA SET X, Y, DERY, IHLF, NDIM AND PRNT
IS PRNT(5) EQUAL TO 0
NO
YES
211
IS IHLF GREATER THAN 10
YES
NO
213
IS X LAST POINT OF INTEGRATION INTERVAL
YES
NO
215
IS H TOO SMALL
YES
B1
216
201
C2
2
NO
212
EXIT
THIS IS LOGICAL END OF PREDICTOR-CORRECTOR PART
THIS IS DOUBLING OF INCREMENT H
216
IS IHLF EQUAL TO 0
YES
201
C2
2
B1
NO
217
ARE THERE ENOUGH PRECEEDING POINTS
NO
201
C2
2
YES
219
IS ISTEP EVEN
NO
201
C2
2
YES
220
REPLACE H BY 2*H AND SUBTRACT 1 FROM IHLF
REDEFINE STEP-COUNTER ISTEP=0
STORE AUX(N-2) IN AUX(N-1) AND AUX(N+5) IN AUX(N+6)
STORE AUX(N-4) IN AUX(N-2) AND AUX(N+3) IN AUX(N+5)
STORE AUX(N-6) IN AUX(N-3) AND AUX(N+1) IN AUX(N+4)
201
COMPUTE NEW ERROR-VECTOR AUX(16) BY FORMULA 10
C2
2
THIS IS HALFING OF INCREMENT H
222
C1
ADD 1 TO IHLF
IS IHLF LESS THAN 11
NO
A1
YES
223
DIVIDE H BY 2 AND REDEFINE STEP-COUNTER ISTEP=0
INTERPOLATE VECTOR Y(X-H) BY MEANS OF FORMULA 7
STORE RESULT IN Y
INTERPOLATE VECTOR Y(X-3H) BY MEANS OF FORMULA 8
STORE RESULT IN AUX(N-4)
STORE AUX(N-2) IN AUX(N-3) AND AUX(N+5) IN AUX(N+4)
REPLACE X BY X-H AND SET DELT=X-2*H
B1
NO
B1
COMPUTE F(DELT,Y) AND STORE RESULT IN DERY
STORE Y IN AUX(N-2) AND DERY IN AUX(N+5)
REPLACE Y BY AUX(N-4) AND DELT BY DELT-2*H
COMPUTE F(DELT,Y) AND STORE RESULT IN DERY
STORE DERY IN AUX(N+3)
COMPUTE NEW ERROR-VECTOR AUX(16) BY FORMULA 9
206
B1
2

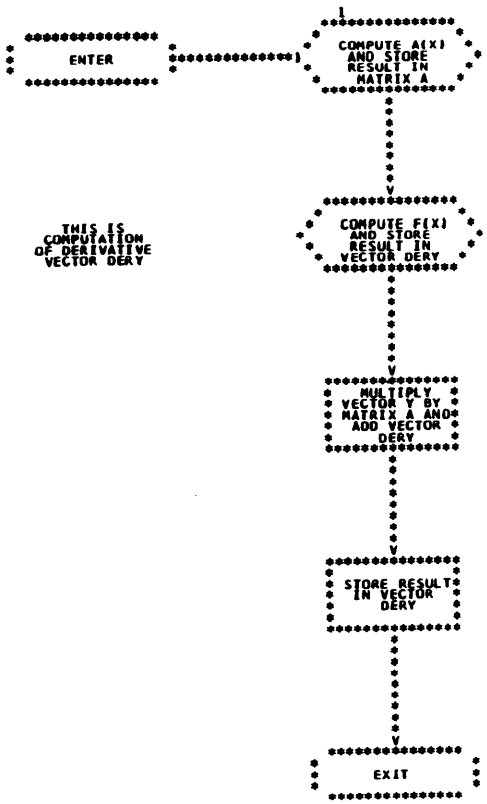

100
ENTER
USE FORMULA 11 AND STORE RESULT AS K1 IN AUX(5)
COMPUTE 2ND AND 1ST ARGUMENT OF F IN FORMULA 12
STORE SECOND ARGUMENT IN Y AND FIRST ARGUMENT IN Z
COMPUTE F(Z,Y) AND STORE RESULT IN DERY
USE FORMULA 12 AND STORE RESULT AS K2 IN AUX(6)
COMPUTE 2ND AND 1ST ARGUMENT OF F IN FORMULA 13
STORE SECOND ARGUMENT IN Y AND FIRST ARGUMENT IN Z
COMPUTE F(Z,Y) AND STORE RESULT IN DERY
USE FORMULA 13 AND STORE RESULT AS K3 IN AUX(7)
COMPUTE 2ND AND 1ST ARGUMENT OF F IN FORMULA 14
CO
STORE SECOND ARGUMENT IN Y AND FIRST ARGUMENT IN Z
COMPUTE F(Z,Y) AND STORE RESULT IN DERY
COMPUTE Y(X+H) BY MEANS OF FORMULA 15
STORE RESULT IN Y
EXIT
THIS IS RUNGE-KUTTA PART OF SUBROUTINE HPCG OR DHPCG

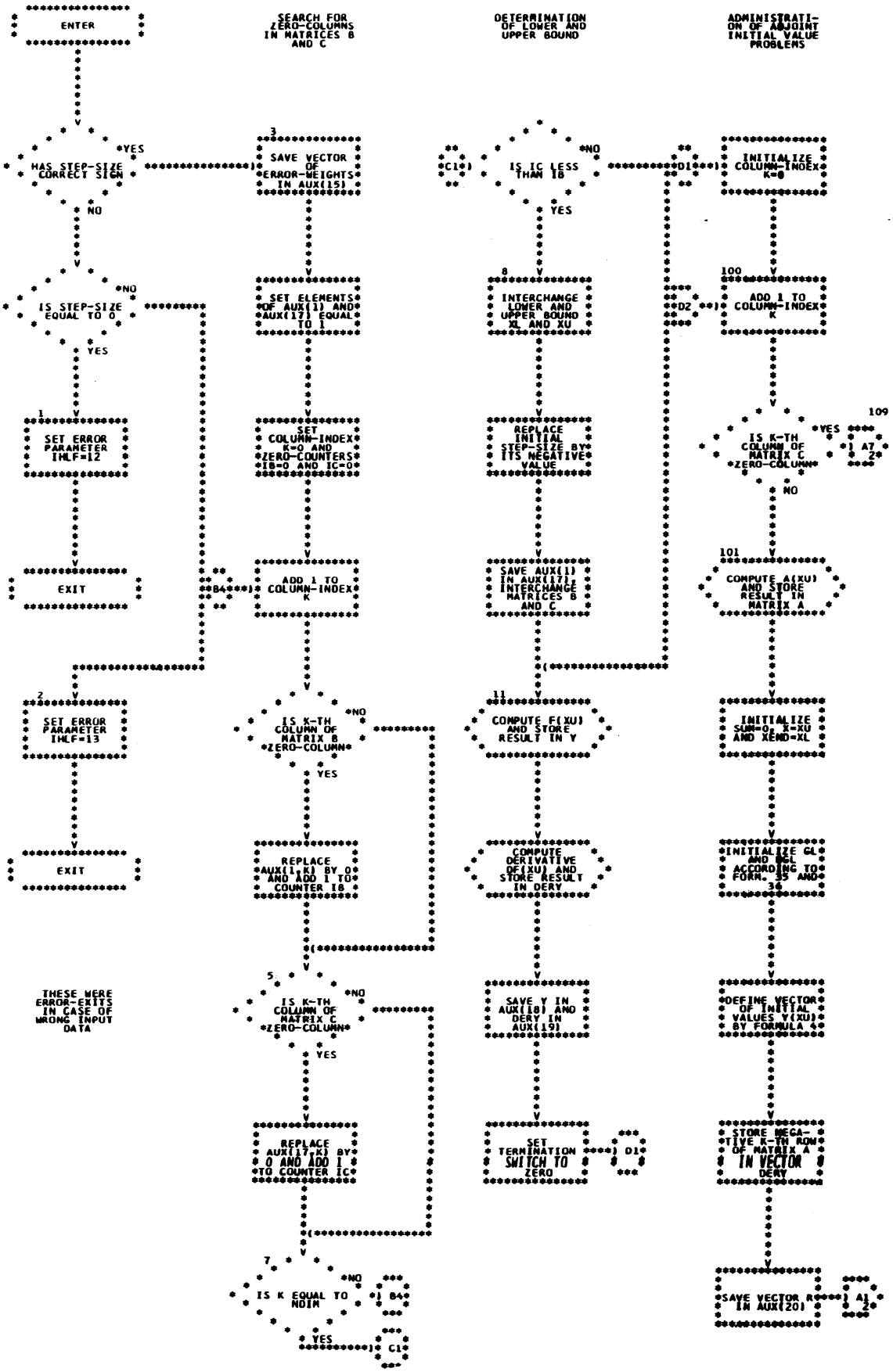

ENTER
100
INITIALIZE N=1, IHLF=0 AND ERROR-VECTOR AUX(16)=0
INITIALIZE X AND H, DEFINE XEND AND SET PRNT(5)=0
SAVE ERROR WEIGHTS IN AUX(15) AND Y IN AUX(1)
IS H*(XEND-X) GREATER THAN 0
YES
NO
IS H*(XEND-X) EQUAL TO 0
YES
NO
102
SET ERROR PARAMETER IHLF=12
103
SET ERROR PARAMETER IHLF=13
104
SET ISW2=1 FOR RETURN FROM COMPUTATION OF DERY
105
COMPUTE VECTOR OF DERIVATIVES DERY
105
OUTPUT OF DATA SET X, Y, DERY, IHLF, NDIM AND PRNT
IS PRNT(5) EQUAL TO 0
NO
YES
106
IS IHLF EQUAL TO 0
YES
NO
107
EXIT
108
STORE DERY IN AUX(8)
SET ISW1=1 FOR RETURN FROM RUNGE-KUTTA PART
COMPUTE Y(X+H) BY MEANS OF RUNGE-KUTTA PART
110
REPLACE X BY X+H AND STORE Y(X+H) IN AUX(2)
112
ADD 1 TO IHLF AND REPLACE X BY X-H
SAVE AUX(2) IN AUX(4), DIVIDE H BY 2 AND SET N=1
SET ISW1=2 FOR RETURN FROM RUNGE-KUTTA PART
COMPUTE Y(X+H) BY MEANS OF RUNGE-KUTTA PART
114
REPLACE X BY X+H
SET ISW2=5 FOR RETURN FROM COMPUTATION OF DERY
COMPUTE VECTOR OF DERIVATIVES DERY
115
SET N=2 AND SAVE Y IN AUX(2) AND DERY IN AUX(9)
SET ISW1=3 FOR RETURN FROM RUNGE-KUTTA PART
COMPUTE Y(X+H) BY MEANS OF RUNGE-KUTTA PART
117
COMPUTE TEST-VALUE DELT BY MEANS OF FORMULA 14
121
IS H SUFFICIENTLY SMALL
YES
NO
SATISFACTORY ACCURACY WITHIN N SUBDIVISIONS OF H
121
REPLACE X BY X+H
COMPUTE VECTOR OF DERIVATIVES DERY
122
SAVE Y IN AUX(1) AND DERY IN AUX(13)
SET N=2
SET ISW1=4 FOR RETURN FROM RUNGE-KUTTA PART
COMPUTE Y(X+H) BY MEANS OF RUNGE-KUTTA PART

