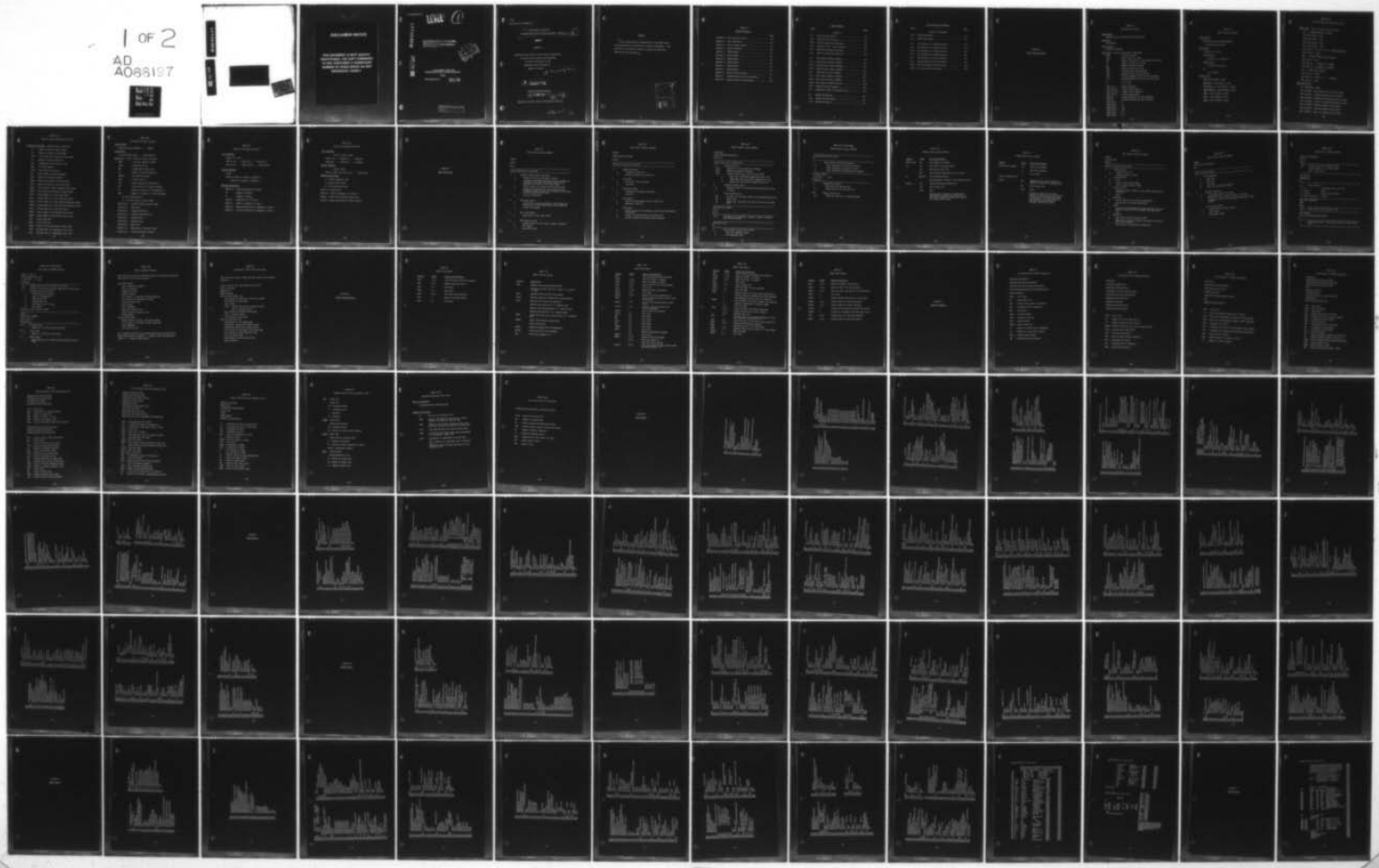
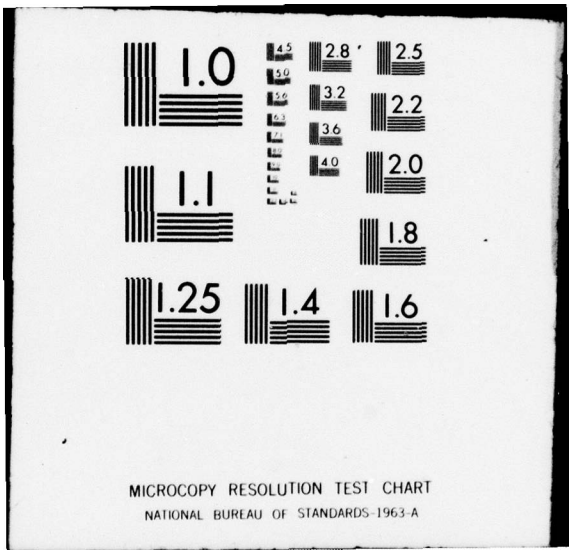
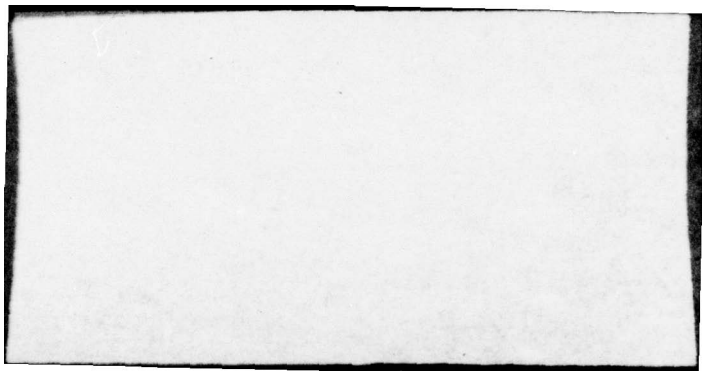
AD-A066 197 AIR FORCE INST OF TECH WRIGHT-PATTERSON AFB OHIO SCH--ETC F/G 5/8
A DEVELOPMENT SYSTEM FOR MICROPROCESSOR BASED PATTERN RECOGNIZE--ETC(U)
DEC 78 J R LEARY
UNCLASSIFIED AFIT/GCS/EE/78-12-VOL-2 NL

1 OF 2
AD
A066197




1.0
2.8
2.5
3.2
2.2
3.6
2.0
4.0
1.1
1.8
1.25
1.4
1.6
MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A



DISCLAIMER NOTICE

THIS DOCUMENT IS BEST QUALITY PRACTICABLE. THE COPY FURNISHED TO DDC CONTAINED A SIGNIFICANT NUMBER OF PAGES WHICH DO NOT REPRODUCE LEGIBLY.

AFIT/GCS/EE/78-12

LEVEL II

AD A066197

DDC FILE COPY

A DEVELOPMENT SYSTEM FOR MICROPROCESSOR BASED PATTERN RECOGNIZERS

THESIS

AFIT/GCS/EE/78-12

John R. Leary
Captain, USAF

(14)
AFIT/GCS/EE/78-12-Vol-2

(6)
A DEVELOPMENT SYSTEM FOR MICROPROCESSOR BASED PATTERN RECOGNIZERS. Volume II.

~~THESIS~~

VOLUME II,

Presented to the Faculty of the School of Engineering
of the Air Force Institute of Technology
in Partial Fulfillment of the
Requirements for the Degree of
Master of Science

(9) Master's thesis,

by

(10) John R. Leary, ~~A.B.~~
~~Captain~~ ~~USAF~~

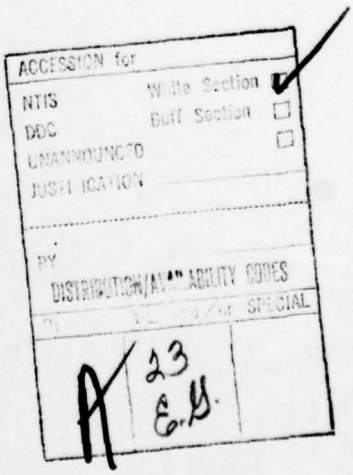
Graduate Computer Systems

(11) December 1978 (12) 147p

Approved for public release; distribution unlimited

Preface

This volume contains documentation for the BOX80 System. This documentation is presented in a series of appendices. Each contains a set of charts, tables, or program listings which describe the BOX 80 System.



Volume II

Table of Contents

List of Tables

List of Tables (Continued)

APPENDIX A

Data Structure Formats

TABLE A-I

FEAT Data File Format

HEADER BLOCK:

NAME,LABLF,JDX,LB,IC,MAXV,IOPT,IHIS,FIRS,FLAS

DATA BLOCK:

FEAT(JDX,LB)

TRAILER BLOCK:

FMIN(JDX),FMAX(JDX)

Definitions: (Implicit type unless indicated)

NAME	- File name, 4HFEAT
LABLF	- Label, nnnn user input
JDX	- Number of features +3, data buffer X-dimension
LB	- Number of fectors, per block
IC	- Number of pattern classes
MAXV	- Maximum vectors per block
IOPT	- Scaling option used by CREATE
IHIS	- Histogram option used by CREATE
FIRS	- Minimum component value over all dimensions
FLAS	- Maximum component value over all dimensions
j	- $1 \leq j \leq JD$
ℓ	- $1 \leq \ell \leq LB$
FEAT (j, ℓ)	- Vector component
FEAT (JD+1,ℓ)	- Integer vector number
FEAT (JD+2,ℓ)	- Integer class identifier
FEAT (JD+3,ℓ)	- Vector energy value
FMIN (j)	- Minimum component for this dimension
FMAX (j)	- Maximum component for this dimension
FMIN (JD+1)	- 1
FMAX (JD+1)	- 2
FMIN (JD+2)	- 98
FMAX (JD+2)	- 99
FMIN (JD+3)	- 0.
FMAX (JD+3)	- 0.

TABLE A-II

CLAS File Data Structure

HEADER RECORD:

NAME,LABLC,JDX,ICX,NTC,NBUC,MAXKV,

... NENT,NCIX,MSCAL,MZAPD

DATA RECORD:

CLAS(JDX,i), $1 \leq i \leq ICX$

VECTOR BLOCK:

CLAS(JDX,i), $1 \leq i \leq (NCIX-1)$

INDEX BLOCK:

CLAS(JDX,i),$NCIX \leq i \leq ICX$

OR

LIST (NENT,2)

VECTOR TYPES:

MEAN: CLAS(J,NCAV), $1 \leq j \leq JD$

DEVIATION: CLAS(j,NCSD), $1 \leq j \leq JD$

DEVIATION (L): CLAS(j,NCSDL), $1 \leq j \leq JD$

DEVIATION (R): CLAS(j,NCSDR), $1 \leq j \leq JD$

HUSK: CLAS (j,NPOS), $1 \leq j \leq JDX$

CMIN: CLAS (j,NCMIN), $1 \leq j \leq JD$

CMAX: CLAS (j,NCMAX), $1 \leq j \leq JD$

TABLE A-III

CLAS File Data Item Definitions (1/2)

VECTOR TAGS: (Type is implicit unless stated)

(A) i = NCAV,NCSD,NCSDL, OR NCSDR

CLAS (JD+1,NCAV) - FLAC

CLAS (DJ+1,NCSD) - FIRC

CLAS (DJ+1,NCSDL) - FIRC

CLAS (JD+1,NCSDR) - FLAC

CLAS (D+2,i) - KTYP (0, 1, or 2 ∿ NCAV,NCSDL,NCSDR)

CLAS (JD+3,i) - NEXC

(B) i = NCMIN OR NCMAX (LIMITS VECTORS)

CLAS (JD+1,i) - 0.0

CLAS (JD+2,i) - -1 (NCMIN), -2 (NCMAX)

CLAS (JD+3,i) - 98 ONCMIN), 99 (NCMAX)

(C) i = NPOS (HUSK VECTORS)

CLAS (JD+1,i) - NEXKE

CLAS (JD+2,i) - KTYP (3, 4 . . . (MAXKV+1)

CLAS (JD+3,i) - NEXC

ITEM DEFINITIONS:

(A) COMPONENTS, $1 \leq j \leq JD$

CLAS (j,NCAV) - Mean component value for the class

CLAS (j,NCSD) - Standard deviation value as above

CLAS (j,NCSDL) - Negative zero-based deviation as above

CLAS (j,NCSDR) - Positive zero-based deviation as above

CLAS (j,NCMIN) - Minimum component value overall

CLAS (j,NCMAX) - Maximum component value overall

CLAS (j,NPOS) - Vector id assigned to class husk

TABLE A-III

CLAS File Data Item Definitions (2/2)

VARIABLE DEFINITIONS: (Variable type is implicit)

IC - Number of FEAT file data classes

ICX - Number of columns in CLAS buffer

NCIX - First column of LIST area in CLAS buffer

JD - Number of feature dimensions

JDX - X-dimension of CLAS buffer = JD+3

NAME - File name, 4HCLAS

LABLC - Label nnnncc set by user

NTC - Total number of class prototypes

NBUC - Number of histogram intervals

MAXKV - Maximum value number of husk vectors

NENT - Number of entries in LIST index array

MSCAL - Flag indicates vector component value range

MZAPD - Flag indicates existence of zapped values

NCAV - Column number of a class mean vector

NCSD - Column number of a class deviation vector

NCSDL - Column number of a class negative deviation vector

NCSDR - Column number of a class positive deviation vector

NPOS - Column number of a class husk vector

NCMIN - Column number of the minimum values vector

NCMAX - Column number of the maximum values vector

KTYP - Vector type code

NEXC - Class identifier

NEXKE - Pointer to next open entry in husk vector

FIRC - Minimum value, all components, this class

FLAC - Maximum value, all components, this class

TABLE A-IV

HIST/DIST File Data Structure

HEADER RECORD:

NAME,LABEL,NHX,JD,IDUM(1), . . . IDUM(6)

DATA RECORDS:

NEXC,ISYM,KTR,NI,NBUC, . . . HISC (NHX,JD)

Definitions: (Type is implicit unless indicated)

NAME	- File name, 4HHIST or 4HDIST
LABLC	- nnnnmm set by user
NHX	- X-dimension of data array
JD	- Number of feature dimensions
IDUM(6)	- Six arbitrary values
NEXC	- Current class id
ISYM	- Switch indicates histogram pair
KTR	- First position of "right" histogram
NI	- Number of intervals if histogram pair
NBUC	- Number of intervals if single histogram

For the following items

K = 1 or KTR, $1 \leq j \leq JD$, $1 \leq n \leq NI$ or NBUC

HISC(K+0,j)	- Count of values processed, NUM
HISC(K+1,j)	- Mean value
HISC(K+2,j)	- Standard Deviation
HISC(K+3,j)	- Interval Containing Mode
HISC(K+4,j)	- Minimum value
HISC(K+5,j)	- Maximum value
HISC(K+6,j)	- Mode value
HISC(K+7,j)	- Mode count as fraction of NUM
HISC(K+n,j)	- Histogram interval counters

TABLE A-V

PROT File and Record Structure

FILE STRUCTURE:

$1 \leq j \leq jx$, $jx = JD+1$

PROT (1,1), . . . PROT (j,1), . . . PROT (jx,1)

PROT (1,IC), . . . PROT (j,IC), . . . PROT (jx,IC)

RECORD STRUCTURE:

$1 \leq i \leq IC$

ICLAS(i), CAV(1,i), CSOL(1,i), CSDR(1,i), . . .