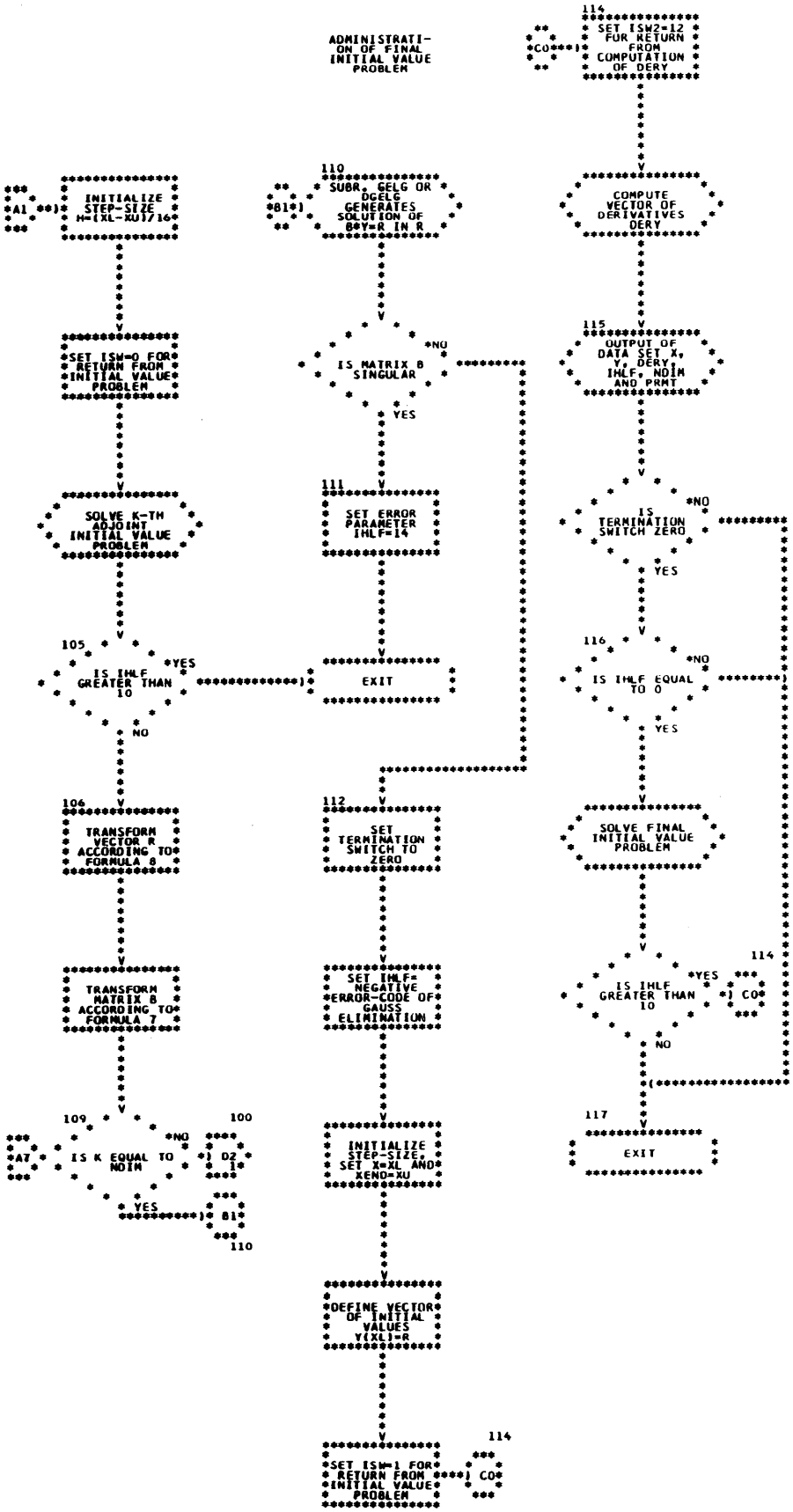

THIS IS START OF REFINEMENT OF STARTING VALUES
124
SET N=1 AND REPLACE X BY X+H
125
REDEFINE X AS LOWER BOUND OF INTEGRATION INTERVAL
SAVE DERY IN AUX(11)
COMPUTE REFINED Y BY MEANS OF FORMULA 17
127
REPLACE X BY X+H AND ADD 1 TO N
SET ISW2=12 FOR RETURN FROM COMPUTATION OF DERY
128
COMPUTE VECTOR OF DERIVATIVES DERY
128
OUTPUT OF DATA SET X, Y, DERY, IHLF, NOIM AND PRMT
IS PRMT(5) EQUAL TO 0
NO
YES
107
B3
129
IS N EQUAL TO 4
YES
300
C1
NO
130
SAVE Y IN AUX(N) AND DERY IN AUX(N+7)
IS N EQUAL TO 3
YES
C4
NO
132
COMPUTE REFINED Y BY MEANS OF FORMULA 18
127
A7
134
COMPUTE REFINED Y BY MEANS OF FORMULA 19
ALL STARTING VALUES ARE COMPUTED AND RECORDED
PREDICTOR-CORRECTOR STARTS ON NEXT PAGE
THIS IS START OF PREDICTOR-CORRECTOR PART
300
INITIALIZE STEP-COUNTER ISTEP=3
C1
301
IS N EQUAL TO 8
NO
YES
C2
302
INITIALIZE N=1 FOR DISPLACEMENT LOOP
ADD 1 TO N
C4
REPLACE AUX(N-1) BY AUX(N) AND AUX(N+6) BY AUX(N+7)
IS N EQUAL TO 7
NO
C4
YES
SET N=7
306
ADD 1 TO N, SAVE Y IN AUX(N-1) AND DERY IN AUX(N+6).
REPLACE X BY X+H
306
ADD 1 TO ISTEP
COMPUTE PREDICTOR BY MEANS OF FORMULA 1
COMPUTE MODIFIED PREDICTOR BY MEANS OF FORMULA 2
STORE PREDICTOR IN AUX(I4) AND MODIFIER IN Y
SET ISW2=8 FOR RETURN FROM COMPUTATION OF DERY
COMPUTE VECTOR OF DERIVATIVES DERY
308
COMPUTE CORRECTOR BY MEANS OF FORMULA 4
SUBTRACT CORRECTOR FROM PREDICTOR
STORE RESULT AS NEW ERROR-VECTOR IN AUX(I4)
COMPUTE FINAL Y BY MEANS OF FORMULA 5
A0

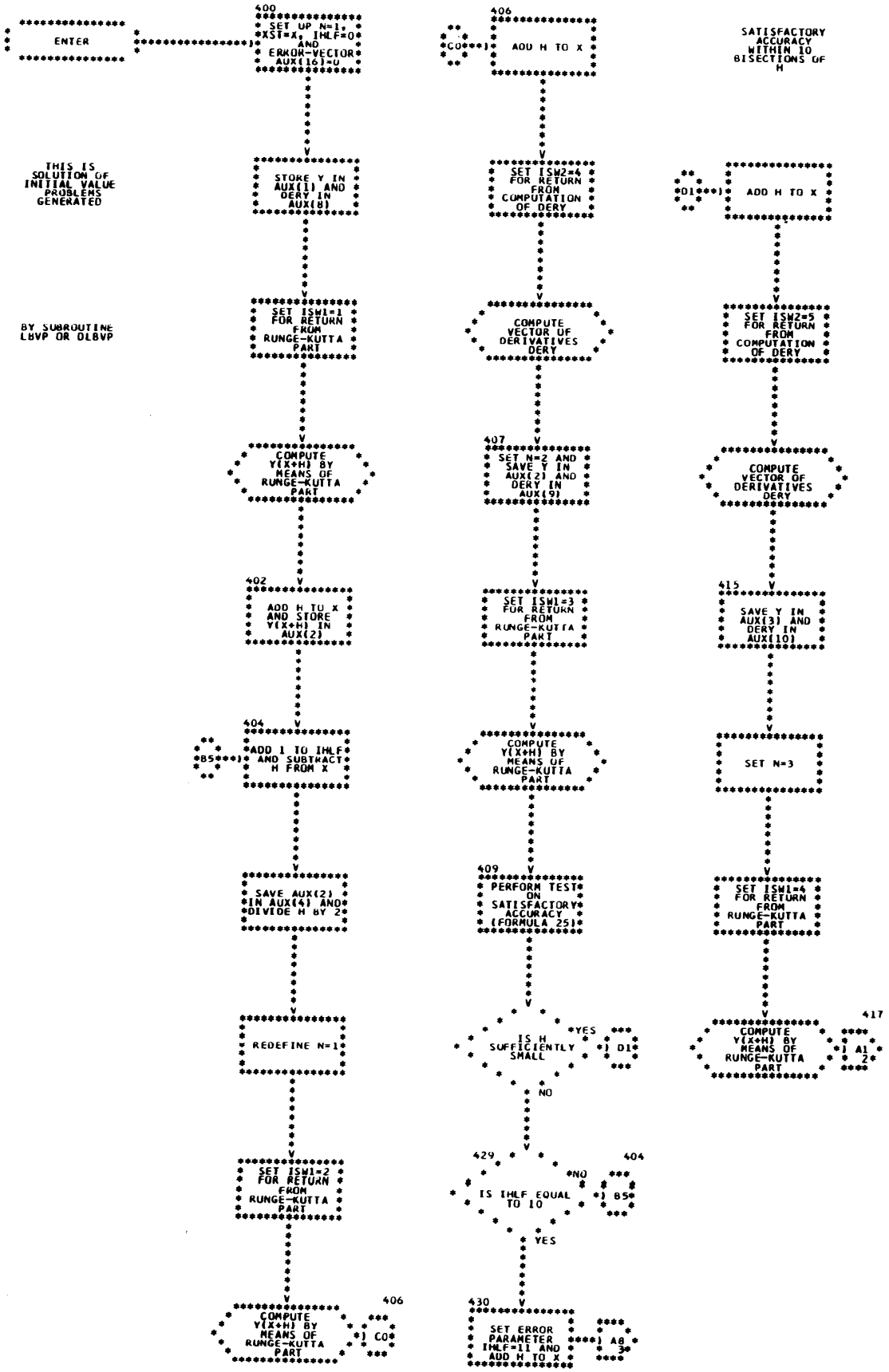

COMPUTE TEST-VALUE DELT BY MEANS OF FORMULA 6
IS H SUFFICIENTLY SMALL
NO
YES
SET ISW2=9 FOR RETURN FROM COMPUTATION OF DERY
COMPUTE VECTOR OF DERIVATIVES DERY
OUTPUT OF DATA SET X, Y, DERY, IHLF, NDIM AND PRMT
IS PRMT(5) EQUAL TO 0
IS IHLF GREATER THAN 10
IS X LAST POINT OF INTEGRATION INTERVAL
EXIT
THIS IS LOGICAL END OF PREDICTOR-CORRECTOR PART
THIS IS DOUBLING OF INCREMENT H
IS H TOO SMALL
IS IHLF EQUAL TO 0
ARE THERE ENOUGH PRECEEDING POINTS
IS ISTEP EVEN
MULTIPLY H BY 2, SUBTRACT 1 FROM IHLF, SET ISTEP=0
STORE AUX(N-2) IN AUX(N-1) AND AUX(N+5) IN AUX(N+6)
STORE AUX(N-4) IN AUX(N-2) AND AUX(N+3) IN AUX(N+5)
STORE AUX(N-6) IN AUX(N-3) AND AUX(N+1) IN AUX(N+4)
COMPUTE NEW ERROR-VECTOR AUX(16) BY FORMULA 10
THIS IS HALFING OF INCREMENT H
ADD 1 TO IHLF
IS IHLF LESS THAN 11
DIVIDE H BY 2 AND REDEFINE STEP-COUNTER ISTEP=0
INTERPOLATE VECTOR Y(X-H) BY MEANS OF FORMULA 7
STORE RESULT IN Y
INTERPOLATE VECTOR Y(X-3H) BY MEANS OF FORMULA 8
STORE RESULT IN AUX(N-4)
STORE AUX(N-2) IN AUX(N-3) AND AUX(N+5) IN AUX(N+4)
SAVE X-H IN DELT
SET X=DELT-2*H
SET ISW2=10 FOR RETURN FROM COMPUTATION OF DERY
COMPUTE VECTOR OF DERIVATIVES DERY
STORE Y IN AUX(N-2) AND DERY IN AUX(N+5)
REPLACE Y BY AUX(N-4) AND X BY X-2*H
SET ISW2=11 FOR RETURN FROM COMPUTATION OF DERY
COMPUTE VECTOR OF DERIVATIVES DERY
STORE DERY IN AUX(N+3) AND DELT IN X
COMPUTE NEW ERROR-VECTOR AUX(16) BY FORMULA 9