. . . CAV(JD,i), CSDL(JD,i), CSDR(JD,i)

VARIABLE DEFINITIONS:

PROT(j,i) - Class (prototype) definition

JD - Number of features

IC - Number of classes

ICLAS(i) - Identifier for I^{th} class

CAV(j,i) - Mean for component J, Class I

CSDL(j,i) - Negative deviation for component J, Class I

CSDR(j,i) - Positive deviation for component J, Class I

TABLE A-VI

FVEC File and Record Structure

FILE STRUCTURE:

$1 \leq k \leq K$, $K = JD+2$, $1 \leq n \leq N$

FVEC(1,1), . . . FVEC(k,1), . . . FVEC(K,1)

FVEC(1,N), . . . FVEC(k,N), . . . FVEC(K,N)

RECORD STRUCTURE:

$1 \leq j \leq JD$, $1 \leq n \leq N$

IKNO (n), IVEC (n), FEAT (j,n), . . . FEAT (JD,n)

VARIABLE DEFINITIONS:

j - Feature component index

n - Feature vector index

k - FVEC record item index

FVEC(k,n) - Record item

FEAT(j,n) - Feature component j of vector n

IKNO(n) - Record item indicates known class

IVEC(n) - Record item identifies feature vector

APPENDIX B

User Input Guide

TABLE B-I

DEFC Control Options (CREATE)

CREATE

ENTER

*abcd

Entries abcd may be in any order		
a		Selects feature vector transforms
	U	feature vectors are unitized for output
	S	a minimum and maximum component value is obtained for the entire input data set; output feature vector components are scaled to this value range
	E	feature vectors are energy normalized for output
	Q	minimum and maximum values are obtained for each component; output feature vector components are scaled to the associated value range
	N	no vector transforms
b		Histogram control
	H	a HIST File is output containing a distribution and statistics for each input feature vector component
	Δ	no HIST file output
c		User input option
	n	$0 \leq n \leq 9$ input to user read routine
d		LOGF control switch
	L	Journal entries of user inputs, summary statistics
	Y	Exit traces
	T	Data buffer dump

TABLE B-II

DEFD Control Options (DEFINE)

DEFINE

ENTER MODE AND OPTIONS

*a,bcde

Entries b-e can be in any order

---

a		Mode of operation
	I	initialize a CLAS file
	F	follow-on process of a CLAS file
b		Print file
	P	Print CLAS file at end job
	Δ	no action
c		Reallocate memory
	E	Extend CLAS file memory allocation
	Δ	no action
d		Save option
	S	Writes newly generated file over input file
	Δ	Generates a new file
e		LOGF Switch
	L	Journal record of file headers, control inputs and memory allocations
	Y	Traces file manipulations and routine exits
	T	Records content of each FEAT record processed

TABLE B-III

NEXCLA Control Inputs (DEFINE)

```
NEXT-CLASS
ENTER FUNCTION AND CLASS ID
*aaa,b,±cc
```

Entries a,b,c are in fixed order

a		Function control
	REFN	generate or revise symmetric prototype
	DEFN	generate asymmetric prototype
	Kdef	kernel options (d,e,f can be in any order)
	c	reset husk with program calculated vector ids
	p	reset husk with interactively picked vector ids
	s	update husk with manually selected vector ids
b		Histogram option control
	H	generate a HIST file record while processing this class
	Δ	no action
c		Class identification
	+NN	use husk list to reject vectors from prototype generation
	-NN	ignore husk
	OO	ignore husk, initialize full CLAS file from all of FEAT file

```
ENTER MAX SIZE KERNEL
XX.XX
*aa.aa
```

aa.aa	Estimate of class boundary 'diameter' (Used in automatic husk selection stub)

```
QUICK PLOT (Y/N)
*a
```

a	Class features cluster plot control
Y	Plot only component points
N	Plot connected lines

TABLE B-III (Continued)

NEXCLA Control Inputs (DEFINE)

```
ENTER SCALING OPTION (U,C,M)
*a
```

a	Class features cluster plot control
U	Scale components to max-min of data set
C	Scale components to max-min of class
M	Scale components with Mahalanobis' Distance

```
SET HUSK VECTOR
A, NV, NV, . . . (15)
*a,bbb,bbb,. . .
```

a	Insert/Delete Control
A	Add feature ids to husk list
D	Delete feature ids from husk list
bbb	Feature selector id
123	Identifiers must be in increasing order

TABLE B-IV

SUBSET Control Inputs (TRYOUT)

Request	Entry	Description/Effect
ENTER CLASS	00	Quit SUBSET; end TRYOUT
+CL	88	Quit SUBSET: continue TRYOUT
*	98	Reinitialize CLAS file
	+99	Zap specified dimensions for all classes
CC	1≤CC≤NTC	(No. Prototype Classes)
		Zap Dimensions as above (±) for specified class
DD,DD,. . .	NN	1≤NN≤JD (No Feature Dimensions)
	99	End of input list
	Else	Error
		NN specifies a component or dimension of the feature/prototype space which will be nullified or zapped for either one class or all classes per CL entry

TABLE B-V

FIGM Control Inputs (TRYOUT)

Request	Entry	Description/Effect
ENTER F/M SET NUMBER	1,2	Select F/M subspace
	0	User sets subspaces
	3	Exit tryout
SPECIFY SUBSPACE TAGS		
KK,KK, . . .	NN	1≤NN≤JD (JD=No. Feature Dimensions) identifies dimension to consider next
	99	End of input list
	Else	Error
		The order in which KK are input defines the order (and so the content) of the set of nested subspaces which are evaluated. A 99 entry indicates that the remaining unspecified subspaces may have any order.

TABLE B-VI

DEFT Control Options (TRYOUT)

TRYOUT

ENTER OPTIONS

*abcdef

Entries a-f can be in any order

a		Summary Answer
	A	overall error rate
	C	confusion matrix on LOGF
b		Scaling Options
	N	none
	B	scale to byte (0-255) range
	W	scale to word (0-32767) range
c		Figure of Merit
	F	calculate a merit figure for each feature dimension for evaluation
	Δ	none
d		Magnitude
	M	use Euclidean norm in distance measurements
	Δ	use Sup norm in distance measurements
e		Subset
	S	allow user to zap dimensions of the prototype space and generate a new CLAS file for FORMAT with these zaps.
	Δ	none
f		Log Switch
	L	logs certain process results on LOGF
	Y	logs exits of critical routines and contents of certain buffers when changed
	T	logs results of intermediate calculations

TABLE B-VII

DEFT Control Options (FORMAT)

```
FORMAT
ENTER FILE AND FUNCTION
*abc
```

Enter a-c in any order

---

a File identifier

- F FEAT file
- C CLAS file
- H HIST file
- D DIST file (produced by CREATE)

b Function identifier

- S strip chart output of any data
- X hexadecimal format output of FEAT or CLAS data
- P picture of a set of CLAS prototype boundaries or FEAT vectors

c LOG function control

- L journal entries of user inputs
- Y exit traces
- T data buffer dumps

TABLE B-VIII

User Inputs to FORMAT Routines

ENTER PLOT PARAMETERS
ICPS,G
*aaa,b

aaaa	Terminal data rate characters/second
b	Picture control 'G' → gridwork plot else → strips only.

NEXT-RECORD
NN-RECORD ID
*aa

aa	Record number, n	
n = 0		n sequences from 1 to last n
n < n-prior		error
n ≥ n-prior		process stated record
n > last n-available		end job

ENTER XMIT PARAMETERS
AAAA
*aaaa

aaaa	Address (Hex) for output data lines

PLOT FOLLOWS
TYPE G TO CONTINUE NOW AND LATER
*a

a	G continues process. When cursors appear, a blank repeats plot.

TABLE B-VIII (Continued)

User Inputs to FORMAT Routines

```
CLASS = a, RB = b
PICK VECTORS FROM c TO d
K, VVV, VVV, . . . (15)
*e,fff,ggg, . . .
```

a,b	Identify data class and current record block
c,d	Identify first and last vectors remaining in current block
e	Selection option for fectors
I	within interval (fff,ggg)
L	from list fff,ggg,hhh, . . .
C	chosen until fff
S	skipped until fff
E	exit this class of vectors
Q	quit FORMAT *

(Options I, and L are not coded)

```
CLASS = a
ENTER SCALE CONTROLS
BB.B, ZZ.Z, EL.
*aa.a,bb.b,cc.
```

aa.a	Base grid units
n	number of x-y units per base grid block
bb.b	Z-axis scale
n	number of z units per z-axis step
cc.	Viewing angle
n	angle between the x-y plane and the observer's line of sight

TABLE B-IX

TAPEIN Execution Procedure

(This procedure assumes an SBC 80/20 system with 16K RAM and HAZELTINE 2000 terminal with cassette recorder.)

Load TAPEIN module

- Insert Bootstrap Cassette*
- Power up recorder
- Engage Drive Head
- Rewind Cassette
- Set recorder to continuous offline page mode
- Set terminal to batch mode, 110 band
- Play back header file
- Clear screen
- Enter I3850(CR)
- Play back execution file
- Enter 'escape' character at lot
- Shift/transmit

Execute TAPEIN Module

- Ready recorder as above, using data cassette
- Set recorder to continuous, online, page mode
- Enter G3850(CR)
- Play back data file

N.B. Bootstrap cassette can be constructed from the listing of the TAPEIN module given in Appendix I. Reference INTEL SBC 80/20 User Guide for 'I' command format details.

TABLE B-X

Generating a PROT File Cassette Tape

(This procedure assumes a HAZELTINE 2000 Terminal with Cassette Recorder.)

Insert Cassette (use right-hand drive at AFIT!)
Power up recorder
Engage drive
Rewind cassette
Write cassette header file
    Set recorder to continuous, offline, page mode
    Set terminal to batch mode
    Clear screen
    Enter: control-shift-period, control-shift-oh
        (use no carriage return)
    Set recorder to record/interlock mode
    Enter: carriage return
    Set recorder to reset mode
Write cassette data file
    Set terminal to full duplex mode
    ATTACH,BYTE (produced by FORMAT module)
    COPYBF,BYTE,OUTPUT (no carriage return)
    Set recorder to continuous, online page mode
    Set recorder to record/interlock mode
    Enter carriage return
    Set recorder to reset mode (at eot)
    Rewind cassette

APPENDIX C

Journal Output Summary

TABLE C-I

CREATE LOGF Outputs

Routine	LOGSW	Output and Conditions
DEFA	L,T,Y	Initialization request and response
DEFA	L,T,Y	Memory allocation record
DEFA	Y,T	Exit trace
SCAN	T,Y	Exit trace
SCAN	L,T,Y	First pass scan statistics
COPY	T	Dump of input data buffer
COPY	Y,T	Exit trace

TABLE C-II

DEFINE Terminal Outputs

Routine	Description
DEFD	Requests processing mode and options
	Identifies initial CLAS file if mode 'I' or option 'E' used
OPENX	Identifies CLAS and FEAT files opened
ALLOC	Requests approval of memory/file size parameters
NEXCL	Requests next class id and function
	Requests max. size husk if 'C' function used
	Requests plot scale options if 'P' function used
	Requests husk entries if 'S' function used
PHUSK	Prints husk entries for current class if 'S' function used
FANDER	Labels class features cluster plot
	Draws cluster plot
WRCLAS	Identifies output CLAS file parameters
DEFINE	Prints end of job error messages
ERR	Prints error prompt ('?')