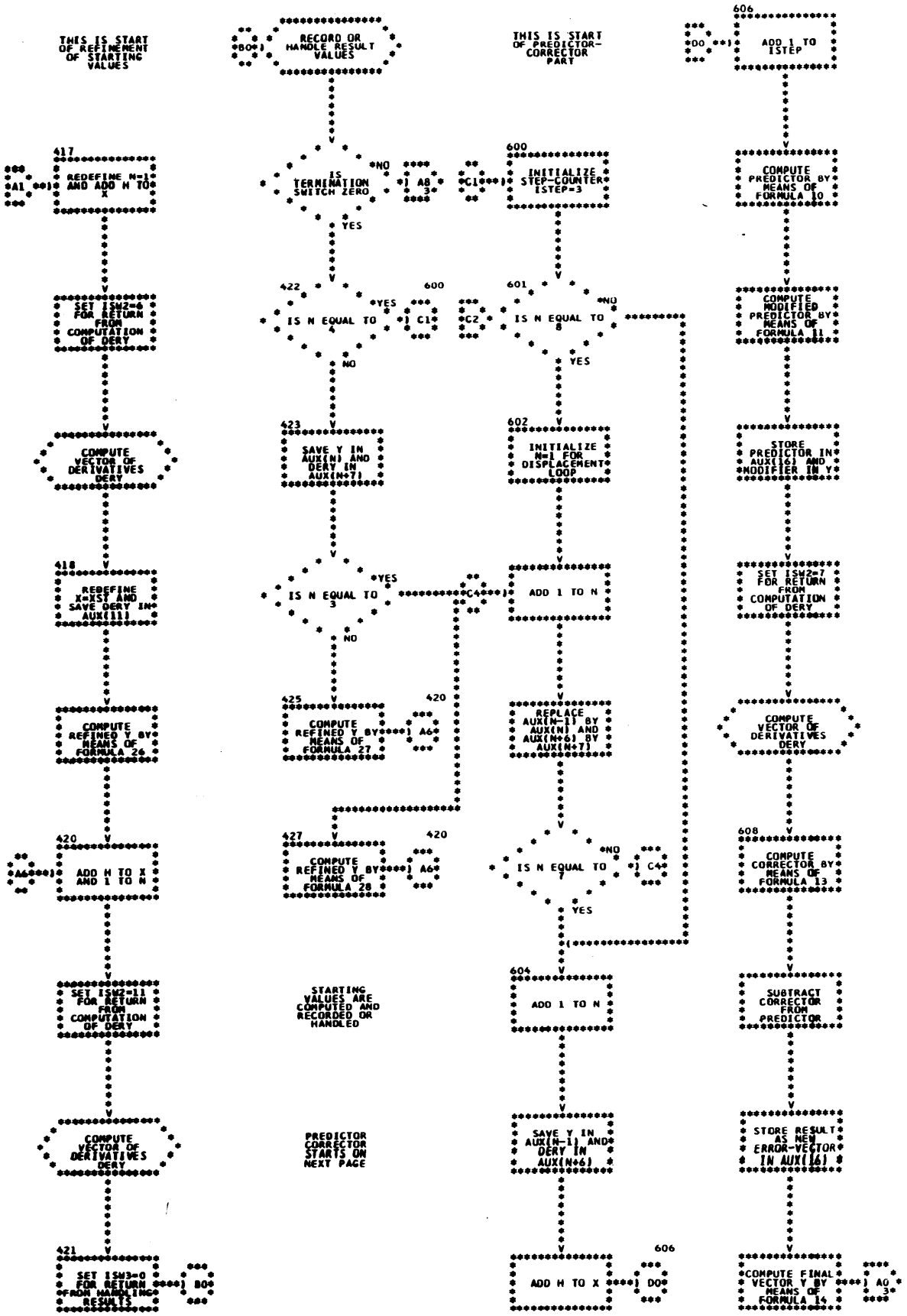

ENTER
THIS IS RUNGE-KUTTA PART OF SUBROUTINE HPCL OR DHPCL
200
SAVE X IN Z
USE FORMULA 11 AND STORE RESULT AS K1 IN AUX(5)
COMPUTE FACTOR OF A IN FORMULA 12 AND STORE IN Y
COMPUTE ARGUMENT X OF F AND A IN FORMULA 12
SET ISW2=2 FOR RETURN FROM COMPUTATION OF DERY
COMPUTE VECTOR OF DERIVATIVES DERY
202
MULTIPLY DERY BY H AND STORE RESULT AS K2 IN AUX(6)
COMPUTE FACTOR OF A IN FORMULA 13 AND STORE IN Y
COMPUTE ARGUMENT X OF F AND A IN FORMULA 13
SET ISW2=3 FOR RETURN FROM COMPUTATION OF DERY
CO
COMPUTE VECTOR OF DERIVATIVES DERY
204
MULTIPLY DERY BY H AND STORE RESULT AS K3 IN AUX(7)
COMPUTE FACTOR OF A IN FORMULA 14 AND STORE IN Y
COMPUTE ARGUMENT X OF F AND A IN FORMULA 14
SET ISW2=4 FOR RETURN FROM COMPUTATION OF DERY
COMPUTE VECTOR OF DERIVATIVES DERY
205
COMPUTE Y(X+H) BY MEANS OF FORMULA 15
STORE RESULT IN Y AND REPLACE X BY Z
EXIT

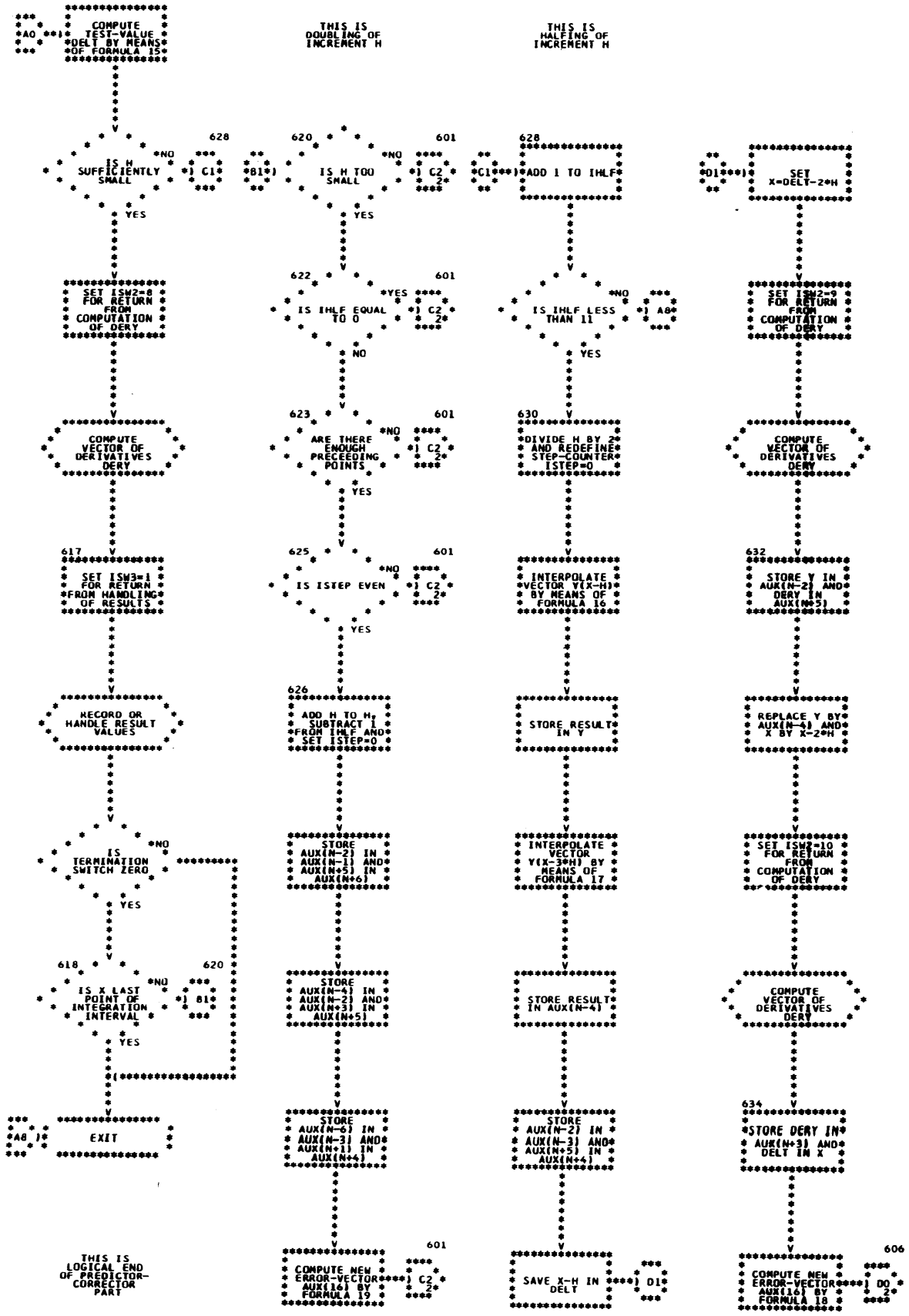

1
ENTER
COMPUTE A(X) AND STORE RESULT IN MATRIX A
COMPUTE F(X) AND STORE RESULT IN VECTOR DERY
THIS IS COMPUTATION OF DERIVATIVE VECTOR DERY
MULTIPLY VECTOR Y BY MATRIX A AND ADD VECTOR DERY
STORE RESULT IN VECTOR DERY
EXIT

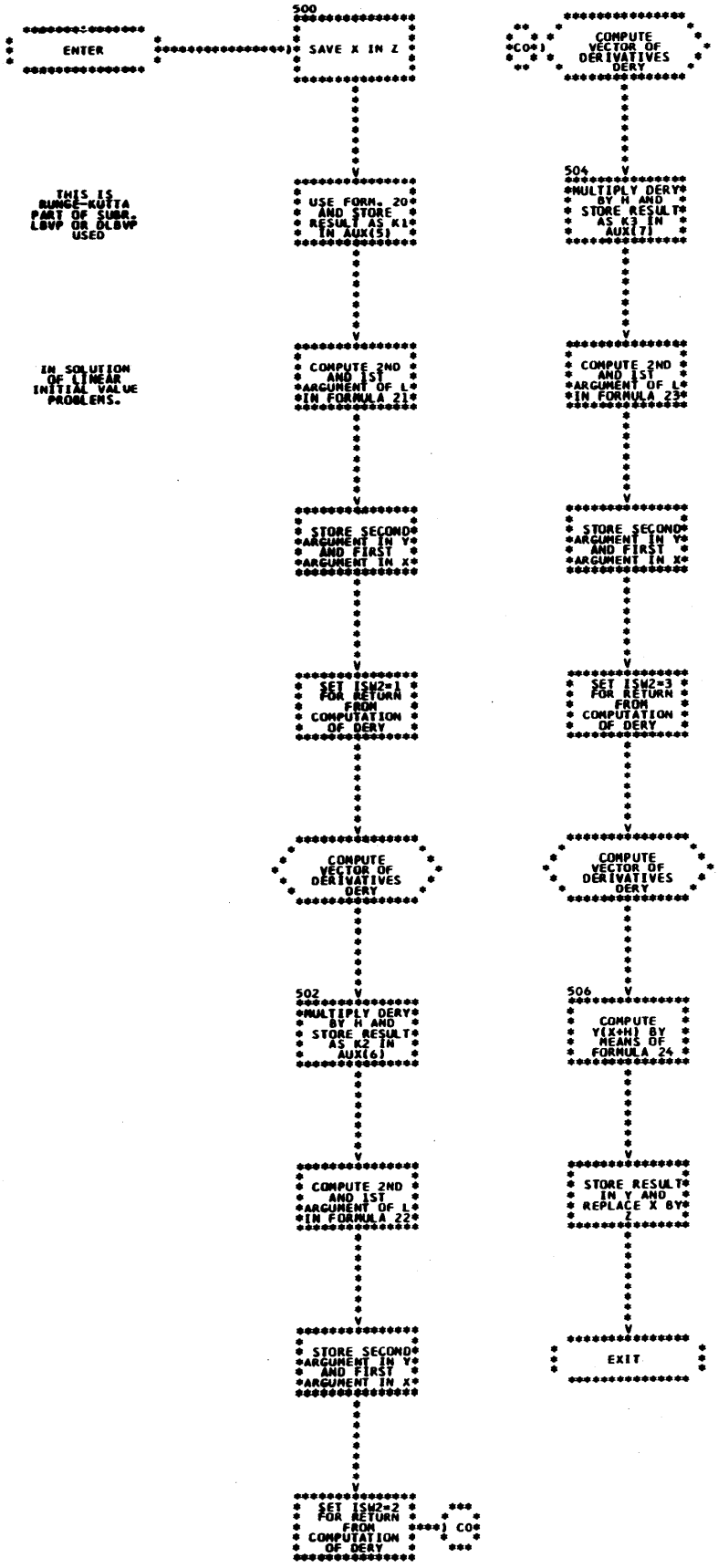

ENTER
HAS STEP-SIZE CORRECT SIGN
YES
NO
IS STEP-SIZE EQUAL TO 0
YES
NO
SET ERROR PARAMETER IHLF=12
EXIT
SET ERROR PARAMETER IHLF=13
EXIT
THESE WERE ERROR-EXITS IN CASE OF WRONG INPUT DATA
SEARCH FOR ZERO-COLUMNS IN MATRICES B AND C
SAVE VECTOR OF ERROR-WEIGHTS IN AUX(15)
SET ELEMENTS OF AUX(I) AND AUX(17) EQUAL TO 1
SET COLUMN-INDEX K=0 AND ZERO-COUNTERS IB=0 AND IC=0
ADD 1 TO COLUMN-INDEX K
IS K-TH COLUMN OF MATRIX B ZERO-COLUMN
YES
NO
REPLACE AUX(I,K) BY 0 AND ADD 1 TO COUNTER IB
IS K-TH COLUMN OF MATRIX C ZERO-COLUMN
YES
NO
REPLACE AUX(17,K) BY 0 AND ADD 1 TO COUNTER IC
IS K EQUAL TO NDIM
YES
NO
C1
B4
DETERMINATION OF LOWER AND UPPER BOUND
IS IC LESS THAN 1
YES
NO
INTERCHANGE LOWER AND UPPER BOUND XL AND XU
REPLACE INITIAL STEP-SIZE BY ITS NEGATIVE VALUE
SAVE AUX(I) IN AUX(17) INTERCHANGE MATRICES B AND C
COMPUTE F(XU) AND STORE RESULT IN Y
COMPUTE DERIVATIVE DF(XU) AND STORE RESULT IN DERY
SAVE Y IN AUX(18) AND DERY IN AUX(19)
SET TERMINATION SWITCH TO ZERO
C1
D1
D2
ADMINISTRATION OF ADJOINT INITIAL VALUE PROBLEMS
INITIALIZE COLUMN-INDEX K=0
ADD 1 TO COLUMN-INDEX K
IS K-TH COLUMN OF MATRIX C ZERO-COLUMN
YES
NO
COMPUTE A(XU) AND STORE RESULT IN MATRIX A
INITIALIZE SUM=0, I=XU AND XEND=XL
INITIALIZE GL AND DGL ACCORDING TO FORM. 35 AND 36
DEFINE VECTOR OF INITIAL VALUES Y(XU) BY FORMULA 4
STORE NEGATIVE K-TH ROW OF MATRIX A IN VECTOR DERY
SAVE VECTOR R IN AUX(20)
A1
A2
A7

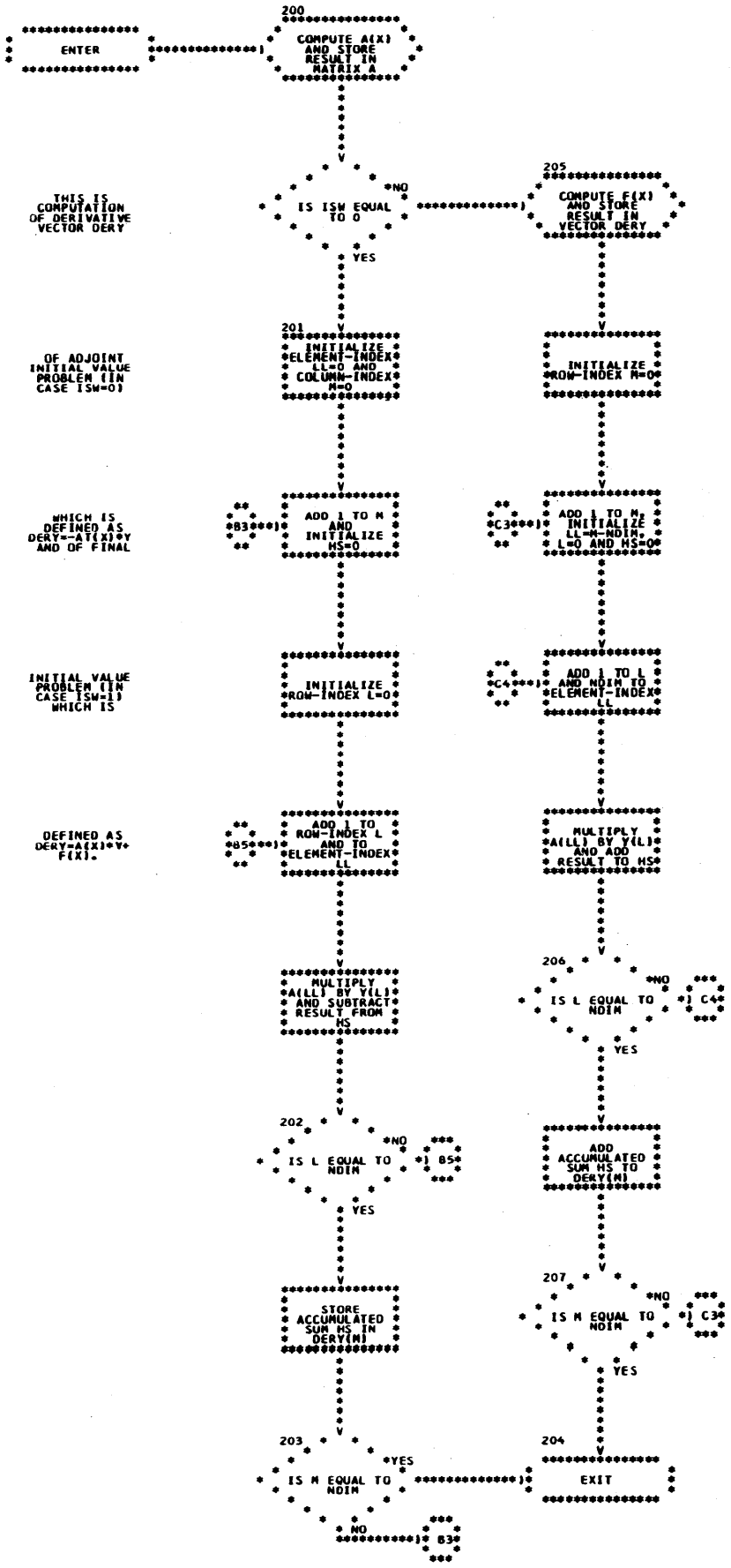

ADMINISTRATI-
ON OF FINAL
INITIAL VALUE
PROBLEM
A1
INITIALIZE
STEP-SIZE
H=(XL-XU)/16
SET ISW=0 FOR
RETURN FROM
INITIAL VALUE
PROBLEM
SOLVE K-TH
ADJOINT
INITIAL VALUE
PROBLEM
105
IS IHLF
GREATER THAN
10
YES
NO
EXIT
104
TRANSFORM
VECTOR R
ACCORDING TO
FORMULA 8
TRANSFORM
MATRIX B
ACCORDING TO
FORMULA 7
109
IS K EQUAL TO
NDIM
NO
YES
100
DZ
IA
81
110
A7
B1
110
SUBR. GELG OR
DGELG
GENERATES
SOLUTION OF
B*Y=R IN R
IS MATRIX B
SINGULAR
NO
YES
111
SET ERROR
PARAMETER
IHLF=14
112
SET
TERMINATION
SWITCH TO
ZERO
SET IHLF=
NEGATIVE
ERROR-CODE OF
GAUSS
ELIMINATION
INITIALIZE
STEP-SIZE,
SET X=XL AND
XEND=XU
DEFINE VECTOR
OF INITIAL
VALUES
Y(XL)=R
SET ISW=1 FOR
RETURN FROM
INITIAL VALUE
PROBLEM
114
CO
114
SET ISW=12
FOR RETURN
FROM
COMPUTATION
OF DERY
COMPUTE
VECTOR OF
DERIVATIVES
DERY
115
OUTPUT OF
DATA SET X,
Y, DERY,
IHLF, NDIM
AND PRMT
IS
TERMINATION
SWITCH ZERO
NO
YES
116
IS IHLF EQUAL
TO 0
NO
YES
SOLVE FINAL
INITIAL VALUE
PROBLEM
114
IS IHLF
GREATER THAN
10
YES
CO
NO
117
EXIT

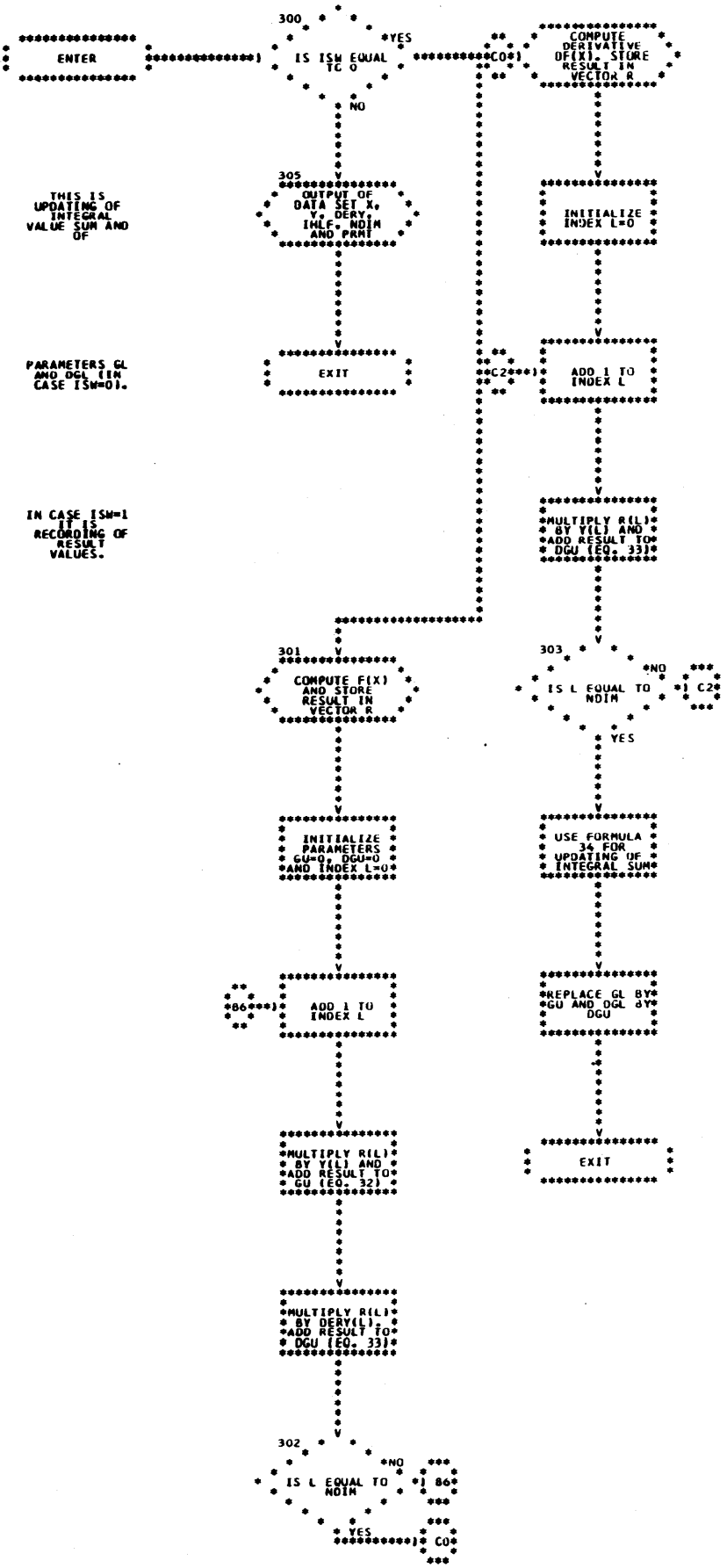

ENTER
400
SET UP N=1, XSI=X, IHLF=0 AND ERROR-VECTOR AUX(16)=0
THIS IS SOLUTION OF INITIAL VALUE PROBLEMS GENERATED BY SUBROUTINE LBVP OR DLBVP
STORE Y IN AUX(1) AND DERY IN AUX(8)
SET ISW1=1 FOR RETURN FROM RUNGE-KUTTA PART
COMPUTE Y(X+H) BY MEANS OF RUNGE-KUTTA PART
402
ADD H TO X AND STORE Y(X+H) IN AUX(2)
404
ADD 1 TO IHLF AND SUBTRACT H FROM X
B5
SAVE AUX(2) IN AUX(4) AND DIVIDE H BY 2
REDEFINE N=1
SET ISW1=2 FOR RETURN FROM RUNGE-KUTTA PART
COMPUTE Y(X+H) BY MEANS OF RUNGE-KUTTA PART
406
C0
406
C0
ADD H TO X
SET ISW2=4 FOR RETURN FROM COMPUTATION OF DERY
COMPUTE VECTOR OF DERIVATIVES DERY
407
SET N=2 AND SAVE Y IN AUX(2) AND DERY IN AUX(9)
SET ISW1=3 FOR RETURN FROM RUNGE-KUTTA PART
COMPUTE Y(X+H) BY MEANS OF RUNGE-KUTTA PART
409
PERFORM TEST ON SATISFACTORY ACCURACY (FORMULA 25)
IS H SUFFICIENTLY SMALL
YES
D1
NO
429
IS IHLF EQUAL TO 10
NO
404
B5
YES
430
SET ERROR PARAMETER IHLF=11 AND ADD H TO X
A8
3
SATISFACTORY ACCURACY WITHIN 10 BISECTIONS OF H
D1
ADD H TO X
SET ISW2=5 FOR RETURN FROM COMPUTATION OF DERY
COMPUTE VECTOR OF DERIVATIVES DERY
415
SAVE Y IN AUX(3) AND DERY IN AUX(10)
SET N=3
SET ISW1=4 FOR RETURN FROM RUNGE-KUTTA PART
COMPUTE Y(X+H) BY MEANS OF RUNGE-KUTTA PART
417
A1
2