TABLE C-III

DEFINE LOGF Output

Routine	LOGSW	Output and Conditions
DEFD	L	Log of Operating mode and options
OPENX	L,T,Y	FEAT file header if opened
OPENX	L,T,Y	CLAS file header if opened
DEFD	L,T,Y	Log of initial CLAS file parameters
DEFD	T,Y	Exit trace
ALLOC	L,T,Y	Record of memory allocation
ALLOC	T,Y	Exit trace
NEXCLA	L,T,Y	Record of control parameters set
PHUSK	--	List of Husk entries for current class if requested
KERPUT	T,Y	Trace of file manipulations as husk is set
CLASSX	T,Y	Trace of CLAS file pointers for current prototypes
CLASSX	T	List of FEAT file contents
CLASSX	T,Y	Exit trace
ADD	T,Y	Exit trace
DEL	T,Y	Exit trace
RFEAT	T,Y	Exit trace
RIX	T,Y	Exit trace
FANDER	T,Y	Exit trace
CDEFI	T,Y	Exit trace
	Y,L	Record of generated prototype
KERGET	T,Y	Exit trace
WRHIS	L,T,Y	Record of HIST file header
	L,T,Y	HIST file record id
	Y	HIST file record content
PRCLAS	L,T,Y	CLAS file dump (This dump can be forced by initial option 'P')

TABLE C-IV

TRYOUT LOGF Outputs

Routine	LOGSW	Output and Conditions
DEFI	L,T,Y	Echo of initiation requests and responses
OPENX	L,T,Y	FEAT file header, if opened
OPENX	L,T,Y	CLAS file header, if opened
DEFT	T,Y	Exit trace
INDEX	L,T,Y	Index to CLAS file
INDEX	L	CLAS file dump if scale transform
INDEX	T,Y	CLAS file dump
MERIT	T	Distance vectors for each unque prototype pair
	L	Average interclass distances for each prototype
	T	Intermediate figures of merit for features
SUBSET	Y	Echo of SUBSET initiation requests and responses
	L	Summary by prototype of features zapped
	Y,T	CLAS file dump
FIGM	L,Y,T	Figures of merit for feature dimensions
	Y,T	Echoes specified set of merit figures
	L,Y,T	Subspace tags list
EVAL	T	Distance vector to selected prototype for first five vectors in each FEAT block
EVAL	Y,T	Feature vectors (first five) of each FEAT block
DOCU	L,T,Y	Summary performance percentages
DOCU	--	Confusion matrix if 'C' user option
DOCU	L,T,Y	Subspace tags and error rates if 'F' user option
PRCLAS	L	CLAS file dump
RFEAT	T,Y	Exit trace

TABLE C-V

FORMAT LOGF Outputs

Routine	LOGSW	Output Description
DEFA	L,Y,T	Record of FORMAT initialization
OPENH	L,T,Y	Record of HIST file initialization
OPENX	L,T,Y	Record of FEAT, and CLAS file initialization
INDEX	T	Exit trace
INDEX	L	Format printout of CLAS file if re-scaled
NEXREC	L,Y,T	Journal entry for record processed
NEXVEC	L,Y,T	Journal entry for vectors selected
FILBUF	L	Printout of statistics from HIST file record
FILBUF	Y	Printout of histograms from HIST file record
PICT	L,T,Y	Journal entry for 3-D plot parameters
RHIST	L,T,Y	Journal entry for each HIST record

APPENDIX D

Module Parameters

TABLE D-I

I/O Parameters for CREATE Subroutines

DEFC(IEOJ,NW,JDK,NHX)

SCAN(BUF,HISG,VECS,IEOJ,NW,NHX)

COPY(BUF,FVEC,HISG,IEOJ,NW,JDX,NHX,VECS)

GETFEA(IUSER,BUF,NW,FVEC,JD,IC,NVEC,IGC,IEOJ)

PRHIST(HISG,NHX)

IEOJ - Error flag

NW - User buffer size

JDX - Feature vector buffer X dimension

NHX - Histogram buffer X dimension

BUF - User buffer

HISG - Histogram buffer

VECS - Temporary work area

FVEC - Feature vector

IUSER - Name of user file

JD - Number of feature vector components

IC - Number of a priori data classes

NVEC - Feature vector number

IGC - Option switch from terminal

TABLE D-II

I/O Parameters for DEFINE Subroutines

DEFD(IEDJ)

ALLOC(KC,NE,HUSKR,IEOJ)

NEXCLA(FEAT,CLAS,JDX,IEOJ)

CLASSX(FEAT,CLAS,HISC,JDX,NHX,IEOJ)

CDEFI(FEAT,CLAS,HISC,JDX,NHX,IEOJ)

FANDER(CLAS,JDX,FEAT,IEOJ)

PHUSK(CLAS,JDX,IEOJ)

KERPUT(CLAS,JDX,NEXC,IDOJ)

SETUM(CLAS,FEAT,JDX,IEOJ)

IEOJ - Error flag

KC - Number of a priori data classes

NE - Number of open CLAS file vectors

HUSKR - Greatest fraction of any class allowed as husk

FEAT - Feature vectors data block

CLAS - Prototype file buffer

JDX - Feature vector buffer X dimension

HISC - Histogram file buffer

NHX - Histogram buffer X dimension

NEXC - Class being processed

TABLE D-III

I/O Parameters for TRYOUT Subroutines

DEFT(IEOJ)

MERIT(FT,DISI,DISN,CLAS,JDX)

SUBSET(CLAS,JDX,ICX,IEOJ)

FIGM(FT,JDX,IEOJ)

EVAL(FEAT,CLAS,CMX,JDX,NAX,IC,IEOJ)

LOOK

DOCU(CMX,CLAS,JDX,NAX,IC)

IEOJ - Error flag

FT - Work area contains feature merit figures

DISI - Work area:interprototype distance to I^{th} prototype

DISN - Work area: interprototype distance to N^{th} prototype

CLAS - Prototype file buffer

JDX - X dimension for feature and prototype vector buffers

ICX - Y dimension for feature and prototype vector buffers

CMX - Confusion matrix buffer

NAX - Confusion matrix Y dimension = NTC+1

IC - Number of a priori classes

Table D-IV

I/O Parameters for FORMAT Subroutines

```
DEFF(IEOJ)
XCLAS(CLAS,BUF,HID,JDX,NBX,IEOJ)
XFEAT(FEAT,BUF,HID,CLAS,JDX,NBX,IEOJ)
XHIST(HIST,BUF,HID,NHX,NBX,IEOJ)
XMIT
PICT(BUF,HID,NX,NY)
STRIP(BUF,NX,NY)
FILBUF(HIST,BUF,NHX,JI,JN,NX,NY,ICON)
NEXREC(IEOJ)
NEXVEC(ISEQ,NVEC,NRB,IEOJ)
```

IEOJ - Error flag
CLAS - CLAS file buffer
BUF - PLOT buffer
HID - Hidden line buffer for PLOT3D
JDX - X dimension of CLAS, and FEAT buffers
NBX - X dimension of PLOT buffer
FEAT - FEAT file buffer
NHX - X dimension of HIST file buffer
NX - X dimension of PLOT buffer
NY - Y dimension of PLOT buffer
HIST - HIST file buffer
JI - Starting component number for buffer fill
JN - Ending component number for buffer fill
ICON - Switch indicates HIST record symmetry-type
ISEQ - Initialization switch
NVEC - Selected vector number
NRB - Feature record block number = NBPR

TABLE D-V

Utility Routine Calling Parameters (1/2)

```
ADD(NAME,LIST,NENT,NPOS,IEOJ)
DEL(NAME,LIST,NENT,NPOS,IEOJ)
INITC(NENT,LIST,IADD)
RIX(NAME,LIST,NENT,NPOS,IEOJ)

IEOJ  - Error Flag
NAME  - Identifier for requested vector
LIST  - CLAS file index area
NENT  - Number of entries in LIST
NPOS  - Pointer to requested vector
IADD  - Switch allows added index initialization

STATH(LAS,FIN,FT,FHIS,NI,FIRS,FLAS)
STATX(LAS,FIN,FT,CAV,CSD,ISW)
XSCAL(FMIN,FMAX,VEC,NN,SCALE,IOP)

LAS   - Switch triggers final calculations
FIN   - Input value
FT    - List of 8 statistics (see code)
FHIS  - Array of containing histogram
NI    - Number of histogram intervals
FIRS  - Least value in data range
FLAS  - Greatest value in data range
CAV   - Mean of input data stream
CSD   - Standard deviation of input data
ISW   - Switch selects deviation algorithm
FMIN  - Vector of minimum component values
FMAX  - Vector of maximum component values
VEC   - Input vector
NN    - Order of input vector
SCALE - Requested output value range
IOP   - Switch selects scaling algorithm
```

TABLE D-V

Utility Routine Calling Parameters (2/2)

```
INDEX(CLAS,JDX,ICX,IEOJ)
KERGET(CLAS,JDX,NEXC,IEOJ)
PROCLAS(CLAK,LIST,CLAS,JDR,ICX)
LOADC(CLAS,JDX,ICX,IEOJ)
OPENH(LABLH,IEOJ)
OPENX(IMOD,IE,LABLC,LABLF,IEOJ)
RHIST(HISC,NHX,IEOJ)
RFEAT(FEAT,JDX,INIT,IEOJ)
WRCLAS(CLAS,JDX,ICX,IS,LABLC)
WRHIS(HISC,HEXC,ISYM,LABLC,NBUC,KTR,NI,NHX,JDX)
```

CLAS - Prototype data file buffer
JDX - X dimension, number of components +3
ICX - Y dimension, number data vectors in CLAS
IEOJ - Error flag
NEXC - Data class indicator
CLAK - Prototype data file for integer reference
LIST - Prototype data file index
LABLH - HIST file label
IMOD - Mode switch, controls opening of FEAT file
IE - Extend switch, controls opening of CLAS file
LABLC - CLAS file label
LABLF - FEAT file label
HISC HIST file buffer
NHX - X dimension, number of intervals +8
FEAT - FEAT file buffer
INIT - Switch controls FEAT file positioning
IS - CLAS file header parameter
ISYM - Flag indicates histogram pair
NBUC - Number of histogram intervals
KTR - First value of second set of statistics
NI - Number of intervals if histogram pair exists

TABLE D-VI

Support Routine Calling Parameters (1/2)

```
MARK(IX,IY,IDX,IDY)
ENER(VEC,N)
PLOT3D(FCN2D,HID,IMAX,JMAX)
PLX(X,Y,II)
ILINE
IASORT(IN,NN)
FDSORT(FIN,ITAG,FOUT,NN)
```

IX - X-position of mark in screen units
IY - Y-position of mark in screen units
IDX - X-movement for mark
IDY - Y-movement for mark
VEC - Array of components of vector
N - Order of vector
FCN2D - Array of surface Z values
HID - Work area
IMAX - X-dimension of FCN2D
JMAX - Y-dimension of FCN2D
X - Pen position in inches
Y - Pen position in inches
II - Control switch for pen up/down/origin
IN - Vector of integer inputs
NN - Order of input vector
FIN - Array of real input values
ITAG - Array of value "tags"
FOUT - Array of real output values

TABLE D-VI

Support Routine Calling Parameters (2/2)

DIV Input BC,D

Output C,B

BC - Unsigned dividend

D - Unsigned divisor

C - Quotient

B - Remainder

BNBCD Input DE,HL

Output BCD characters

DE - Unsigned integer

HL - Pointer to lowest buffer location

CLOOP Input B,HL

Output HL,B,A, character print

B - Number of characters

HL - Pointer to lowest character in string

A,B,HL - Destroyed on output

OUTB Input HL,DE,B

Output Typewriter print

HL - Address of values list

DE - Address of putput line

B - Length of output line

TABLE D-VII

SBC 80/20 Hexadecimal Data Format

DATA LINE VARIABLES:

LS,LNUM,LADR,LTYPE,LBYTE(16),LSUM

VARIABLE DEFINITIONS:

LS - Start of line character (1H:)

LNUM - Count of the number of data bytes to follow, encoded in hexadecimal notation (A2)

LADR - Address of the storage location for the first data byte, encoded in hexadecimal notation (A4)

LTYP - Line type (00-data line; 01-end record line)

LBYTE - A set of up to 16 data values, each represented as a hexadecimal number (A2)

LSUM - A checksum, in hexadecimal of the sum 255 - $\sum_{i=1}^{I} K_i$ where one K_i represents each 2-character hexadecimal value following the start of line character (1H:)

TABLE D-VIII

User Input Subroutine Parameters

GETFEA(IUSER,BUF,NW,FVEC,JD,IC,NVEC,IGC,IEOJ)