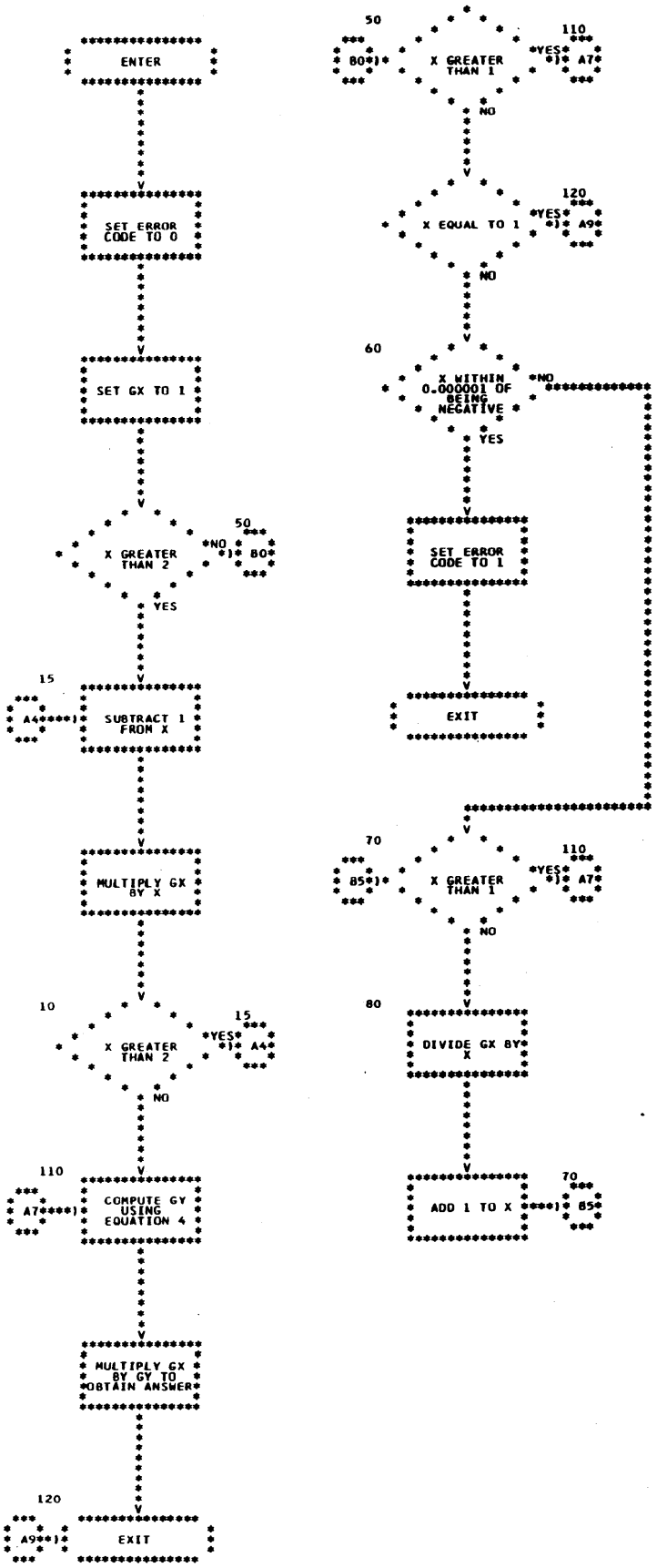

THIS IS START OF REFINEMENT OF STARTING VALUES
A1
417
REDEFINE N=1 AND ADD H TO X
SET ISW2=6 FOR RETURN FROM COMPUTATION OF DERY
COMPUTE VECTOR OF DERIVATIVES DERY
418
REDEFINE X=XST AND SAVE DERY IN AUX(11)
COMPUTE REFINED Y BY MEANS OF FORMULA 26
420
ADD H TO X AND 1 TO N
A6
SET ISW2=11 FOR RETURN FROM COMPUTATION OF DERY
COMPUTE VECTOR OF DERIVATIVES DERY
421
SET ISW3=0 FOR RETURN FROM HANDLING RESULTS
B0
RECORD OR HANDLE RESULT VALUES
IS TERMINATION SWITCH ZERO
NO
A8
3
YES
422
IS N EQUAL TO 4
YES
600
C1
NO
423
SAVE Y IN AUX(N) AND DERY IN AUX(N+7)
IS N EQUAL TO 3
YES
C4
NO
425
420
COMPUTE REFINED Y BY MEANS OF FORMULA 27
A6
427
420
COMPUTE REFINED Y BY MEANS OF FORMULA 28
A6
STARTING VALUES ARE COMPUTED AND RECORDED OR HANDLED
PREDICTOR CORRECTOR STARTS ON NEXT PAGE
THIS IS START OF PREDICTOR-CORRECTOR PART
600
INITIALIZE STEP-COUNTER ISTEP=3
C1
601
IS N EQUAL TO 8
NO
C2
YES
602
INITIALIZE N=1 FOR DISPLACEMENT LOOP
ADD 1 TO N
REPLACE AUX(N-1) BY AUX(N) AND AUX(N+6) BY AUX(N+7)
IS N EQUAL TO 7
NO
C4
YES
604
ADD 1 TO N
SAVE Y IN AUX(N-1) AND DERY IN AUX(N+6)
606
ADD H TO X
DO
606
DO
ADD 1 TO ISTEP
COMPUTE PREDICTOR BY MEANS OF FORMULA 10
COMPUTE MODIFIED PREDICTOR BY MEANS OF FORMULA 11
STORE PREDICTOR IN AUX(16) AND MODIFIER IN Y
SET ISW2=7 FOR RETURN FROM COMPUTATION OF DERY
COMPUTE VECTOR OF DERIVATIVES DERY
608
COMPUTE CORRECTOR BY MEANS OF FORMULA 13
SUBTRACT CORRECTOR FROM PREDICTOR
STORE RESULT AS NEW ERROR-VECTOR IN AUX(16)
COMPUTE FINAL VECTOR Y BY MEANS OF FORMULA 14
A0
3

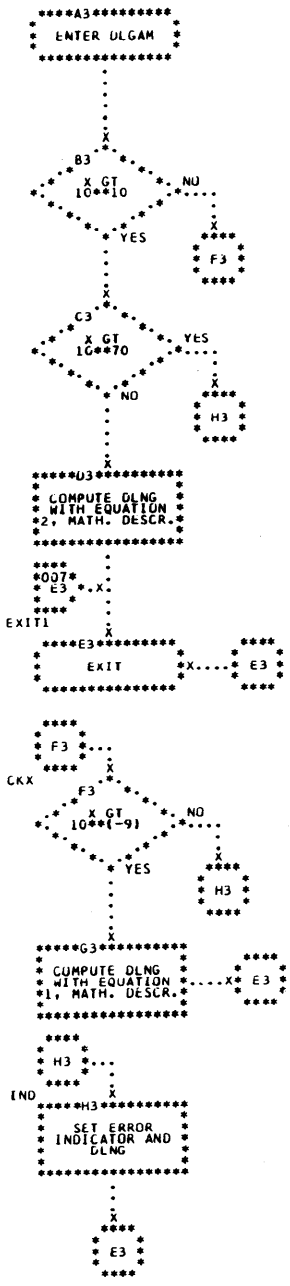

COMPUTE TEST-VALUE DELT BY MEANS OF FORMULA 15
A0
IS H SUFFICIENTLY SMALL
NO
YES
628
C1
B1
SET ISW2=0 FOR RETURN FROM COMPUTATION OF DERY
COMPUTE VECTOR OF DERIVATIVES DERY
617
SET ISW3=1 FOR RETURN FROM HANDLING OF RESULTS
RECORD OR HANDLE RESULT VALUES
IS TERMINATION SWITCH ZERO
618
IS X LAST POINT OF INTEGRATION INTERVAL
620
B1
A8
EXIT
THIS IS LOGICAL END OF PREDICTOR-CORRECTOR PART
THIS IS DOUBLING OF INCREMENT H
620
IS H TOO SMALL
601
C2
2*
622
IS IHLF EQUAL TO 0
623
ARE THERE ENOUGH PRECEEDING POINTS
625
IS ISTEP EVEN
626
ADD H TO H, SUBTRACT 1 FROM IHLF AND SET ISTEP=0
STORE AUX(N-2) IN AUX(N-1) AND AUX(N+5) IN AUX(N+6)
STORE AUX(N-4) IN AUX(N-2) AND AUX(N+3) IN AUX(N+5)
STORE AUX(N-6) IN AUX(N-3) AND AUX(N+1) IN AUX(N+4)
COMPUTE NEW ERROR-VECTOR AUX(16) BY FORMULA 19
THIS IS HALFING OF INCREMENT H
628
ADD 1 TO IHLF
C1
IS IHLF LESS THAN 11
A8
630
DIVIDE H BY 2 AND REDEFINE STEP-COUNTER ISTEP=0
INTERPOLATE VECTOR Y(X-H) BY MEANS OF FORMULA 16
STORE RESULT IN Y
INTERPOLATE VECTOR Y(X-3*H) BY MEANS OF FORMULA 17
STORE RESULT IN AUX(N-4)
STORE AUX(N-2) IN AUX(N-3) AND AUX(N+5) IN AUX(N+4)
SAVE X-H IN DELT
D1
SET X=DELT-2*H
SET ISW2=9 FOR RETURN FROM COMPUTATION OF DERY
COMPUTE VECTOR OF DERIVATIVES DERY
632
STORE Y IN AUX(N-2) AND DERY IN AUX(N+5)
REPLACE Y BY AUX(N-4) AND X BY X-2*H
SET ISW2=10 FOR RETURN FROM COMPUTATION OF DERY
COMPUTE VECTOR OF DERIVATIVES DERY
634
STORE DERY IN AUX(N+3) AND DELT IN X
COMPUTE NEW ERROR-VECTOR AUX(16) BY FORMULA 18
D0
606
2*

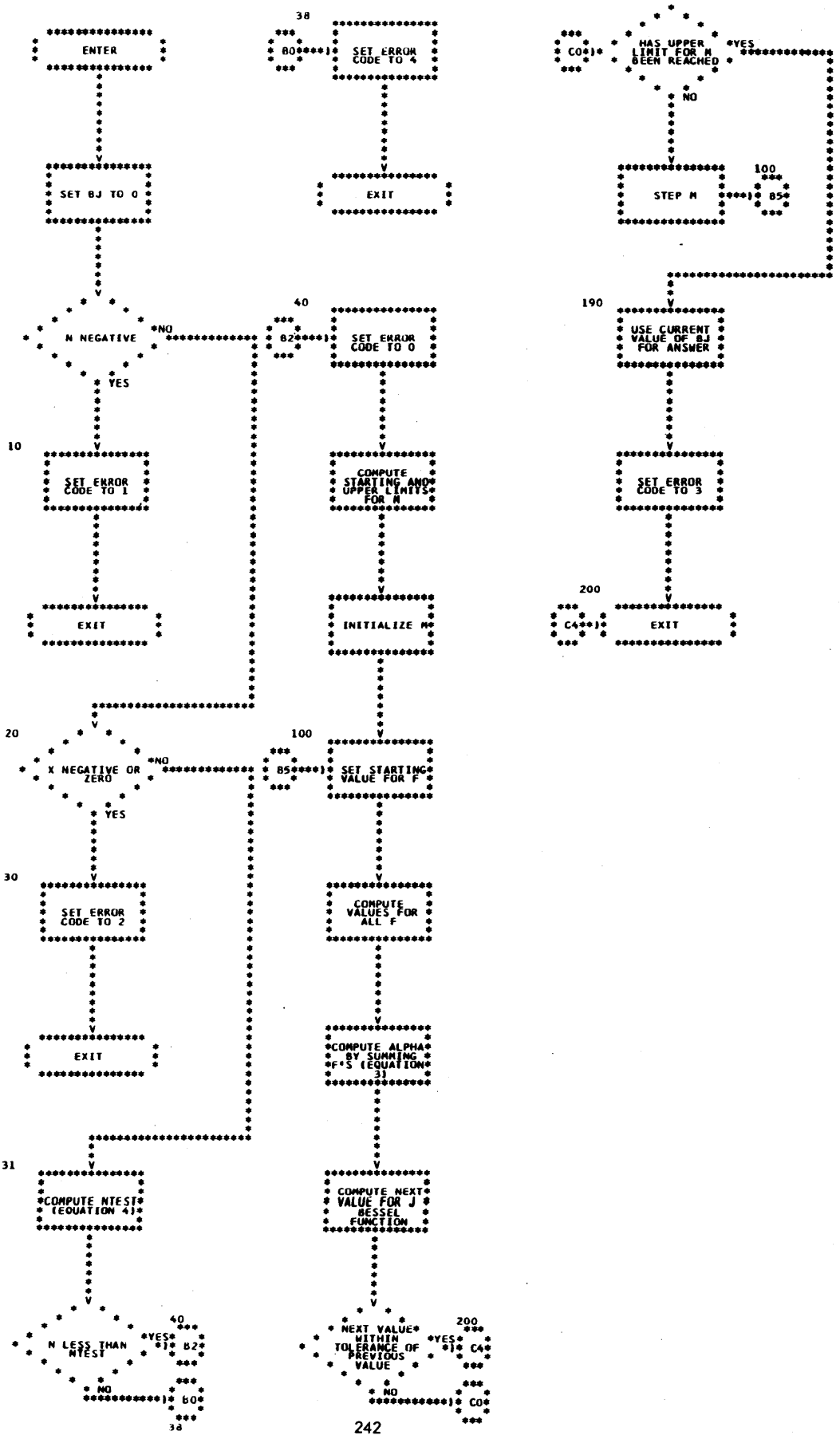

ENTER
500
SAVE X IN Z
USE FORM. 20 AND STORE RESULT AS K1 IN AUX(5)
COMPUTE 2ND AND 1ST ARGUMENT OF L IN FORMULA 21
STORE SECOND ARGUMENT IN Y AND FIRST ARGUMENT IN X
SET ISW2=1 FOR RETURN FROM COMPUTATION OF DERY
COMPUTE VECTOR OF DERIVATIVES DERY
502
MULTIPLY DERY BY H AND STORE RESULT AS K2 IN AUX(6)
COMPUTE 2ND AND 1ST ARGUMENT OF L IN FORMULA 22
STORE SECOND ARGUMENT IN Y AND FIRST ARGUMENT IN X
SET ISW2=2 FOR RETURN FROM COMPUTATION OF DERY
CO
CO
COMPUTE VECTOR OF DERIVATIVES DERY
504
MULTIPLY DERY BY H AND STORE RESULT AS K3 IN AUX(7)
COMPUTE 2ND AND 1ST ARGUMENT OF L IN FORMULA 23
STORE SECOND ARGUMENT IN Y AND FIRST ARGUMENT IN X
SET ISW2=3 FOR RETURN FROM COMPUTATION OF DERY
COMPUTE VECTOR OF DERIVATIVES DERY
506
COMPUTE Y(X+H) BY MEANS OF FORMULA 24
STORE RESULT IN Y AND REPLACE X BY Z
EXIT
THIS IS RUNGE-KUTTA PART OF SUBR. LBVP OR DLBVP USED IN SOLUTION OF LINEAR INITIAL VALUE PROBLEMS.

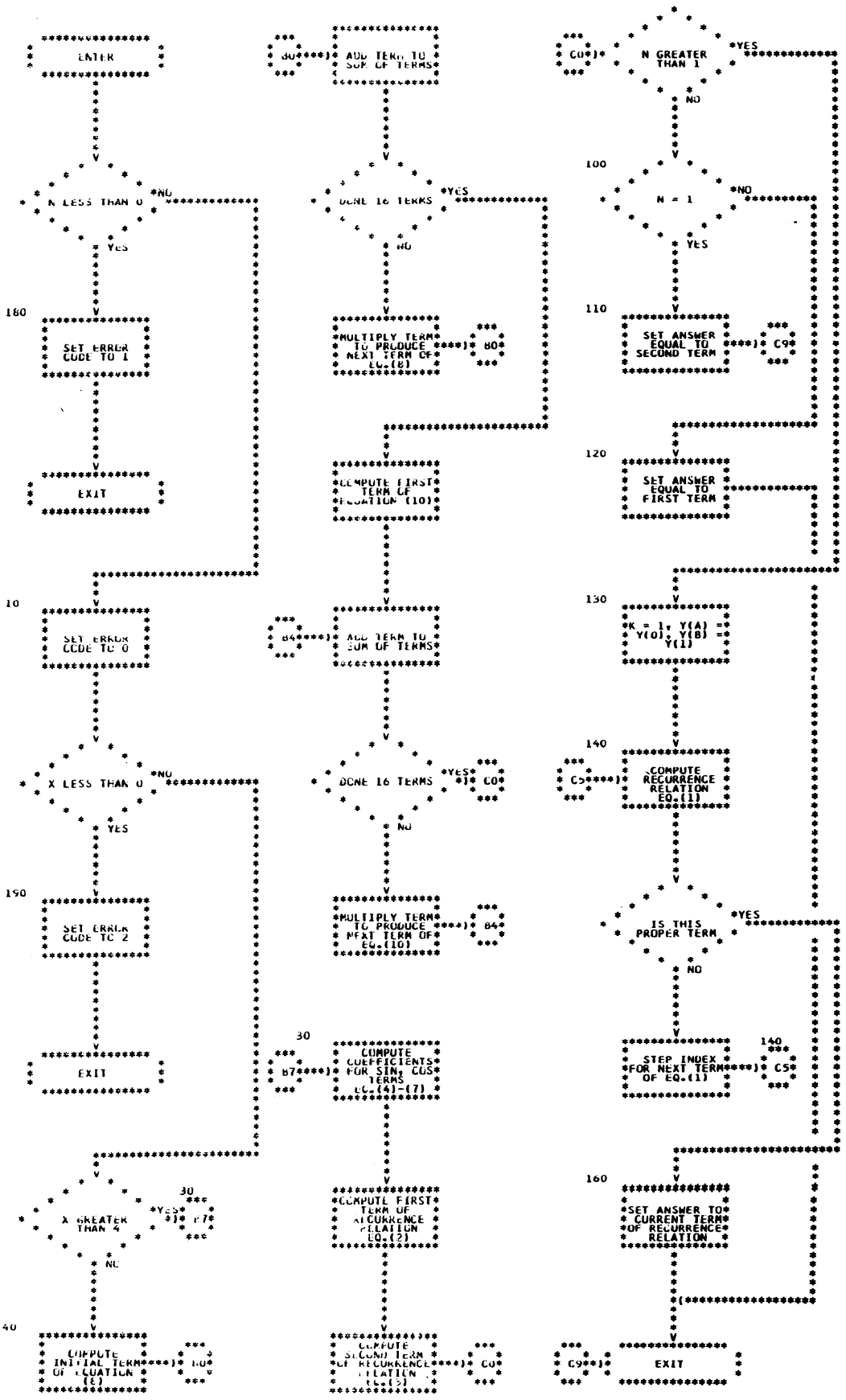

ENTER
200
COMPUTE A(X) AND STORE RESULT IN MATRIX A
THIS IS COMPUTATION OF DERIVATIVE VECTOR DERY
IS ISW EQUAL TO 0
NO
YES
205
COMPUTE F(X) AND STORE RESULT IN VECTOR DERY
OF ADJOINT INITIAL VALUE PROBLEM (IN CASE ISW=0)
201
INITIALIZE ELEMENT-INDEX LL=0 AND COLUMN-INDEX M=0
INITIALIZE ROW-INDEX M=0
WHICH IS DEFINED AS DERY=-A(X)*Y AND OF FINAL
B3
ADD 1 TO M AND INITIALIZE MS=0
C3
ADD 1 TO M, INITIALIZE LL=M-NDIM, L=0 AND HS=0
INITIAL VALUE PROBLEM (IN CASE ISW=1) WHICH IS
INITIALIZE ROW-INDEX L=0
C4
ADD 1 TO L AND NDIM TO ELEMENT-INDEX LL
DEFINED AS DERY=A(X)*Y+F(X).
B5
ADD 1 TO ROW-INDEX L AND TO ELEMENT-INDEX LL
MULTIPLY A(LL) BY Y(L) AND ADD RESULT TO HS
MULTIPLY A(LL) BY Y(L) AND SUBTRACT RESULT FROM HS
206
IS L EQUAL TO NDIM
NO
C4
YES
202
IS L EQUAL TO NDIM
NO
B5
YES
ADD ACCUMULATED SUM HS TO DERY(M)
STORE ACCUMULATED SUM HS IN DERY(M)
207
IS M EQUAL TO NDIM
NO
C3
YES
203
IS M EQUAL TO NDIM
YES
NO
B3
204
EXIT

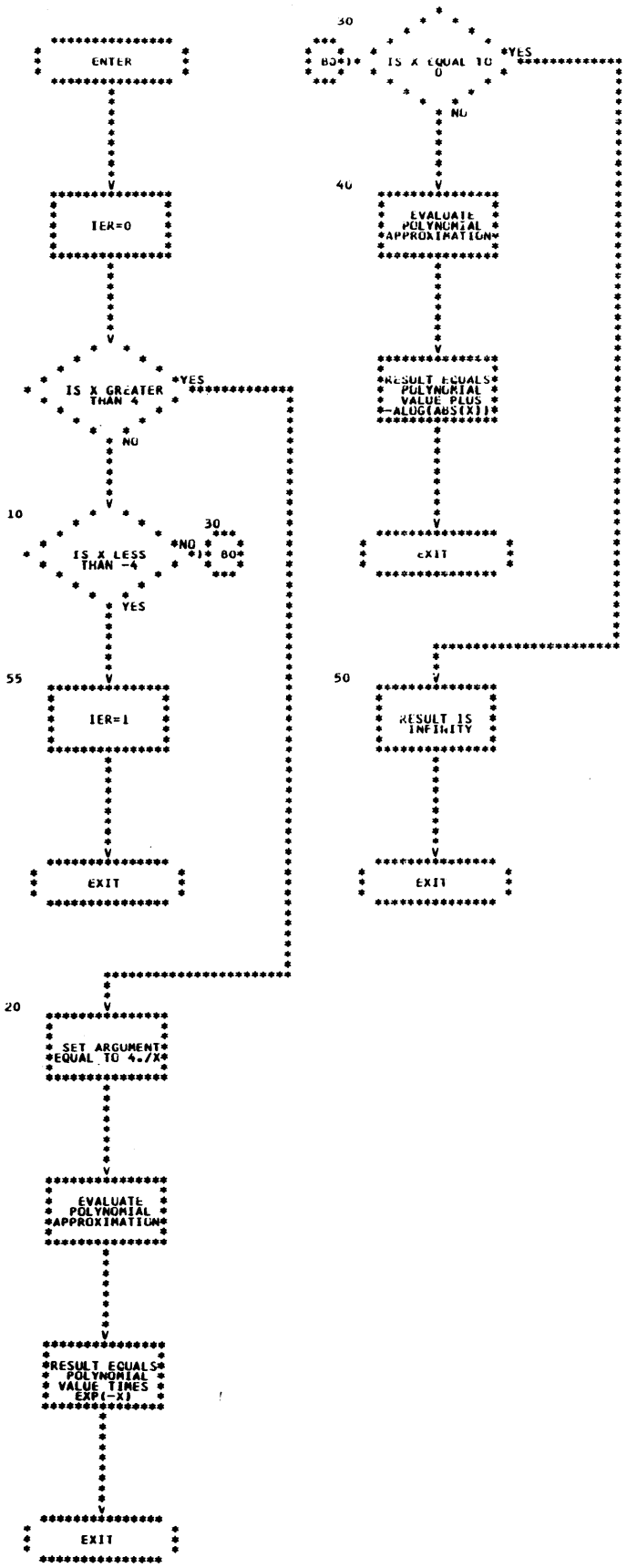

ENTER
300
IS ISW EQUAL TO 0
YES
NO
C0
COMPUTE DERIVATIVE DF(X), STORE RESULT IN VECTOR R
305
OUTPUT OF DATA SET X, Y, DERY, IHLF, NDIM AND PRMT
INITIALIZE INDEX L=0
THIS IS UPDATING OF INTEGRAL VALUE SUM AND OF
EXIT
PARAMETERS GL AND DGL (IN CASE ISW=0).
C2
ADD 1 TO INDEX L
MULTIPLY R(L) BY Y(L) AND ADD RESULT TO DGU (EQ. 33)
IN CASE ISW=1 IT IS RECORDING OF RESULT VALUES.
301
COMPUTE F(X) AND STORE RESULT IN VECTOR R
303
IS L EQUAL TO NDIM
NO
YES
C2
INITIALIZE PARAMETERS GU=0, DGU=0 AND INDEX L=0
USE FORMULA 34 FOR UPDATING OF INTEGRAL SUM
86
ADD 1 TO INDEX L
REPLACE GL BY GU AND DGL BY DGU
MULTIPLY R(L) BY Y(L) AND ADD RESULT TO GU (EQ. 32)
MULTIPLY R(L) BY DERY(L), ADD RESULT TO DGU (EQ. 33)
302
IS L EQUAL TO NDIM
NO
YES
86
C0
EXIT

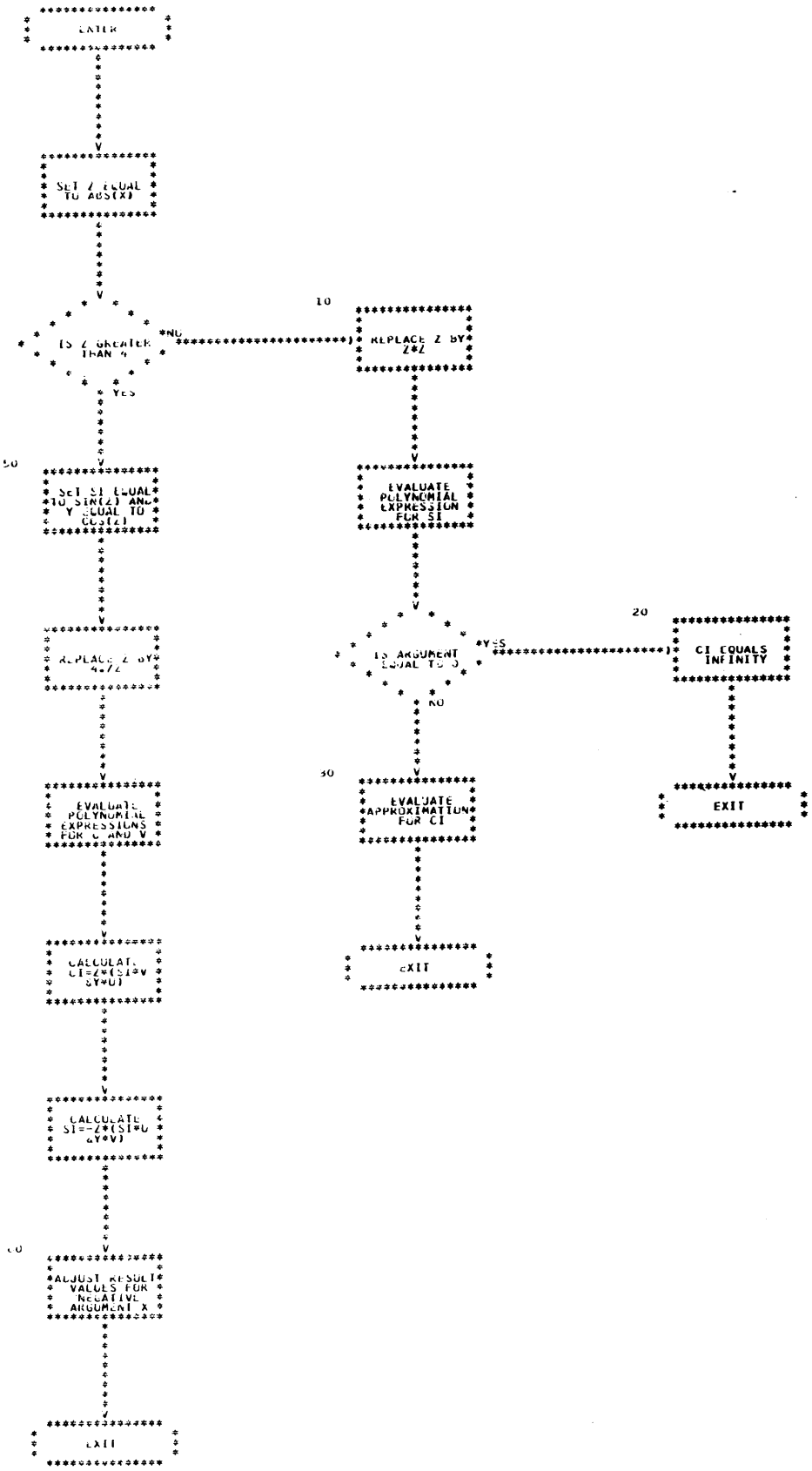

ENTER
SET ERROR CODE TO 0
SET GX TO 1
X GREATER THAN 2
50
NO
80
YES
15
A4
SUBTRACT 1 FROM X
MULTIPLY GX BY X
10
X GREATER THAN 2
15
YES
A4
NO
110
A7
COMPUTE GY USING EQUATION 4
MULTIPLY GX BY GY TO OBTAIN ANSWER
120
A9
EXIT
50
X GREATER THAN 1
110
YES
A7
NO
120
X EQUAL TO 1
YES
A9
NO
60
X WITHIN 0.000001 OF BEING NEGATIVE
NO
YES
SET ERROR CODE TO 1
EXIT
70
X GREATER THAN 1
110
YES
A7
NO
80
DIVIDE GX BY X
ADD 1 TO X
70
B5