IUSER - Name of user input file

BUF - Address of input buffer

NW - Number of words allotted input buffer

FVEC - Feature vector passed to system from module

JD - Number of feature dimensions

IC - Number of pattern classes

NVEC - Identifier for FVEC within its class

IGC - Input control switch

IEOJ - Control flag

APPENDIX E

CREATE Module

```
100=      PROGRAM CREATE(USER,FEAT,DIST,LOGF=64,INPUT=64,OUTPUT=64)
110=C**         JULY 27 78   JRL
120=      COMMON WO(8200)
130=      COMMON/NAMES/INPF,IOUTF,IFEAT,IUSER,IDUM,ILOGF,IDIST
140=      COMMON/CREA/IOPT,IHIS,ISCAN,NHI,NFI,NLI,IGC,
150=     1    LABLF,JD,LB,IC,MAXV,NBUC,NTC,KEM,FIRS,FLAS,IU
160=      DATA NEXC,ISYM,KTR,NI/1,1,0,0/
170=C**********
180=      MEM=8200
190=      DO 2 M=1,MEM
200= 2    WO(M)=0
210=      CALL DEFA(IEOJ,NW,JDI,NHX)
220=      IF(IEOJ.GT.0) GOTO 10
230=      IF(ISCAN.GT.0) CALL SCAN(WO(1),WO(NH1),WO(NL1),IEOJ,NW,NHX)
240=      IF(IEOJ.GT.0) GOTO 10
250=      CALL COPY(WO(1),WO(NF1),WO(NH1),IEOJ,NW,JDX,NHX,WO(NL1))
260=      IF(IEOJ.GT.0) GOTO 10
270=      IF(IHIS.GT.0)
280=     1    CALL WRHIS(WO(NH1),NEXC,ISYM,LABLF,NBUC,KTR,NI,NHX,JDX)
290=      STOP
300= 10   CONTINUE
310=      WRITE(IOUTF,900)IEOJ
320= 900  FORMAT(7H ABEND ,I3)
330=      STOF
340=      END
350=
```

```
360=
370=      SUBROUTINE DEFA(IEOJ,NN,JDX,NHX)
380=      COMMON/TRACE/ILOG,ITRACE
390=      COMMON/NAMES/INPF,IOUTF,IFEAT,IUSER,IDUM,ILOGF,IDIST
400=      COMMON/CREA/IOPT,IHIS,ISCAN,NH1,NF1,NL1,IGC,
410=     1    LABLF,JD,LB,IC,MAIV,NBUC,NTC,MEM,FIRS,FLAS,IU
420=      DIMENSION KOPS(4)
430=      DATA IHB,IHS,IHU,IHE,IHH,IHY,IHT/1H ,1HS,1HU,1HE,1HH,1HY,1HT/
440=      DATA IHL,IHQ,IHI,IHJ,IHN/IHL,IHQ,IHI,IHJ,IHN/
450=      DATA IH1,IH2,IH3,IH4,IH5,IH6,IH7,IH8,IH9/
460=     1IH1,IH2,IH3,IH4,IH5,IH6,IH7,IH8,IH9/
470=C              INITIALIZE PARAMETERS
480=      IUSER =4LUSER
490=      IFEAT=4LFEAT
500=      IDIST=4LDIST
510=      ILOGF=4LLOGF
520=      INPF=5LINPUT
530=      IOUTF=6LOUTPUT
540=      ITRACE=6LOUTPUT
550=      IOPT=0
560=      ILOG=0
570=      IHIS=2
580=      NN=4
590=      ISCAN=0
600=C              GET OPTIONS
610=C           ISCAN-->CALC (FIRS,FLAS) FOR SHIFT(S),AND
620=C           TRANSFORM(T) OPTIONS, AND FOR FEAT FILE
630=      WRITE(IOUTF,900)
640= 900  FORMAT(7H CREATE/14H ENTER OPTIONS/2H *)
650=   5  READ(INPF,901)KOPS
660= 901  FORMAT(4A1)
670= 911  FORMAT(2I,4A1)
680=      DO 8 K=1,4
690=      IF(KOPS(K).EQ.IHT) ILOG=1
700=      IF(KOPS(K).EQ.IHL) ILOG=3
710=      IF(KOPS(K).EQ.IHY) ILOG=2
720=      IF(KOPS(K).EQ.IHQ) IOPT=4
730=      IF(KOPS(K).EQ.IHI) IOPT=5
740=      IF(KOPS(K).EQ.IHS) IOPT=1
750=      IF(KOPS(K).EQ.IHU) IOPT=3
760=      IF(KOPS(K).EQ.IHE) IOPT=2
770=      IF(KOPS(K).EQ.IH1)IGC=1
780=      IF(KOPS(K).EQ.IH2)IGC=2
790=      IF(KOPS(K).EQ.IH3)IGC=3
800=      IF(KOPS(K).EQ.IH4)IGC=4
810=      IF(KOPS(K).EQ.IH5)IGC=5
820=      IF(KOPS(K).EQ.IH6)IGC=6
830=      IF(KOPS(K).EQ.IH7)IGC=7
840=      IF(KOPS(K).EQ.IH8)IGC=8
850=      IF(KOPS(K).EQ.IH9)IGC=9
860=      IF(KOPS(K).EQ.IHH)IHIS=1
870=      IF(KOPS(K).EQ.IHN)IHIS=0
880= 8    CONTINUE
890=      IF(IOPT.EQ.1.OR.IOPT.EQ.4) ISCAN=1
900=      IF(IHIS.EQ.1)ISCAN=1
910=      FIRS=+1E8
920=      FLAS=-1E8
930=      IF(ISCAN.EQ.1.OR.IOPT.NE.2.OR.IOPT.NE.3) GOTO 9
940=      FIRS=-1.
950=      FLAS=+1.
960= 9    CONTINUE
970=      IF((IOPT+ISCAN).NE.0.AND.IHIS.NE.2) GOTO 10
980=      IEOJ=-1
990=      CALL ERR(IOUTF,NN,IEOJ)
1000=     IF(IEOJ.GT.0) RETURN
1010=     GOTO 5
```

```
1020=C            GET PARAMETERS
1030= 10   CONTINUE
1040=      IF(ILOG.NE.0)WRITE(ILOGF,900)
1050=      IF(ILOG.NE.0)WRITE(ILOGF,911)KOPS
1060=      NN=4
1070=      WRITE(IOUTF,902)
1080= 902  FORMAT(17H ENTER PARAMETERS/
1090=     1  27H  LABL,NBUF,JD,IC,NI,NE,MXV/2H *)
1100= 15   READ(INPF,903) LABLF,NW,JD,IC,NBUC,NEC,MAXV
1110= 903  FORMAT(I4,1X,I4,4(1X,I2),1X,I3)
1120=      IF(JD.GE.1.AND.NBUC.GE.0.AND.NW.GT.JD.
1130=     1 AND.NEC.GE.0.AND.IC.GE.1.AND.IC.LE.30) GOTO 16
1140=C            ERROR INPUT
1150= 14   IEOJ=-2
1160=      CALL ERR(IOUTF,NN,IEOJ)
1170=      IF(IEOJ.GT.0) RETURN
1180= 16   CONTINUE
1190=C            SET CONTROLS
1200=      JDX=JD+3
1210=      MEMC=2*IC*JDX
1220=      MEMCS=0
1230=      IF((NW-MEMC).LT.0) MEMCS=IABS(NW-MEMC)
1240=      MTOT1=MEM-200-NW-(8*JD)-MEMCS
1250=      NV=MTOT1/JDX
1260=C       *NV SPLITS INTO HISTO INTERVALS(NBUC,MIN=10)
1270=C       *AND TO FVEC BLOCK SIZE (LB,MIN=10)
1280=C       *MTOT1=REM(WORK-(SCR(200)-BUF(NW)-DIST(8,JD)-CSAVE) )
1290=C       *REDUCE LB,THEN NBUC,THEN NEC
1300= 20   LB=NV-NBUC
1310=      IF(LB.GT.10) GOTO 27
1320=      IF(NBUC.GE.10) GOTO  27
1330=      IF(NV.LT.NEC) GOTO 30
1340=      IF(IHIS.EQ.1) GOTO 25
1350=      IF(LB.LT.10) GOTO 30
1360=      WRITE(IOUTF,907)
1370= 907  FORMAT(15H  MEM TOO SMALL)
1380=      GOTO 14
1390=C          RESET NBUC SINCE NOT ENOUGH ROOM LEFT
1400= 25   IHIS=0
1410=      NBUC=0
1420=      ISCAN=IOPT+IHIS
1430=      GOTO 20
1440=C          RESET NEC SINCE NOT ENOUGH ROOM
1450= 27   IF((LB-10).GE.NEC) GOTO 29
1460=      NEC=LB-10
1470=      LB=10
1480=      GOTO 30
1490= 29   LB=LB-NEC
1500=      IF(LB.GT.MAXV) LB=MAXV
1510= 30   CONTINUE
1520=C          PRINT ADJUSTED PARAMETERS
1530=      IF(ILOG.NE.0)
1540=     1 WRITE(ILOGF,904)KOPS,LABLF,NW,JD,IC,NBUC,NEC,MAXV
1550=      WRITE(IOUTF,904)KOPS,LABLF,NW,JD,IC,NBUC,NEC,MAXV
1560= 904  FORMAT(29H  APPROVE ADJUSTED PARAMETERS/,
1570=     1 31H OPTS LABL,NBUF,JD,IC,NI,NE,MXV/,
1580=     2 1X,4A1,2X,I4,1X,I4,4(1X,I2),1X,I3/
1590=     3 14H  ENTER Y OR N/2H *)
1600=      READ(INPF,905)IOK
1610= 905  FORMAT(A1)
1620=      IF(IOK.NE.IHY) GOTO 10
1630=C               SET CONTROLS
1640=      NTC=IC+NEC
1650=      MEMF=JDX*LB
1660=      NHX=8+NBUC
1670=      MEMH = NHI*JD
1680=      NH1=1+NW
1690=      NF1=NH1+MEMH
1700=      NL1=NF1+MEMF
1710=C               ECHO SPACE ALLOCATIONS
1720=      IF(ILOG.NE.0)WRITE(ILOGF,906)MEMF,LB,MEMH,NBUC
1730=      WRITE(IOUTF,906) MEMF,LB,MEMH,NBUC
1740= 906  FORMAT(16H  FEAT FILE HAS ,I5,10H WORDS AND,I5,
1750=     1 10H VECTORS PER BLOCK/
1760=     2 18H  DIST FILE HAS ,I5,10H WORDS AND,I3,
1770=     3 20H STEPS PER HISTOGRAM)
1780=C**             TRACE DEFA
1790=      IF(ILOG.NE.0.AND.ILOG.LE.2)
1800=     1   WRITE(ITRACE,957)NTC,MEMF,NHI,MEMH,NH1,NF1,NL1,LB,
1810=     2     NEC,ISCAN,IOPT,IHIS,NBUC,NV,JDX
1820= 957  FORMAT(8H  DEFAX ,2015)
1830=      RETURN
1840=      END
```

```
3190=
3200=      SUBROUTINE SCAN(BUF,DIST,VECS,IEOJ,NW,NHX)
3210=      COMMON/TRACE/ILOG,ITRACE
3220=      COMMON/NAMES/INPF,IOUTF,IFEAT,IUSER,IDUM,ILOGF,IDIST
3230=      COMMON/CREA/IOPT,IHIS,ISCAN,NH1,NF1,NL1,IGC,
3240=     1    LABLF,JD,LB,IC,MAXV,NBUC,NTC,MEM,FIRS,FLAS,IU
3250=      DIMENSION BUF(NW),DIST(NHX,1),VECS(1)
3260=C           CALCS NUM,AV,SD,MIN,MAX; SETS FIRS,FLAS
3270=      LAS=0
3280=      J1=JD
3290=      J2=2*JD
3300=C           VECS HOLDS FVEC,THEN CAV,THEN CSD TEMPORARILY
3310= 5    CONTINUE
3320=      JFIL=1
3330=      CALL GETFEA(IUSER,BUF,NW,VECS(1),JD,IC,NVEC,IGC,IEOJ)
3340=      IF(NVEC.GT.MAXV)MAXV=NVEC
3350=      IF(IEOJ.EQ.1) LAS=1
3360=      DO 10 J=1,JD
3370=      J4=J1+J
3380=      J5=J2+J
3390=      CALL STATX(0,VECS(J),DIST(1,J),VECS(J4),VECS(J5),0)
3400=      IF(LAS.NE.1) GOTO 10
3410=      CALL STATX(1,0.0,DIST(1,J),VECS(J4),VECS(J5),0)
3420= 10   CONTINUE
3430=      IF(IEOJ.LT.1) GOTO 5
3440=C           ASSEMBLE STATS INTO DIST
3450=C           AND GENERATE GLOBAL MIN,MAX VALUES
3460=C           NUM,AV,SD,NBUC,MIN,MAX,MODE,VMOD
3470=C             1  2  3   4  5  6   7   8
3480= 20   CONTINUE
3490=      IEOJ=0
3500=C           FIRS,FLAS ARE USED TO SET HISTO RANGE
3510=C           AND TO SHIFT FVEC COMPONENTS POSITIVE FOR
3520=C           CONVENIENT FIRST OCTANT PLOTS AND EASY SCALING
3530=      IF(LB.GT.MAXV)LB=MAXV
3540=      DO 30 J=1,JD
3550=      IF(DIST(5,J).LT.FIRS) FIRS=DIST(5,J)
3560=      IF(DIST(6,J).GT.FLAS) FLAS=DIST(6,J)
3570= 30   CONTINUE
3580=C**         TRACE SCAN
3590=      IF(ILOG.NE.0.AND.ILOG.LE.2)
3600=     1  WRITE(ITRACE,957) NW,NHX,JD,MAXV,J4,J5,FIRS,FLAS
3610= 957  FORMAT(8H  SCANX ,6I5,2E9.3)
3620=C**          LOG STATS FROM SCAN
3630=      IF(ILOG.NE.0)WRITE(ILOGF,901)FIRS,FLAS
3640=      DO 33 J=1,JD
3650= 33   IF(ILOG.NE.0)WRITE(ILOGF,902)J,(DIST(N,J),N=1,8)
3660= 901  FORMAT(34H FIRST PASS SCAN STATISTICS ,LIM= ,2(1XE9.3)/
3670=     1  10H COMPONENT,2X3HNUM,8X2HAV,6X4HCSDL,2X4HCSDR,7X3HMIN,
3680=     2  7X3HMAX,6X4HMODE,5X5HVMODE)
3690= 902  FORMAT(1X12,2H= ,8(1XE9.3) )
3700=      RETURN
3710=      END
3720=
```

```
1970=
1980=      SUBROUTINE COPY(BUF,FVEC,DIST,IEOJ,NW,JDX,NHX,VECS)
1990=      COMMON/TRACE/ILOG,ITRACE
2000=      COMMON/NAMES/INPF,IOUTF,IFEAT,IUSER,IDUM,ILOGF,IDIST
2010=      COMMON/CREA/IOPT,IHIS,ISCAN,NH1,NF1,NL1,IGC,
2020=     1    LABLF,JD,LB,IC,MAXV,NBUC,NTC,MEM,FIRS,FLAS,IU
2030=      DIMENSION BUF(NW),FVEC(JDX,1),DIST(NHX,1),VECS(1)
2040=      EQUIVALENCE(AVEC,NVEC),(AIC,KC)
2050=      DATA NAME/4HFEAT/
2060=      DATA N1,N2,I1,I2,F0/1,2.98,99,0./
2070=      DATA FMK/.999E99/
2080=C**           REINITIALIZE DIST FOR STATX
2090=      DO 1 J=1,JD
2100= 1    DIST(1,J)=0.
2110=C             INITIALIZE VARIABLES
2120=      J1=JD
2130=      J2=2*JD
2140=      LAS=0
2150=      LASV=0
2160=      L=1
2170=      NBLKS=0
2180=      REWIND IFEAT
2190=      IF(IOPT.EQ.2.OR.IOPT.EQ.3)FIRS=-1.0
2200=      IF(IOPT.EQ.1.OR.IOPT.EQ.4)FIRS=0.0
2210=      IF(IOPT.NE.5)FLAS=1.0
2220=C             WRITE HEADER RECORD
2230=      WRITE(IFEAT) NAME,LABLF,JDX,LB,IC,MAXV,IOPT,IHIS,FIRS,FLAS
2240=C
2250=C************  LOOP TO COPY FILE CLASS BY CLASS
2260=C
2270= 5    CONTINUE
2280=      JFIL=1
2290=      CALL GETFEA(IUSER,BUF,NW,FVEC(1,L),JD,KC,NVEC,IGC,IEOJ)
2300=      IF(IEOJ.EQ.1) LAS=1
2310=      LASV=L
2320=      FVEC(JD+1,L)=AVEC
2330=      FVEC(JD+2,L)=AIC
2340=      FVEC(JD+3,L)=ENER(FVEC(1,L),JD)
2350=      T=FVEC(JD+3,L)
2360=      IF(IOPT.EQ.3) T=SQRT(T)
2370=      DO 118 J=1,JD
2380=C**           GLOBAL HISTOGRAM