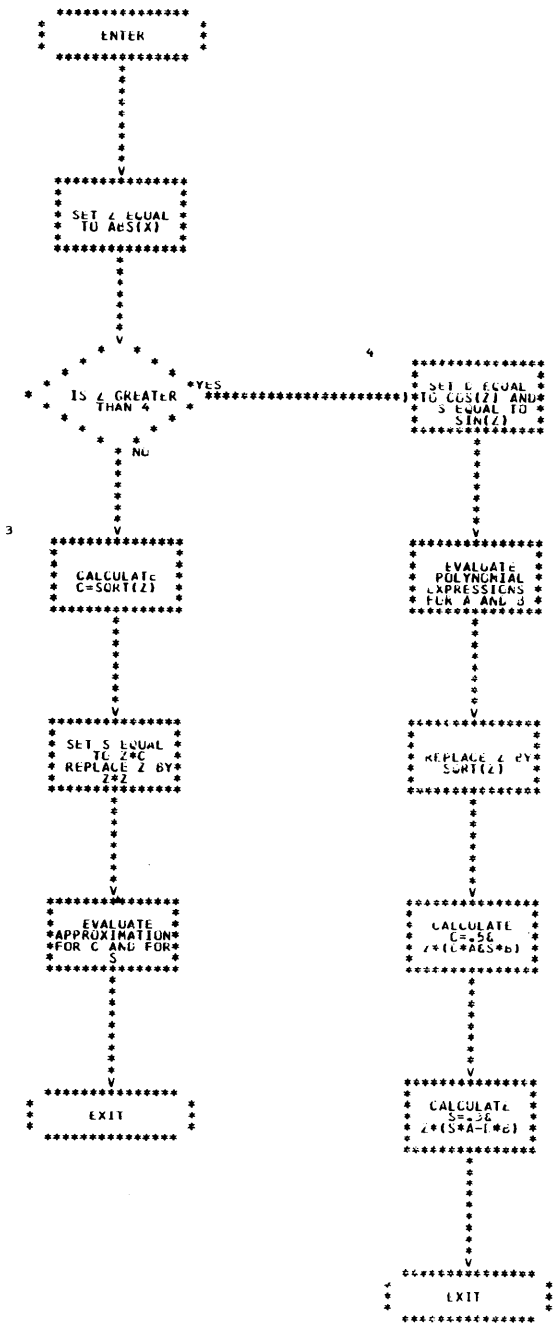

A3
ENTER DLGAM
B3
X GT 10**10
NO
YES
F3
C3
X GT 10**70
YES
NO
H3
D3
COMPUTE DLNG WITH EQUATION 2, MATH. DESCR.
007
E3
EXIT1
E3
EXIT
E3
F3
CKX
F3
X GT 10**(-9)
NO
YES
H3
G3
COMPUTE DLNG WITH EQUATION 1, MATH. DESCR.
E3
H3
END
H3
SET ERROR INDICATOR AND DLNG
E3

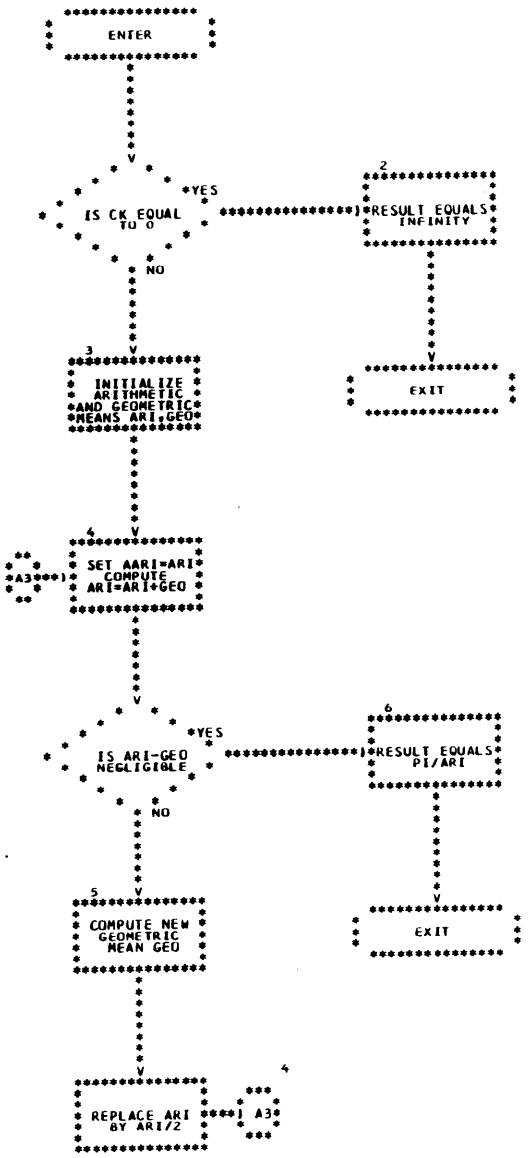

ENTER
SET BJ TO 0
N NEGATIVE
*NO
YES
10
SET ERROR CODE TO 1
EXIT
20
X NEGATIVE OR ZERO
*NO
YES
30
SET ERROR CODE TO 2
EXIT
31
COMPUTE NTEST (EQUATION 4)
N LESS THAN NTEST
40
YES
B2
NO
B0
38
B0
SET ERROR CODE TO 4
EXIT
40
B2
SET ERROR CODE TO 0
COMPUTE STARTING AND UPPER LIMITS FOR M
INITIALIZE M
100
B5
SET STARTING VALUE FOR F
COMPUTE VALUES FOR ALL F
COMPUTE ALPHA BY SUMMING *F'S (EQUATION* 3)
COMPUTE NEXT* VALUE FOR J * BESSEL FUNCTION
NEXT VALUE* WITHIN TOLERANCE OF PREVIOUS VALUE
YES
200
C4
NO
C0
C0
HAS UPPER LIMIT FOR M BEEN REACHED
*YES
NO
STEP M
100
B5
190
USE CURRENT VALUE OF BJ FOR ANSWER
SET ERROR CODE TO 3
200
C4
EXIT

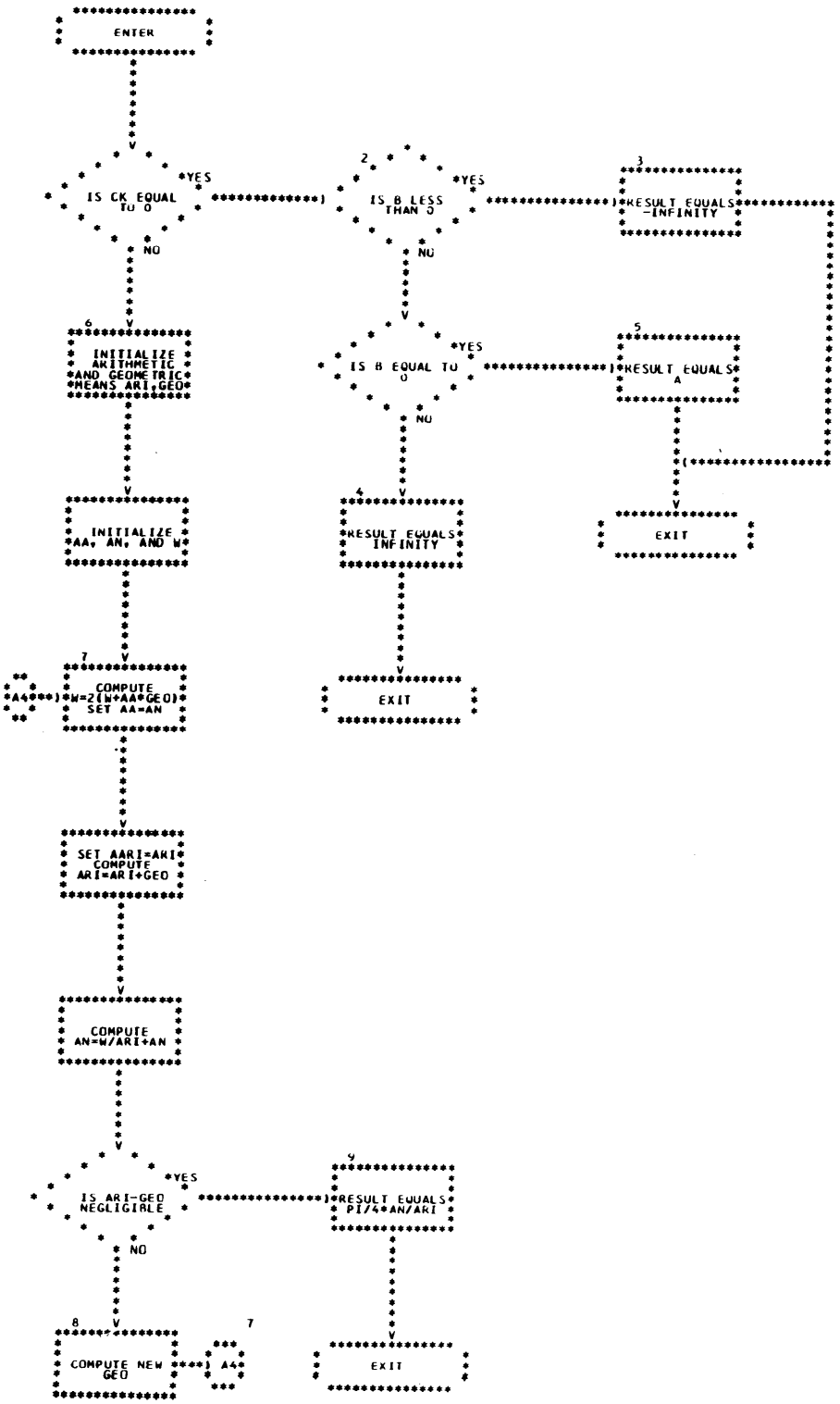

ENTER
N LESS THAN 0
NO
YES
160
SET ERROR CODE TO 1
EXIT
10
SET ERROR CODE TO 0
X LESS THAN 0
NO
YES
190
SET ERROR CODE TO 2
EXIT
X GREATER THAN 4
30
YES
NO
40
COMPUTE INITIAL TERM OF EQUATION (6)
B0
ADD TERM TO SUM OF TERMS
DONE 16 TERMS
YES
NO
MULTIPLY TERM TO PRODUCE NEXT TERM OF EQ.(8)
B0
COMPUTE FIRST TERM OF EQUATION (10)
B4
ADD TERM TO SUM OF TERMS
DONE 16 TERMS
YES
C0
NO
MULTIPLY TERM TO PRODUCE NEXT TERM OF EQ.(10)
B4
30
B7
COMPUTE COEFFICIENTS FOR SIN, COS TERMS EQ.(4)-(7)
COMPUTE FIRST TERM OF RECURRENCE RELATION EQ.(2)
COMPUTE SECOND TERM OF RECURRENCE RELATION EQ.(3)
C0
C0
N GREATER THAN 1
YES
NO
100
N = 1
NO
YES
110
SET ANSWER EQUAL TO SECOND TERM
C9
120
SET ANSWER EQUAL TO FIRST TERM
130
K = 1, Y(A) = Y(0), Y(B) = Y(1)
140
C5
COMPUTE RECURRENCE RELATION EQ.(1)
IS THIS PROPER TERM
YES
NO
140
STEP INDEX FOR NEXT TERM OF EQ.(1)
C5
160
SET ANSWER TO CURRENT TERM OF RECURRENCE RELATION
C9
EXIT

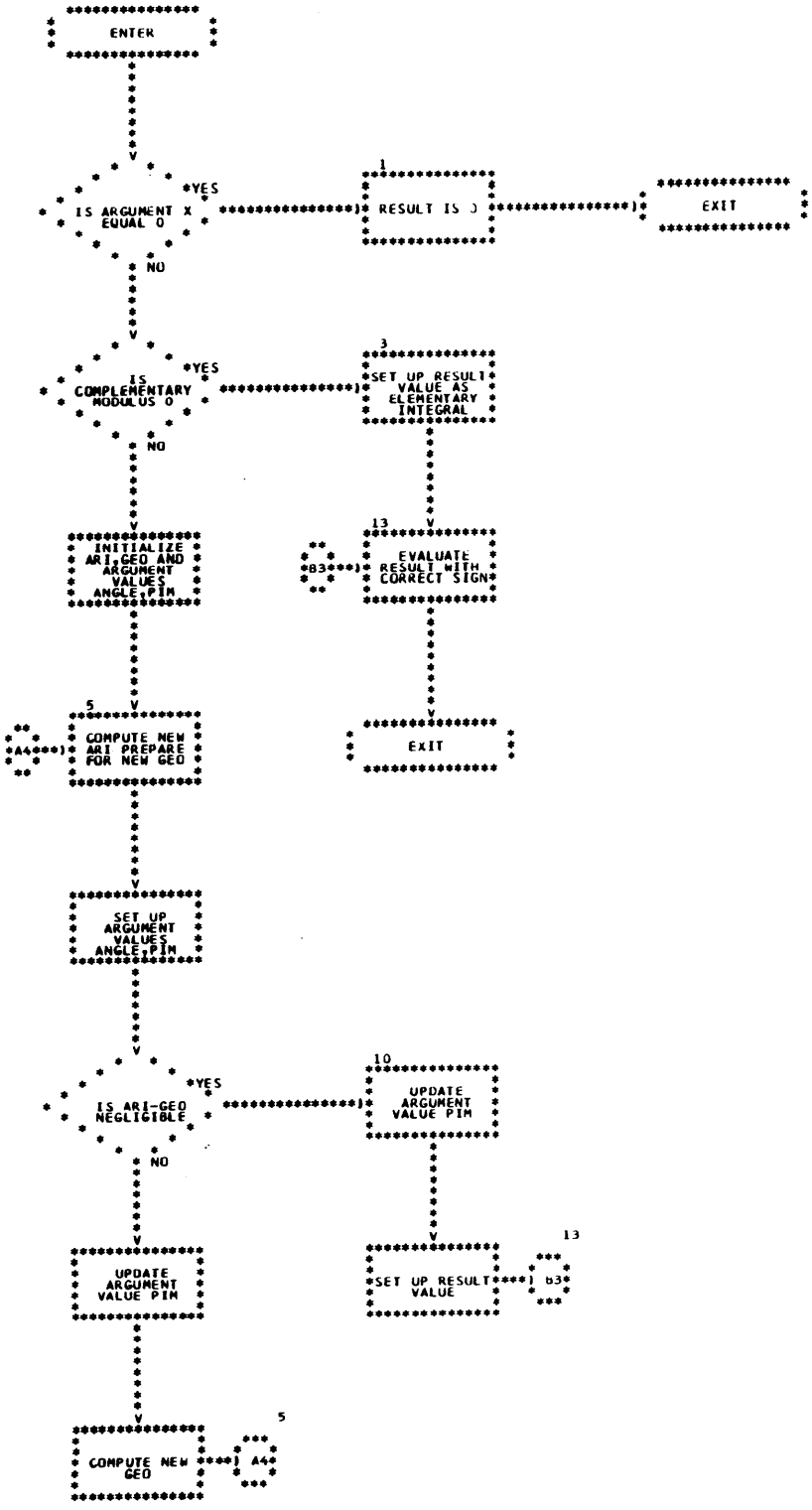

ENTER
IER=0
IS X GREATER THAN 4
YES
NO
10
IS X LESS THAN -4
30
NO
80
YES
55
IER=1
EXIT
20
SET ARGUMENT EQUAL TO 4./X
EVALUATE POLYNOMIAL APPROXIMATION
RESULT EQUALS POLYNOMIAL VALUE TIMES EXP(-X)
EXIT
30
80
IS X EQUAL TO 0
YES
NO
40
EVALUATE POLYNOMIAL APPROXIMATION
RESULT EQUALS POLYNOMIAL VALUE PLUS -ALOG(ABS(X))
EXIT
50
RESULT IS INFINITY
EXIT