2390=C**           HISTOGRAM MAY PRECEDE OR FOLLOW VECTOR TRANSFORMS
2400=C**           SINCE EACH (T) AFFECTS EACH FVEC COMPONENT
2410=C**           INDIVIDUALLY AS A LINEAR SHIFT OF ITS HISTOGRAM
2420=C**           USE GLOBAL FIRS,FLAS FOR COMPARABLE DISTRIBUTIONS
2430=C             NORMALIZE AS OPTED AND SAVE VECTOR MAG
2440=      IF(IOPT.EQ.2.OR.IOPT.EQ.3)
2450=     1  FVEC(J,L)=FVEC(J,L)/T
2460=C             SINCE F1,FN ARE COMPONENT DEPENDENT THIS TRANSFORM
2470=C             SHIFTS RELATIVE ANGLES, KEEPS RELATIVE MAGS(OPT=0)
2480=C**           (FIRS,FLAS) WILL BECOME (0.,1.0) LATER
2490=      IF(IOPT.NE.4) GOTO 9
2500= 8    FVEC(J,L)=FVEC(J,L)-DIST(5,J)/(DIST(6,J)-DIST(5,J)+1E-7)
2510=      GOTO 15
2520= 9    CONTINUE
2530=C             SINCE FIRS,FLAS ARE GLOBAL, THIS SHIFTING
2540=C             MAINTAINS RELATIVE ANGLES AND MAGS(OPT=5)
2550=C**           (FIRS,FLAS) WILL BECOME (0.,1.0) LATER
2560=      IF(IOPT.NE.1) GOTO 15
2570= 13   FVEC(J,L)=FVEC(J,L)-FIRS/(FLAS-FIRS+1E-7)
2580= 15   CONTINUE
2590=C**           COMPUTE MIN,MAX STATS ON ALL OUTPUT VECTORS
2600=      J4=J1+J
2610=      J5=J2+J
2620=      CALL STATX(0,FVEC(J,L),DIST(1,J),VECS(J4),VECS(J5),0)
2630=      IF(IHIS.NE.1) GOTO 7
2640=      CALL STATH(0,FVEC(J,L),DIST(1,J),DIST(9,J),NBUC,FIRS,FLAS)
2650= 7    CONTINUE
2660=C**           LAST VECTOR OF RECORD
2670=      IF(LAS.NE.1) GOTO 118
2680=      CALL STATX(1,0.0,DIST(1,J),VECS(J4),VECS(J5),0)
2690=      CALL STATH(1,0.0,DIST(1,J),DIST(9,J),NBUC,FIRS,FLAS)
2700= 118  CONTINUE
2710=      IF(IEOJ.NE.0) GOTO 18
2720=      L=L+1
2730=      IF(L.LE.LB) GOTO 5
```

```
2740=C******        PROCESS FEATURE VECTOR BLOCK
2750=C******        FVEC BLOCK IS FULL OR CLASS IS DONE
2760=C******        FVEC(JD+1,L)-->NVEC(I),  FVEC(JD+3,L)-->ENER(F)
2770=C******        FVEC(JD+2,L) -->IC(I), FVEC(JD+2,LASV)(--NEGATIVE
2780=C******        FVEC(1-->JD,L) --> VECTOR COMPONENTS(F)
2790= 18   CONTINUE
2800=      LS=LASV-1
2810=      IF(LS.LE.0) LS=1
2820=      IF(ILOG.NE.0.AND.ILOG.EQ.1)
2830=     1  WRITE(ILOGF,999)NBLKS,LASV,((FVEC(J,L),J=1,JDX),FMK,
2840=     2  L=1,LASV,LS)
2850= 999  FORMAT(15H  BUFFER TRACE ,2I6/((((10(1XE9.3) ))))
2860=      L1=LASV+1
2870=      IF(IEOJ.NE.0)FVEC(JD+2,LASV)=-FVEC(JD+2,LASV)
2880=      IF(L1.GT.LB)GOTO 40
2890= 22   CONTINUE
2900=      DO 30 L=L1,LB
2910=      DO 25 J=1,JDX
2920= 25   FVEC(J,L)=0
2930= 30   CONTINUE
2940= 40   CONTINUE
2950=      NBLKS=NBLKS+1
2960=      IF(ILOG.NE.0.AND.ILOG.LE.2)
2970=     1    WRITE(ITRACE,957)L,L1,LB,LS,LASV,IEOJ,NVEC
2980= 957  FORMAT(8H  COPYX ,8I8)
2990=      WRITE(IFEAT) ((FVEC(K,M),K=1,JDX),M=1,LB)
3000=      L=1
3010=      IF(IEOJ.NE.1) GOTO 5
3020= 50   CONTINUE
3030=      IEOJ=0
3040=C**           WRITE TRAILER RECORD(MINS,MAXS)
3050= 56   CONTINUE
3060=      WRITE(IFEAT)
3070=     1  (DIST(5,J),J=1,JD),N1,I1,F0,
3080=     2  (DIST(6,J),J=1,JD),N2,I2,F0
3090=C**           PUTS AV AND SD INTO DIST
3100=      Do 59 J=1,JD
3110=      J4=J1+J
3120=      J5=J2+J
3130=      DIST(2,J)=VECS(J4)
3140=      DIST(3,J)=VECS(J5)
3150= 59   CONTINUE
3160=      RETURN
3170=      END
3180=
```

```
4010=
4020=      SUBROUTINE GETFEA(IUSER,BUF,NW,FVEC,JD,IC,NVEC,IGC,IEOJ)
4030=C
4040=C********    USER INPUT MODULE SPECIFICATIONS
4050=C*    IUSER-->NAME OF USER INPUT FILE
4060=C*    BUF  -->BUFFER REQUIRED FOR WORK SPACE BY MODULE
4070=C*    NW   -->NO WORDS IN BUFFER
4080=C*    FVEC -->FEATURE VECTOR PASSED TO SYSTEM FROM MODULE
4090=C*    JD   -->NO OF COMPONENTS OF FEATURE VECTOR
4100=C*    IC   -->CLASS TO WHICH FVEC A PRIORI BELONGS
4110=C*    NVEC -->NUMBER OF FVEC WITHIN CLASS
4120=C*    IGC  -->INPUT OPTION
4130=C*    IEOJ -->CONTROL FLAG  (SEE BELOW)
4140=C********
4150=C
4160=C***  REWINDS IUSER ON FIRS ENTRY
4170=C***  RETURNS FVEC,IC,NVEC,IEOJ  ON EACH ENTRY
4180=C***  SETS IC AND NVEC SO AS TO ALWAYS REFLECT
4190=C**     CLASS,AND VEC-NO W/IN CLASS OF RETURNED FVEC
4200=C***  SETS IEOJ--> 0 IF NVEC IS FIRST TO (LAST-1)TH OF CLASS
4210=C***  SETS IEOJ--> -1 IF NVEC IS LAST OF CLASS
4220=C***  SETS IEOJ--> +1 IF NVEC IS LAST OF FILE
4230=C***  RESETS INTERNAL FLAGS AT LAST OF FILE
4240=C******
4250=C       THIS SUBMODULE READS FEATURE VECTORS FROM A DATA BASE OF
4260=C       RADAR SIGNATURE PATTERNS. THE FEATURES WERE COMPUTED BY
4270=C       MAJOR DEL KULCHAK IN THE COURSE OF HIS THESIS WORK ON
4280=C       A PATTERN RECOGNITION PROBLEM. THIS SUBMODULE MEETS
4290=C       SPECIFICATIONS FOR USER INPUT TO THE 'CREATE' MODULE OF
4300=C       THE 'BOX80' SYSTEM.
4310=C*******
4320=      DIMENSION BUF(1),FVEC(1),KC(2)
4330=C**       INITIALIZE ON FIRST ENTRY
4340=      IF(IFIRS.EQ.1) GOTO 10
4350=      IFIRS=1
4360=      REWIND IUSER
4370=      IA=1
4380=      IB=2
4390=      J1=(IA-1)+JD+1
4400=      JN=J1+JD-1
4410=      READ(IUSER,900)KC(IA),NS,NUM,(BUF(J),J=J1,JN)
4420= 900  FORMAT(1X,I1,I2,2X,I3,14F5.3)
4430=      NVEC=0
4440=C**       PROCESS ON FOLLOW ON ENTRIES
4450= 10   CONTINUE
4460=      IF(IEOJ.NE.0) NVEC=0
4470=      IEOJ=0
4480=      IK=IA
4490=      IA=IB
4500=      IB=IK
4510=      NVEC=NVEC+1
4520=      J1=(IB-1)*JD+1
4530=      DO 15 J=1,JD
4540=      FVEC(J)=BUF(J1)
4550=      J1=J1+1
4560= 15   CONTINUE
4570=      IC=KC(IB)
4580=      IF(EOF(IUSER))20,16
4590= 16   J1=(IA-1)+JD+1
4600=      JN=J1+JD-1
4610=      READ(IUSER,900)KC(IA),NS,NUM,(BUF(J),J=J1,JN)
4620=      IF(KC(IA).EQ.KC(IB)) GOTO  30
4630=      IEOJ=-1
4640=      GOTO 30
4650=C**       END OF FILE
4660= 20   CONTINUE
4670=      IFIRS=0
4680=      IEOJ=1
4690=C**       EXIT
4700= 30   CONTINUE
4710=      RETURN
4720=      END
4730=
```

```
4740=
4750=      SUBROUTINE CLAFEA(IUSER,BUF,NW,FVEC,JD,IC,NVEC,IGC,IEOJ)
4760=C******
4770=C       THIS SUBMODULE READS FEATURE VECTORS FROM A DATA BASE OF
4780=C       RADAR SIGNATURE PATTERNS. THE FEATURES WERE COMPUTED BY
4790=C       MAJOR CLAY STEWART IN THE COURSE OF HIS DISSERTATION ON
4800=C       A  PATTERN RECOGNITION ALGORITHM. THIS SUBMODULE MEETS
4810=C       SPECIFICATIONS FOR USER INPUT TO THE 'CREATE' MODULE OF
4820=C       THE 'BOI80' SYSTEM.
4830=C*******
4840=      DIMENSION NCLAS(7),BUF(NW),FVEC(1)
4850=C#    DATA NCLAS(1),NCLAS(2),NCLAS(3),NCLAS(4),
4860=C#   1NCLAS(5),NCLAS(6),NCLAS(7)
4870=C#   2   /751,349,471,83,18,65,0/
4880=      DATA NCLAS(1),NCLAS(2),NCLAS(3),NCLAS(4)
4890=     1 /620,296,471,0/
4900=C**           INITIALIZE ON FIRST ENTRY
4910=      IF(IFIRS.EQ.1) GOTO 5
4920=      REWIND IUSER
4930=      IFIRS=1
4940=      NVEC=0
4950=      KC=1
4960=C**           PROCESS FOLLOW ON ENTRIES
4970= 5    CONTINUE
4980=      IF(IEOJ.NE.0) NVEC=0
4990=      IEOJ=0
5000=      READ(IUSER,900)(FVEC(J),J=1,JD)
5010= 900  FORMAT(F10.5)
5020=      NVEC=NVEC+1
5030=      IF(EOF(IUSER))20,6
5040=  6   IF(NVEC.LT.NCLAS(KC)) GOTO 30
5050=      KC=KC+1
5060=      IEOJ=-1
5070=      IF(NCLAS(KC).GT.0) GOTO 30
5080=C**           END OF FILE
5090= 20   CONTINUE
5100=      IEOJ=1
5110=      IFIRS=0
5120=C**           EXIT
5130= 30   CONTINUE
5140=      IC=KC
5150=      IF(IEOJ.NE.0) IC=KC-1
5160=      RETURN
5170=      END
5180=
```

```
5190=
5200=      SUBROUTINE SPOFEA(IUSER,BUF,NW,FVEC,JD,IC,NVEC,IGC,IEOJ)
5210=C*******
5220=C       THIS SUBMODULE READS FEATURE VECTORS FROM A DATA BASE OF
5230=C       ALPHABETIC CHARACTER PATTERNS. THESE FEATURES WERE
5240=C       PRODUCED BY A FOURIER TRANSFORM OF CENTERED CHARACTER
5250=C       PICTURES (32X32) BY CAPT TROY SPONNAUGLE IN THE COURSE
5260=C       OF HIS THESIS WORK ON LETTER RECOGNITION ALGORITHMS.
5270=C       THIS SUBMODULE MEETS SPECIFICATIONS FOR USER INPUT TO THE
5280=C        THE 'BOX80' SYSTEM.
5290=C*******
5300=      DIMENSION BUF(NW),FVEC(JD)
5310=C**            INITIALIZE ON FIRST ENTRY
5320=C**              OPEN FILE ONCE ONLY PER JOB
5330=C        IGC CONTROLS LIM(VECTORS READ), AND NDIM(INPUT DIM)
5340=C          1,   2,   3,   4,   5,   ...
5350=C         780,1850,3900, 780,1850,   ...(LIM)
5360=C          9,  25,  49,   9,  25,   ...(NDIM)
5370=      ISW=(IGC-1)/3 + 1
5380=      IF(ISW.EQ.1)LIM=780
5390=      IF(ISW.EQ.2)LIM=1850
5400=      IF(ISW.EQ.3)LIM=3900
5410=      NDIM=( (ISW-2)*2)**2
5420=      IF(IFIRS.EQ.2) GOTO 5
5430=      IF(IFIRS.EQ.1) GOTO 3
5440=      CALL OPENMS(IUSER,BUF(90),3901,0)
5450=      IFIRS=1
5460= 3    CONTINUE
5470=      IFIRS=2
5480=      KC=1
5490=      NVEC=0
5500=      NIN=1
5510=C**              PROCESS FOLLOW ON ENTRIES
5520= 5    CONTINUE
5530=      IF(IEOJ.NE.0) NVEC=0
5540=      IEOJ=0
5550=      CALL READMS(IUSER,BUF,85,NIN)
5560=      NIN=NIN+26
5570=      CALL REDUCE(BUF,81,BUF,NDIM)
5580=      DO 6 J=1,JD
5590= 6    FVEC(J)=BUF(J)
5600=      NVEC=NVEC+1
5610=      IF(NIN.LE.LIM) GOTO 7
5620=C         LIM=2600-->100 ALPHABETS, ETC
5630=      KC=KC+1
5640=      NIN=KC
5650=      IEOJ=-1
5660= 7    CONTINUE
5670=      IF(KC.LT.27) GOTO 30
5680=C**            END OF FILE
5690= 20   CONTINUE
5700=      IEOJ=1
5710=      IFIRS=1
5720=C**            EXIT
5730= 30   CONTINUE
5740=      IC=KC
5750=      IF(IEOJ.NE.0) IC=KC-1
5760=      RETURN
5770=      END


5780=
5790=      SUBROUTINE REDUCE(INP,NI,IOT,NO)
5800=C             **PER CAPT T SPONAUGLE 7/77**
5810=      DIMENSION INP(NI),IOT(NO)
5820=      IF(NI.NE.NO) GOTO 90
5830=C             MOVE INP TO IOT IF NO SHUFFLE NEEDED
5840=      DO 85 N=1,NO
5850= 85   IOT(N)=INP(N)
5860=      GOTO 125
5870=C             BEGIN REDUCE LOOP
5880= 90   CONTINUE
5890=      KI=NI**.5
5900=      KO=NO**.5
5910=      KSKIP=(KI-KO)*2
5920=      KDOWN = KO/2 + 1
5930=      IOT(1)=INP(1)
5940=      IF(NO.EQ.1) RETURN
5950=      DO 100 N=2,KO
5960= 100  IOT(N)=INP(N)
5970=      KSTART=M=KO+1
5980=C             INNER LOOP
5990=      DO 120 NROW=2,KDOWN
6000=      M=M+KSKIP
6010=      KSTOP=KSTART+KO*2-1
6020=      DO 110 N=KSTART,KSTOP
6030=      IOT(N)=INP(M)
6040=      M=M+1
6050= 110  CONTINUE
6060=      KSTART=KSTOP+1
6070= 120  CONTINUE
6080=C             END INNER LOOP
6090= 125  DO 130 I=1,4
6100= 130  IOT(NO+I)=INP(NI+I)
6110=      RETURN
6120=      END
```

APPENDIX F

DEFINE Module

```
  100=      PROGRAM DEFINE(FEAT,OLDC,NEWC,HISC,LOCF=64,
  200=     1     INPUT=64,OUTPUT=64)
  300=C**           JULY 26 1978   JRL
  600=C
  700=      COMMON WO(7500)
  800=      COMMON/CNUM/NEXC,NTC
  900=      COMMON/CONTRL/ IMOD,IALL,ISYM,ICAL,IPIC,IH,IS,
 1000=     1  IUKER,LABLC,LABLF,SIZE,IPK
 1100=      COMMON/TRACE/ILOG,ITRACE
 1200=      COMMON/PARAM/MEMC,MEM,NHX,JDX,ICX,NC1,NF1,NH1,IC
 1300=      COMMON/NAMES/INPF,IOUTF,IFEAT,IOLDC,INEWC,ILOGF,IHISC
 1400=      DATA IHF,IHI/1HF,1HI/
 1500=C**********
 1600=      MEM=7500
 1700=      DO 5 M=1,MEM
 1800= 5    WO(M)=0.
 1900=C**********
 2000=      CALL DEFD(IEOJ)
 2300= 957  FORMAT(9H  DEFD-E ,10I5)
 2400=      IF(IEOJ.GT.0) GOTO 100
 2500=      IF(IMOD.EQ.IHF) CALL LOADC(WO(NC1),JDX,ICX)
 2600=C**          LOOP PICKS CLASSES FOR DEFINITION
 2700= 10   IF(IALL.NE.1) CALL NEXCLA(WO(NF1),WO(NC1),JDX,IEOJ)
 3200=      IF(IEOJ.GT.0) GOTO 100
 3300=      IF(IALL.EQ.1) NEXC=NEXC+1
 3400=      IF(NEXC.GT.IC) GOTO 800
 3500=      ICON=ISYM+ICAL+IPIC
 3600=      IF(ICON.EQ.0) GOTO 10
 3700=      CALL CLASSX(WO(NF1),WO(NC1),WO(NH1),
 3800=     1   JDX,NHX,IEOJ)
 3900=      IF(IEOJ.GT.0) GOTO 100
 4000=      GOTO 10
 4100=C**          END OF JOB OPTIONS
 4200=  800 CONTINUE
 4250=      IF(IMOD.EQ.IHI)  CALL SETLIM(WO(NC1),WO(NF1),JDX)
 4260=      IF(IPK.EQ.1.OR.ILOG.GT.0)
 4300=     1  CALL PRCLAS(WO(NC1),WO(NC1),WO(NC1),JDX)
 4500=      CALL WRCLAS(WO(NC1),JDX,ICX,IS,LABLC)
 4600=      IF(IEOJ.GT.0) GOTO 100
 4700=      STOP
 4800=C**          ERROR EXIT PRINTS ERROR KEY
 4900= 100  CONTINUE
 5000=      WRITE(IOUTF,999)
 5100=      IF(IEOJ.EQ.1) WRITE(IOUTF,900)
 5200=      IF(IEOJ.EQ.2) WRITE(IOUTF,901)
 5300=      IF(IEOJ.EQ.3) WRITE(IOUTF,902)
 5400=      IF(IEOJ.EQ.4) WRITE(IOUTF,903)
 5500=      IF(IEOJ.EQ.5) WRITE(IOUTF,904)
 5600=      IF(IEOJ.EQ.6) WRITE(IOUTF,905)
 5700= 900  FORMAT(1X,18H TOO MANY RETRIES )
 5800= 901  FORMAT(2X,27H  NO SUCH CLASS OF FEATURES)
 5900= 902  FORMAT(2X,19H  NOT ENOUGH MEMORY)
 6000= 903  FORMAT(22H  CLAS AND FEAT FILE DISAGREE)
 6100= 904  FORMAT(2X,30H  NO MORE ROOM FOR NEW CLASSES)
 6200= 905  FORMAT(2X,20H  NO MAX,MIN VECTORS)
 6300= 999  FORMAT(2X,12H  ERROR EXIT)
 6400=      STOP
 6500=      END
```

```
133500=
133600=      SUBROUTINE DEFD(IEOJ)
133700=      COMMON/CNUM/NEXC,NTC
133800=      COMMON/CONTRL/IMOD,IALL,ISYM,ICAL,IPIC,IH,IS,
133900=     1  IUKER,LABLC,LABLF,SIZE,IPK
133950=      COMMON/PCLUS/ICPS,IQUIK,ISCF
134000=      COMMON/TRACE/ILOG,ITRACE
134100=      COMMON/DIST/FIRS,FLAS,FIRC,FLAC,NBUC,KTR,NI
134200=      COMMON/CREC/LASV,IEOFR,NBFR,LB,MAXV
134300=      COMMON/PARAM/MEMC,MEM,NHX,JDX,ICX,NC1,NF1,NHI,IC
134400=      COMMON/NAMES/INPF,IOUTF,IFEAT,IOLDC,INEWC,ILOGF,IHISC
134500=      COMMON/KFILE/MAXKV,NCIX,NENT,MSCAL,MZAPD
134600=      DIMENSION IOPNS(4)
134700=      DATA IHE,IHF,IHI,IHP,IHS,IHY,IHL,IHT/
134800=     1 1HE,1HF,1HI,1HP,1HS,1HY,1HL,1HT/
134900=C**
135000=C**      INITIALIZE VARIABLES AND DEFAULT PARAMETERS
135010=C**        NON-ANSI CODE DEFINES FILE NAMES PER CDC FTN
135100=      INPF=5LINPUT
135200=      IOUTF=6LOUTPUT
135300=      IFEAT=4LFEAT
135400=      IOLDC=4LOLDC
135500=      INEWC=4LNEWC
135600=      IHISC=4LHISC
135700=      ILOGF=4LLOGF
135800=      ITRACE=6LOUTPUT
135810=C**
135900=      ILOG=0
136000=      IE=0
136100=      IPK=0
136200=      IS=0
136300=      NBUC=0
136400=      IEOJ=-1
136500=      NN=6
136600=C**      SET OPERATING MODE AND VARIABLES
136700=C**         MODE-I -->  INITIALIZE CLAS FILE PROTOTYPES
136800=C**         MODE-F -->  FOLLOW ON PROCESS OF PROTOTYPES
136900=C**         OPN-L  -->  PRINT SELECTED DATA ON LOGF
137000=C**                        T-->TRACE,L-->LOG,Y-->BOTH
137100=C**         OPN-P  -->  PRINT HUSKS ON LOGF AT END
137200=C**         OPN-S  -->  SAVE CLAS OVER OLDC
137300=C**         OPN-E  -->  EXTEND CLAS FILE BEYOND ICX
137400= 5    CONTINUE
137500=      WRITE(IOUTF,901)
137700= 901  FORMAT(8H  DEFINE/25H  ENTER MODE AND OPTIONS /2H *)
137800= 7    READ(INPF,902)IMOD,IOPNS
137900=      WRITE(ILOGF,903) IMOD,IOPNS
138000= 903  FORMAT(1XA1,1X,4A1)
138100= 902  FORMAT(A1,1X,4A1)
138200=      IF(IMOD.EQ.IHI)IALL=1
138300=      IF(IMOD.EQ.IHF)IALL=0
138400=      DO 10 J=1,4
138500=      IF(IOPNS(J).EQ.IHP) IPK=1
138600=      IF(IOPNS(J).EQ.IHS)IS=1
138700=      IF(IOPNS(J).EQ.IHE)IE=1
138800=      IF(IOPNS(J).EQ.IHL)ILOG=3
138900=      IF(IOPNS(J).EQ.IHT)ILOG=1
139000=      IF(IOPNS(J).EQ.IHY) ILOG=2
139100= 10   CONTINUE
139200=      IF(IALL.NE.9) GOTO 15
139300=      CALL ERR(IOUTF,NN,IEOJ)
139400=      IF(IEOJ.GT.0) GOTO 160
139500=      GOTO 7
139600=C**      OPEN FILES
139700= 15   CONTINUE
139750=      IF(ILOG.NE.0)WRITE(ILOGF,901)
139800=      CALL OPENX(IMOD,IE,LABLC,LABLF,IEOJ)
139900=      IF(IEOJ.GT.0) GOTO 160
140000=C**      SET MEMORY PARAMETERS
140100=C**         JDX   VECTOR DIM +3,  FEAT FILE
140200=C**         ICX   TOTAL ALLOTED COLS, CLAS FILE
140300=C**         IC    NO DATA CLASSES, FEAT FILE
140400=C**         NTC   TOTAL DEFINED CLASSES
140500=C**         LB    NO VECTORS PER BLOCK, FEAT FILE
140600=C**         NHX   NO HISTO BUCKETS + STATS TEMPORARIES
140700=C**         MAXKV MAX HUSK VECTORS PER CLASS
140800=C**         NCIX  FIRST COL OF CLAS INDEX AREA(IA)
140900=C**         NENT  NO. INDEX PAIRS IN(IA)
141000=C**         NTCOL TOTAL ALLOCATED COLUMNS
141100=      IF(IMOD.NE.IHI.AND.IE.NE.1)GOTO 150
141200=C****************
141300=C**      OBTAIN INITIAL PARAMS FOR NEW OR REVISED CLAS FILE
141400=C****************
141500=      LABLC=LABLF*100+1
141600= 98   WRITE(IOUTF,905) LABLC
141700= 905  FORMAT(19H  INITIAL CLAS FILE,2X,I6/
141800=     1      18H  ENTER PARAMETERS/
141900=     2      13H  ND,NE,HUSKR/2H *)
142000= 99   READ(INPF,906)NBUC,NE,HUSKR
142100= 906  FORMAT(I2,1X,I2,1X,F5.4)
142200= 907  FORMAT(2X,I2,1X,I2,1X,F5.4)
```

```
142300=      IF(ILOG.NE.0)WRITE(ILOGF,905) LABLC
142400=      IF(ILOG.NE.0)WRITE(ILOGF,907)NBUC,NE,HUSKR
142500=C**      CHECK PARAMETERS
142600=      IF(NBUC.LT.0.OR.NBUC.GT.50)GOTO 40
142700=      IF(NE.LT.0.OR.HUSKR.LT.0.) GOTO 40
142800=      GOTO 50
142900= 40   CALL ERR(IOUTF,NN,IEOJ)
143000=      IF(IEOJ.GT.0) GOTO 160
143100=      GOTO 99
143200= 50   CONTINUE
143300=      IF(IMOD.NE.IHI) GOTO 110
143400=C**      ALLOCATE MEM TO INITIAL FILE
143500=C**      ADJUST NE,NBUC,MAXV TO SET MEMC
143600= 100  CONTINUE
143700=      ISYM=1
143800=      CALL ALLOC(IC,NE,HUSKR,IEOJ)
143900=      IF(IEOJ.GT.0)  GOTO 160
144000=      IF(NBUC.GT.0) IH=1
144200=      NTC=IC
144300=      GOTO 140
144400=C**      REALLOCATE MEMORY TO GLDC
144500= 110  CONTINUE
144600=      CALL ALLOC(NTC,NE,HUSKR,IEOJ)
144800=      IF(IEOJ.GT.0) GOTO 160
144900=C**      REVISE CLAS FILE POINTERS
145000= 140  CONTINUE
145100=      ICX=MEMC/JDX
145400=C***************
145500=C**      SETUP BUFFER POINTERS
145600=C***************
145700= 150  CONTINUE
145800=      IEOJ=0
145900=      NHX=NBUC+8
146000=      NF1=1
146100=      NC1=NF1+JDX*LB
146200=      NH1=NC1+MEMC
146300=C**      EXIT
146400= 160  CONTINUE
146500=      IF(IEOJ.EQ.5) GOTO 98
146600=      IF(ILOG.NE.0.AND.ILOG.LE.2)
146700=     1   WRITE(ITRACE,957)HUSKR,NF1,NC1,NH1,NCIX,NENT,ICX,MEMC,
146800=     1 IE,NBUC,MAXV,NTC,IC,JDX,NE
146900= 957  FORMAT(8H  DEFD  ,F6.4,20I5)
147000=      RETURN
147100=      END
```

```
125500=
125600=
125700=      SUBROUTINE ALLOC(KC,NE,HUSKR,IEOJ)
125800=      COMMON/NAMES/INPF,IOUTF,IFEAT,IOLDC,INEWC,ILOGF,IHISC