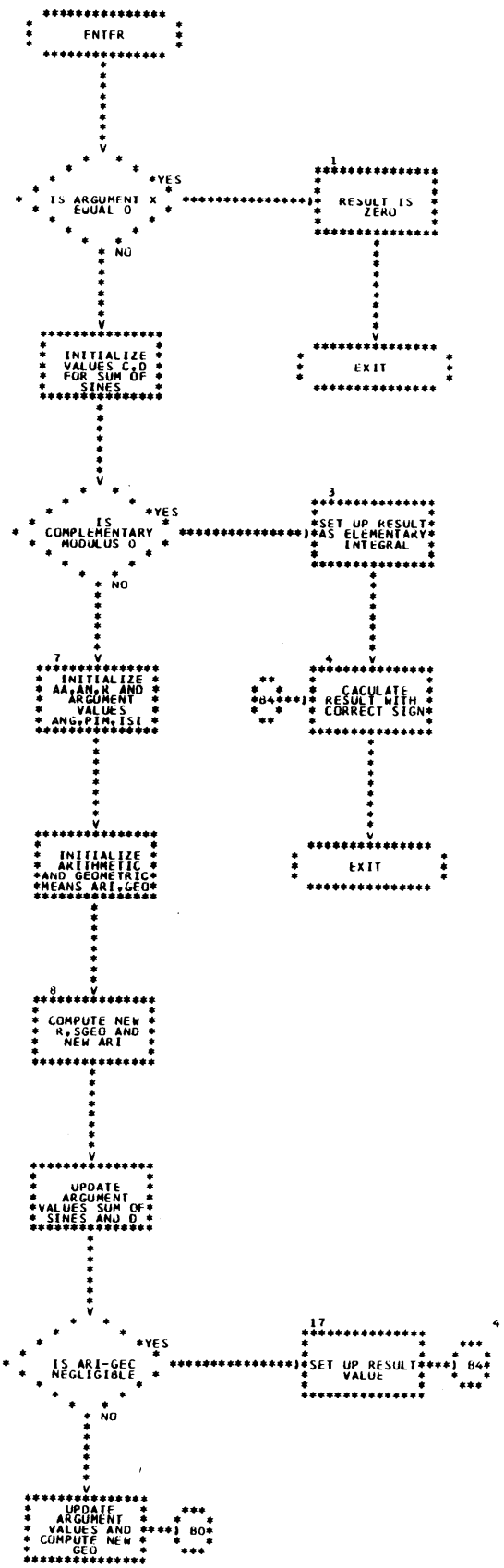

ENTER
SET Z EQUAL TO ABS(X)
IS Z GREATER THAN 4
NO
YES
10
REPLACE Z BY Z*Z
EVALUATE POLYNOMIAL EXPRESSION FOR SI
IS ARGUMENT EQUAL TO 0
YES
20
CI EQUALS INFINITY
EXIT
NO
30
EVALUATE APPROXIMATION FOR CI
EXIT
50
SET SI EQUAL TO SIN(Z) AND Y EQUAL TO COS(Z)
REPLACE Z BY 4/Z
EVALUATE POLYNOMIAL EXPRESSIONS FOR U AND V
CALCULATE CI=Z*(SI*V-CY*U)
CALCULATE SI=-Z*(SI*U+CY*V)
60
ADJUST RESULT VALUES FOR NEGATIVE ARGUMENT X
EXIT

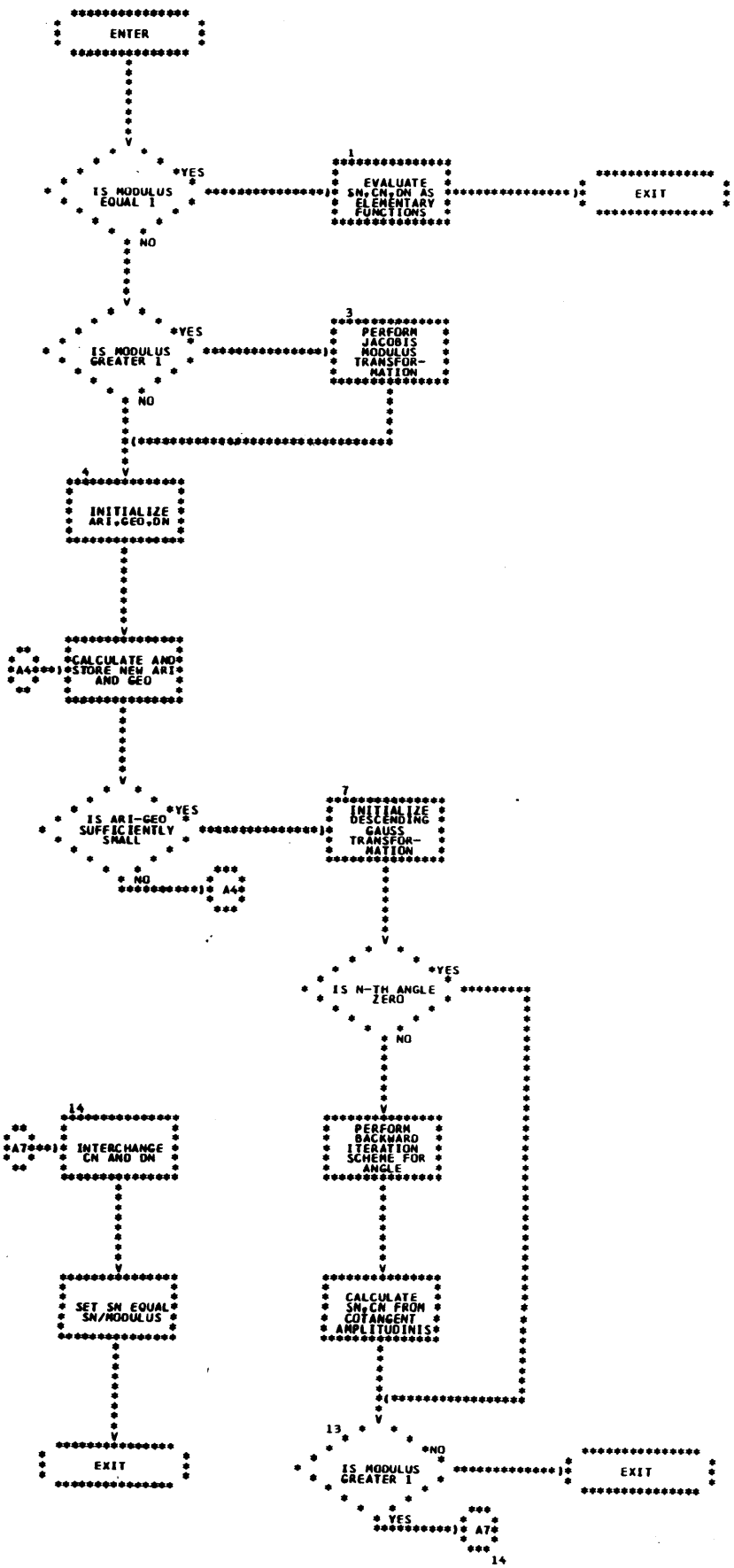

ENTER
SET Z EQUAL TO ABS(X)
IS Z GREATER THAN 4
YES
NO
SET C EQUAL TO COS(Z) AND S EQUAL TO SIN(Z)
CALCULATE C=SQRT(Z)
EVALUATE POLYNOMIAL EXPRESSIONS FOR A AND B
SET S EQUAL TO Z*C REPLACE Z BY Z*Z
REPLACE Z BY SQRT(Z)
EVALUATE APPROXIMATION FOR C AND FOR S
CALCULATE C=.5C Z*(1.+A&S*B)
CALCULATE S=.5S Z*(S*A-C*B)
EXIT
EXIT


ENTER
IS CK EQUAL TO 0
YES
NO
2
RESULT EQUALS INFINITY
EXIT
3
INITIALIZE ARITHMETIC AND GEOMETRIC MEANS ARI,GEO
4
SET AARI=ARI COMPUTE ARI=ARI+GEO
A3
IS ARI-GEO NEGLIGIBLE
YES
NO
6
RESULT EQUALS PI/ARI
EXIT
5
COMPUTE NEW GEOMETRIC MEAN GEO
REPLACE ARI BY ARI/2
4
A3


ENTER
IS CK EQUAL TO 0
YES
NO
IS B LESS THAN 0
YES
NO
RESULT EQUALS -INFINITY
IS B EQUAL TO 0
YES
NO
RESULT EQUALS A
EXIT
INITIALIZE ARITHMETIC AND GEOMETRIC MEANS ARI,GEO
RESULT EQUALS INFINITY
EXIT
INITIALIZE AA, AN, AND W
COMPUTE W=2(W+AA*GEO) SET AA=AN
A4
SET AARI=ARI COMPUTE ARI=ARI+GEO
COMPUTE AN=W/ARI+AN
IS ARI-GEO NEGLIGIBLE
YES
NO
RESULT EQUALS PI/4*AN/ARI
COMPUTE NEW GEO
A4
EXIT


ENTER
IS ARGUMENT X EQUAL 0
YES
NO
1
RESULT IS 1
EXIT
IS COMPLEMENTARY MODULUS 0
YES
NO
3
SET UP RESULT VALUE AS ELEMENTARY INTEGRAL
INITIALIZE ARI,GEO AND ARGUMENT VALUES ANGLE,PIM
13
B3
EVALUATE RESULT WITH CORRECT SIGN
EXIT
5
A4
COMPUTE NEW ARI PREPARE FOR NEW GEO
SET UP ARGUMENT VALUES ANGLE,PIM
IS ARI-GEO NEGLIGIBLE
YES
NO
10
UPDATE ARGUMENT VALUE PIM
UPDATE ARGUMENT VALUE PIM
13
SET UP RESULT VALUE
B3
COMPUTE NEW GEO
5
A4


ENTER
IS ARGUMENT X EQUAL 0
YES
NO
1
RESULT IS ZERO
EXIT
INITIALIZE VALUES C,D FOR SUM OF SINES
IS COMPLEMENTARY MODULUS 0
YES
NO
3
SET UP RESULT AS ELEMENTARY INTEGRAL
EXIT
7
INITIALIZE AA,AN,R AND ARGUMENT VALUES ANG,PIM,ISI
4
CACULATE RESULT WITH CORRECT SIGN
EXIT
INITIALIZE ARITHMETIC AND GEOMETRIC MEANS ARI,GEO
8
COMPUTE NEW R,SGEO AND NEW ARI
UPDATE ARGUMENT VALUES SUM OF SINES AND D
IS ARI-GEO NEGLIGIBLE
YES
NO
17
SET UP RESULT VALUE
4
84
UPDATE ARGUMENT VALUES AND COMPUTE NEW GEO
80


ENTER
IS MODULUS EQUAL 1
*YES
1
EVALUATE SN,CN,DN AS ELEMENTARY FUNCTIONS
EXIT
*NO
IS MODULUS GREATER 1
*YES
3
PERFORM JACOBIS MODULUS TRANSFOR-MATION
*NO
4
INITIALIZE ARI,GEO,DN
A4
CALCULATE AND STORE NEW ARI AND GEO
IS ARI-GEO SUFFICIENTLY SMALL
*YES
7
INITIALIZE DESCENDING GAUSS TRANSFOR-MATION
*NO
A4
IS N-TH ANGLE ZERO
*YES
*NO
14
A7
INTERCHANGE CN AND DN
PERFORM BACKWARD ITERATION SCHEME FOR ANGLE
SET SN EQUAL SN/MODULUS
CALCULATE SN,CN FROM COTANGENT AMPLITUDINIS
EXIT
13
IS MODULUS GREATER 1
*NO
EXIT
*YES
A7
14

Printed in U.S.A. GY20-0092-2

IBM

International Business Machines Corporation
Data Processing Division
1133 Westchester Avenue, White Plains, New York 10604
(U.S.A. only)

IBM World Trade Corporation
821 United Nations Plaza, New York, New York 10017
(International)