125900=      COMMON/TRACE/ILOG,ITRACE
126000=      COMMON/PARAM/MEMC,MEM,KHX,JDX,ICX,NC1,NF1,KH1,IC
126100=      COMMON/KFILE/MAXKV,NCIX,NENT,MSCAL,MZAPD
126200=      COMMON/DIST/FIRS,FLAS,FIRC,FLAC,NBUC,KTR,NI
126300=      COMMON/CREC/LASV,IEOFR,NEPR,LB,MAXV
126400=      DATA IHY/1HY/
126500=C**     USES MEM,JD,KC,MAXV,HUSKR,,TO SET MAXKV,MEMC
126600=C**     AFTER ADJUSTING NE,NBUC,MAXKV USING STMT FNS
127000=C**     KMC-->NO VECTORS FOR CAV,CSDL,CSDR,HUSKV
127100=      KMC(KC,NE,MAXKV)=
127200=     1  (MAXKV+3)*(KC+NE)
127290=C**     KIX-->NO ENTRIES IN LIST.IX FOR CLAS FILE
127300=      KIX(MEMC,MEMX,JDX) =
127400=     1  ((2*(MEMC/JDX)+2*(MEMX/JDX)+4)/JDX+1)*JDX
127490=C**     KME-->AMOUNT OF UNUSED MEMORY,GIVEN EXISTING REQUESTS
127500=      KME(MEMT,MEMX,MEMH,MEMC,JDX)=
127600=     1  MEMT-MEMC-MEMH-MEMX-KIX(MEMX,MEMC,JDX)
127700=C**     END OF STATEMENT FUNCTIONS
127800=      JD=JDX-3
127900=      NBUC=(((NBUC-1)/5)+1)*5
128000=      KNB=NBUC
128010=C**        ADD 1 TO NE FOR MIN/MAX VECTORS
128020=      NE=NE+1
128100=      KNE=NE
128200=      MAXKV=FLOAT(MAXV)*HUSKR
128300=      IF(MAXKV.GT.0)MAXKV=MAXKV/JD+1
128400=      KKV=MAXKV
128500=      MEMT=MEM-JDX*LB-8*JD
128600=      MEMX=JDX*KMC(0,NE,MAXKV)
128700=      MEMH=JD*NDUC
128800=      MEMC=JDX*KMC(KC,0,MAXKV)
128900= 105  CONTINUE
129000=      IF(KME(MEMT,MEMX,MEMH,MEMC,JDX).GT.0) GOTO 140
129100=      IF(NE.LE.1) GOTO 110
129200=      NE=NE-1
129300=      MEMX=JDX*KMC(0,NE,MAXKV)
129400=      GOTO 105
129500= 110  CONTINUE
129600=      IF(NDUC.LE.0) GOTO 120
129700=      NBUC=NBUC-5
129800=      MEMH=JD*NBUC
129900=      GOTO 105
130000= 120  CONTINUE
130100=      IF((JD*MAXKV).LE.0) GOTO 130
130200=      MAXKV=MAXKV-1
130300=      MEMC=JDX*KMC(KC,0,MAXKV)
130400=      GOTO 105
130500= 130  CONTINUE
130600=      IEOJ=3
130700=      GOTO 145
130800= 140  CONTINUE
130900=      IF(KNB.NE.NBUC)GOTO 150
131000=      IF(KNE.NE.NE) GOTO 150
131100=      IF(KKV.NE.MAXKV) GOTO 150
131200= 145  CONTINUE
131300=      MEMC=MEMC+MEMX+KIX(MEMC,MEMX,JDX)
131400=      MAXKE=MAXKV*JD
131410=C**        SET NENT AND NCIX
131420=      NTCOL=KMC(KC,NE,MAXKV)
131430=      NENT=((NTCOL*2+4)/JDX+1)*JDX/2
131440=      NCIX=NTCOL+1
131450=      IF(ILOG.NE.0.AND.ILOG.LE.3)
131500=     1WRITE(ILOGF,900)NBUC,NE,MAXKE,MEMC
131600=      IF(ILOG.NE.0.AND.ILOG.LE.2)
131700=     1  WRITE(ITRACE,957)HUSKR,KNB,KNE,KKV,MEMT,MEMX,MEMH,MEMC,
131800=     2  MAXV,MAXKV,KC,NE,NBUC,LB,JD,IEOJ
131900= 957  FORMAT(8H  ALLOC ,E9.3,1715)
132000=      RETURN
132100=C**        APPROVE ADJUSTED PARAMETERS
132200= 150  CONTINUE
132300=      WRITE(IOUTF,908)NBUC,NE,MAXKE,MEMC
132400= 908  FORMAT(28H APPROVE ADJUSTED PARAMETERS/
132500=     1 28H  NB,NE,MAXRV,CLAS FILE SIZE/2X,I2,1X,I2,1X,I5,9X,I6/)
132600=      WRITE(IOUTF,910)
132700=  910 FORMAT (14H  ENTER Y OR N/2H *)
132800=      READ(INPF,909)IOK
132900= 909  FORMAT(A1)
133000=      IF(IOK.EQ.IHY) GOTO 145
133100=      IEOJ=5
133200=      GOTO 145
133300=      END
133400=
```

```
7000=      SUBROUTINE NEXCLA(FEAT,CLAS,JDX,IEOJ)
7100=      COMMON/CNUM/NEXC,NTC
7200=      COMMON/TRACE/ILOG,ITRACE
7300=      COMMON/CONTRL/ IMOD,IALL,ISYM,ICAL,IPIC,IH,IS,
7400=     1   IUKER,LABLC,LABLF,SIZE,IPK
7450=      COMMON/FCLUS/ICPS,IQUIK,ISCF
7500=      COMMON/KVEC/ KK,ILIS(15),ISA
7600=      COMMON/KANES/INPF,IOUTF,IFEAT,IOLDC,INEWC,ILOGF,IHISC
7700=C**
7800=C**         SUBMODULE SETS NEXC AND CONTROLS
7900=C**
8000=C**      CONTROL OPTIONS FOR NEXCLAS
8100=C**      KCPS H -NN
8200=C**      ...KERNEL OPTIONS CALC,PICK,SET
8300=C**      DEFN H -NN
8400=C**         DEFINE PROCESS CAN GENERATE ASSYMETRIC CLASSES
8500=C**      REFN H +NN
8600=C**         REFINE PROCESS ONLY REGENERATES SYMMETRIC CLASSES
8700=C**      QUIT
8800=C**         ENDS JOB
8900=C**         H--> HISTOGRAM THIS CLAS AND SAVE IT
9000=C**         +NN-->  USE HUSK VECTOR TO REJECT FVECS
9100=C**         -NN-->  IGNORE HUSK VECTOR
9200=C**          00-->  DO FULL FILE INITIALIZE PROCESS
9300=C**
9400=      DIMENSION FEAT(JDX,1),CLAS(JDX,1),ITYP(5)
9500=      DATA LASTN/0/,LASTI/0/,KINIT/0/
9600=      DATA IHD,IHN,IHQ,IHR,IHA,IHK,IHC,IHB,IHS,IHP,IHH,IHU,IHM,IHY/
9700=     1  1HD,1HN,1HQ,1HR,1HA,1HK,1HC,1HB,1HS,1HP,1HH,1HU,1HM,1HY/
9800=C               SET CONTROLS AND PRINT HEADER
9900=      ISYM=0
10000=     IEOJ=-1
10100=     NN=6
10200=     ISA=IHD
10300=     IQUIK=0
10400=     DO 2 I=1,15
10500= 2   ILIS(I)=0
10600=C*******        OBTAIN CONTROLS FOR NEXT CLASS
10700= 111 CONTINUE
10800=     WRITE(IOUTF,900)
10900= 900 FORMAT(12H  NEXT-CLASS/29H  ENTER FUNCTION AND CLASS ID/
11000=    1 12H  AAAA,H,+NN/2H *)
11100=C**            GET FUNCTIONS,VALIDATE AND RETRY
11200= 5   READ(INPF,901) ITYP,ICLX
11300= 901 FORMAT(4A1,1X,A1,1X,I3)
11400=     IF(ITYP(1).EQ.IHQ) GOTO 220
11500=     IF(ITYP(1).EQ.IHR) ISYM=1
11600=     IF(ITYP(1).EQ.IHD) ISYM=2
11700=     IF(ICLX.LT.0) IUKER=1
11800=     ICLX=IABS(ICLX)
11900=     IH=0
12000=     IF(ITYP(5).EQ.IHH) IH=1
12100=     IF(IH.EQ.1.AND.ISYM.EQ.0) GOTO 6
12200=     IF(ITYP(1).EQ.IHR.OR.ITYP(1).EQ.IHD.OR.
12300=    1  ITYP(1).EQ.IHK) GOTO 10
12400= 6   CONTINUE
12500=     CALL ERR(IOUTF,NN,IEOJ)
12600=     IF(IEOJ.GT.0) GOTO 222
12690=C**    KNEW-->NO ENTRIES FOR CAV,CSDL,CSDR,HUSKV
12700=     GOTO 5
12800=C**            SETS FILE BEFORE FIRST DATA RECORD
12900=C**      THIS PERMITS HUSK SET AND THEN FULL DEFINE
13000= 10  CONTINUE
13100=     IF(ICLX.NE.0) GOTO 15
13200=        IALL =1
13300=        INIT=1
13400=        CALL RFEAT(FEAT,JDX,INIT,IEOJ)
13500=        NEXC=0
13600=        GOTO 222
13700= 15  CONTINUE
13800=     IF(ICLX.LT.LASTN) GOTO 6
13900=     LASTN=ICLX
14000=C**            SET NEXC
14100=     NEXC=ICLX
14200=     IF(ISYM.NE.0) GOTO 222
14300=C**            GET  HUSK  OPTIONS
14400=     DO 18 J=2,4
14500=     IF(ITYP(J).EQ.IHC) ICAL=1
14600=     IF(ITYP(J).EQ.IHP) IPIC=1
15100= 18  IF(ITYP(J).EQ.IHS) ISET=1
15200=C**             SIZE AND IQUIK  (KERSL AND FANDER)
15300=     IF(ICAL.NE.1) GOTO 25
15400=     WRITE(IOUTF,902)
15500= 902 FORMAT(  26H  ENTER MAX SIZE  KERNEL ,/7H  XX.XX/2H *)
15600= 20  CONTINUE
15700=     READ(INPF,903) SIZE
15800= 903 FORMAT(F5.2)
15900=     IF(SIZE.GT.(2.).OR.SIZE.LT.(8.)) GOTO 25
16000=     CALL ERR(IOUTF,NN,IEOJ)
16100=     IF(IEOJ.GT.0) GOTO 222
16200=     GOTO 20
```

```
16300= 25   CONTINUE
16400=      IF(IPIC.NE.1) GOTO 35
16500=      WRITE(IOUTF,904)
16600= 904  FORMAT(33H  ENTER BAUD, AND QUICK PLOT(Y,N)/7H  CPS,Q/2H *)
16700=      READ(INPF,905)ICPS,IQ
16800= 905  FORMAT(A1)
16900=      IF(IQ.EQ.IHY) IQUIK=1
17000=C**                SET SCALING OPTIONS FOR FANDER
17100=      WRITE(IOUTF,906)
17200= 906  FORMAT(30H  ENTER SCALING OPTION (U,C,M)/2H *)
17300=  34  READ(INPF,907)ITYP(1)
17400= 907  FORMAT(A1)
17500=      ISCF=9
17600=C**     U-->GLOBAL, C-->CLASS,  M-->MEANS
17700=C**     U-->FIRS.., C-->FIRC.., M-->MAHDIS
17800=      IF(ITYP(1).EQ.IHU) ISCF=0
17900=      IF(ITYP(1).EQ.IHC) ISCF=1
18000=      IF(ITYP(1).EQ.IHM) ISCF=2
18100=      IF(ISCF.NE.9) GOTO 35
18200=      CALL ERR(IOUTF,NN,IEOJ)
18300=      IF(IEOJ.GT.0) GOTO 222
18400=      GOTO 34
18500= 35   CONTINUE
18600=C**                SET  HUSK  VECTOR
18700= 45   CONTINUE
18750=      IF(ISET.NE.1) GOTO 222
18800=      WRITE(IOUTF,912)
18900= 912  FORMAT(19H  SET  HUSK  VECTOR/19H  A, NV, NV,...(15)/2H *)
19000= 55   READ (INPF,913) ISA,ILIS
19100= 913  FORMAT(A1,15(1X,I3))
19200=      IF(ISA.NE.IHP) GOTO 46
19300=C**                PRINT HUSK
19400=      CALL PHUSK(CLAS,JDX,IEOJ)
19500=      IF(IEOJ.GT.0) GOTO 222
19600=      GOTO 45
19700=C**                SET HUSK
19800= 46   CONTINUE
19900=      KK=0
20000=      IF(ISA.NE.IHB.AND.ISA.NE.IHD.
20100=     1  AND.ISA.NE.IHA) GOTO 65
20200=      IF(ILIS(1).EQ.0) GOTO 80
20300=C                ENSURE SUCCESSIVE NEXC'S ASCEND
20400=      DO 60 I=1,14
20500=      KK=I
20600=      IF(ILIS(I+1).EQ.0) GOTO 80
20700=      IF(ILIS(I).GE.ILIS(I+1)) GOTO  65
20800= 60   CONTINUE
20900=      GOTO 80
21000= 65   CALL ERR(IOUTF,NN,IEOJ)
21100=      IF(IEOJ.GT.0) GOTO 222
21200=      GOTO 55
21300= 80   CONTINUE
21400=C**              INSERT ILIS INTO  HUSK  PER ISA
21500=      IF(ILOG.NE.0.AND.ILOG.LE.2)
21600=     1  WRITE(ITRACE,958)ISA,KK,ILIS
21700= 958  FORMAT(10H  KERP-IN ,A1,1X,20I4)
21800=      CALL KERPUT(CLAS,JDX,NEXC,IEOJ)
21900=      DO 85 I=1,15
22000= 85   ILIS(I)=0
22100=      IF(IEOJ.LE.0) GOTO 222
22200=      IF(IEOJ.EQ.2)WRITE(IOUTF,914)
22300=      IF(IEOJ.EQ.4)WRITE(IOUTF,915)
22400=      IF(IEOJ.EQ.6) GOTO 90
22500= 914  FORMAT(16H  CLAS FILE FULL)
22600= 915  FORMAT(14H  HUSK IS FULL)
22700=      IEOJ=0
22800=      GOTO 222
22900=C**              KERPUT OUT OF SEQUENCE
23000= 90   CONTINUE
23100=      IEOJ=-1
23200=      GOTO 65
23300=C**              QUIT EXIT FINISHES JOB
23400= 220  CONTINUE
23500=      NEXC=999
23600=C**               NORMAL EXIT
23700= 222  CONTINUE
23705=      IF(ILOG.NE.0.AND.ILOG.LE.3)
23710=     1  WRITE(ILOGF,916)ICLX,IH,IUKER,ICAL,IPIC,ISET,ICPS,IQUIK,
23720=     2  ISCF,SIZE
23730= 916  FORMAT(20H  NEXT-CLASS OPTIONS/ 7H  CLASS,
23740=     1 49H,HIST,SHUCK,CLUSTER,S-HUSK,BAUD,Q-PLOT,SCALE,SIZE/
23750=     2  2X,I5,1X,I4,1X,I5,1X,I7,1X,I6,1X,I4,1X,I6,1X,I5,1X,F5.2)
23800=      IF(ILOG.NE.0.AND.ILOG.LE.2)
23900=     1 WRITE(ITRACE,959)ISTM,IEOJ,NN,ICLX,LASTI,LASTN
24000= 959  FORMAT(11H  NEXCLA-I ,10I5)
24100=      RETURN
24200=      END
```

```
80000=
80900=      SUBROUTINE KERPUT(CLAS,JDX,NEXC,IEOJ)
81000=      COMMON/KVEC/KK,ILIS(15),ISA
81100=      COMMON/TRACE/ILOG,ITRACE
81200=      COMMON/KFILE/ MAXKV,NCIX,NENT,MSCAL,MZAPD
81300=      INTEGER CLAS(JDX,1)
81400=      DATA LCLAS,KIX,IHD/0,0,1HD/
81500=      IEOJ=0
81600=C**      INSERTS ILIS VECTOR NUMBERS INTO KERVEC
81700=      JD=JDX-3
81800=      KIX=0
81900=      IF(NEXC.EQ.LCLAS) GOTO 10
82000=C****    SETUP AT NEWC FIND LAST KERVEC
82100=      KTYP=2
82200=      NAME=NEXC*1000+301
82300=      CALL RIX(NAME,CLAS(1,NCIX),NENT,NPOS,IEOJ)
82400=      IF(IEOJ.EQ.0) GOTO 10
82500=      IF(ISA.NE.IHD) GOTO 35
82600=      IF(KK.EQ.0) GOTO 60
82700=      GOTO 35
82800=C****    INSERT KVECS FROM ILIS
82900= 10   CONTINUE
83000=      KERP=10
83100=      IF(ILOG.NE.0.AND.ILOG.LE.2)
83200=     1  WRITE(ITRACE,957)KERP,KIX,KK,NEXKE,NAME,NEXC,NPOS
83300=      IF(ISA.EQ.IHD) GOTO 50
83400=C**      ENSURE ASCENDING ORDER OF ENTRIES
83500=      LASTN=CLAS(JD+1,NPOS)-1
83600=      IF(LASTN.LT.1) GOTO 20
83700=      IF(ILIS(1).LE.CLAS(LASTN,NPOS)) GOTO 45
83800=C**      INSERT INTO EXISTING KERVEC
83900= 20   CONTINUE
84000=      KERP=20
84100=      IF(ILOG.NE.0.AND.ILOG.LE.2)
84200=     1  WRITE(ITRACE,957)KERP,KIX,KK,NEXKE,NAME,NEXC,NPOS
84300=      IF(KIX.GE.KK) GOTO 60
84400=      IF(ILIS(KIX+1).LE.0) GOTO 60
84500=      NEXKE=CLAS(JD+1,NPOS)
84600=      IF(NEXKE.GT.JD) GOTO 30
84700=      KIX=KIX+1
84800=C**      BUMP NEXKE AND INSERT NVEC
84900=      CLAS(NEXKE,NPOS)=ILIS(KIX)
85000=      CLAS(JD+1,NPOS)=CLAS(JD+1,NPOS)+1
85100=      GOTO 20
85200=C*****;   CREATE ANOTHER  HUSK  VECTOR
85300=C*****;   A MIN OF 1 IS ALWAYS OK(MAXKV=0)
85400= 30   CONTINUE
85500=      KERP=30
85600=      IF(ILOG.NE.0.AND.ILOG.LE.2)
85700=     1  WRITE(ITRACE,957)KERP,KIX,KK,NEXKE,NAME,NEXC,NPOS
85800=      KTYP=CLAS(JD+2,NPOS)
85900=      LIM=MAXKV+3
86000=      IF(KTYP.GT.LIM) GOTO 40
86100=      IF(NEXC.EQ.LCLAS) NAME=NAME+1
86200=  35  CONTINUE
86300=      LCLAS=NEXC
86400=      CALL ADD(NAME,CLAS(1,NCIX),NENT,NPOS,IEOJ)
86500=      IF(IEOJ.GT.0) GOTO 60
86600=C**      SET NEXKE ,NEXC,AND KTYP IN CLAS
86700=      CLAS(JD+1,NPOS)=1
86800=      CLAS(JD+2,NPOS)=KTYP+1
86900=      CLAS(JD+3,NPOS)=NEXC
87000=      GOTO 20
87100=C*****;   TOO MANY  HUSK  VECTORS
87200= 40   CONTINUE
87300=      IEOJ=4
87400=      GOTO 60
87500=C**      ENTRIES OUT OF ORDER
87600=  45  CONTINUE
87700=      IEOJ=6
87800=      GOTO 60
87900=C*****;   DELETE ALL  HUSK  VECTORS BEFORE INSERT
88000= 50   CONTINUE
88100=      LNAME=NEXC*1000+300
88200= 55   CONTINUE
88300=      KERP=55
88400=      IF(ILOG.NE.0.AND.ILOG.LE.2)
88500=     1  WRITE(ITRACE,957)KERP,KIX,KK,NEXKE,NAME,NEXC,NPOS,LNAME
88600= 957  FORMAT(2X,I2,6H  KERP  ,10I7)
88700=      IF(NAME.EQ.LNAME) GOTO 20
88800=      CALL DEL(NAME,CLAS(1,NCIX),NENT,NPOS,IEOJ)
88900=      IF(IEOJ.GT.0) GOTO 20
89000=      DO 58 J=1,JDX
89100= 58   CLAS(J,NPOS)=0
89200=      NAME=NAME-1
89300=      GOTO 55
89400=C*****;   END OF CALL
89500= 60   CONTINUE
89600=      IF(ILOG.NE.0.AND.ILOG.LE.2)
89700=     1  WRITE(ITRACE,957)KERP,NEXC,LCLAS,NAME,NENT,NPOS,
89800=     2  KIX,NEXKE,KTYP
89900=      RETURN
90000=      END
```

```
27600=      SUBROUTINE CLASSX(FEAT,CLAS,HISC,JDX,NHX,IEOJ)
27700=      COMMON/NAMES/INPF,IOUTF,IFEAT,IOLDC,INEWC,ILOGF,IHISC
27800=      COMMON/TRACE/ILOG,ITRACE
27900=      COMMON/CNUM/NEXC,NTC
28000=      COMMON/CONTRL/ IMOD,IALL,ISYM,ICAL,IPIC,IH,IS,
28100=     1   IUKER,LABLC,LABLF,SIZE,IPK
28200=      COMMON/CREC/  LASV,IEOFR,NEPR,LB,MAXV
28300=      COMMON/CLID/NCAV,NCSDL,NCSDR,NCMAX,NCMIN
28400=      COMMON/DIST/FIRS,FLAS,FIRC,FLAC,NDUC,KTR,NI
28500=      COMMON/KFILE/MAXKV,NCIX,NENT,MSCAL,MZAPD
28600=      DIMENSION FEAT(JDX,1),CLAS(JDX,1),HISC(NHX,1)
28700=      EQUIVALENCE(IZER,ZER),(IONE,ONE),(ITWO,TWO)
28800=      EQUIVALENCE(FEXC,NEXC)
28900=      DATA IZER,IONE,ITWO/0,1,2/,FMARK,IHI/.999E99,1HI/
29000=C**             INITIALIZE LIST.IX ON FIRST ENTRY
29100=      IF(IMOD.EQ.IHI.AND.IGO.NE.1)
29200=     1    CALL INITC(NENT,CLAS(1,NCIX),0)
29210=      IF(IGO.EQ.1.OR.IMOD.EQ.IHI) GOTO 210
29212=      CALL RIX(99000,CLAS(1,NCIX),NENT,NCMAX,IEOJ)
29214=      IF(IEOJ.GT.0) GOTO 205
29216=      CALL RIX(98000,CLAS(1,NCIX),NENT,NCMIN,IEOJ)
29218=      IF(IEOJ.GT.0) GOTO 205
29220=      GOTO 210
29222=C**      NO MIN MAX VECTORS
29224= 205    CONTINUE
29226=      IEOJ=6
29228=      GOTO 490
29230=C**      BEGIN NORMAL PROCESS
29232= 210   CONTINUE
29300=      INIT=0
29400=      IGO=1
29500=      JD=JDX-3
29600=C**      FEEDS FEAT RECORD PAGES TO OPTED FUNCTIONS
29700=C**      INITIALIZE FOR COMPUTATIONS THIS CLASS
29800=C**            SET UP POINTERS TO PROTOTYPES
29900=      NAMC=NEXC*1000+100
30000=      NAML=NEXC*1000+201
30100=      NAMR=NEXC*1000+202
30200=C**      GET NCAV
30300= 101  CALL RIX(NAMC,CLAS(1,NCIX),NENT,NCAV,IEOJ)
30400=      IF(IEOJ.GT.0) GOTO 111
30500=      FLAC=CLAS(JD+1,NCAV)
30600=      GOTO 102
30700=C**               SETUP CAV
30800= 111  CALL ADD(NAMC,CLAS(1,NCIX),NENT,NCAV,IEOJ)
30900=      IF(IEOJ.GT.0) GOTO 300
31000=       CLAS(JD+2,NCAV)=ZER
31100=      CLAS(JD+3,NCAV)=FEXC
31200=C**      GET NCSDL
31300= 102  CALL RIX(NAML,CLAS(1,NCIX),NENT,NCSDL,IEOJ)
31400=      IF(IEOJ.GT.0) GOTO 112
31500=      FIRC=CLAS(JD+1,NCSDL)
31600=      GOTO 103
31700=C**               SETUP CSDL
31800= 112  CALL ADD(NAML,CLAS(1,NCIX),NENT,NCSDL,IEOJ)
31900=      IF(IEOJ.GT.0) GOTO 300
32000=      CLAS(JD+2,NCSDL)=ONE
32100=      CLAS(JD+3,NCSDL)=FEXC
32200=C**      GET NCSDR
32300= 103  CALL RIX(NAMR,CLAS(1,NCIX),NENT,NCSDR,IEOJ)
32400=      IF(IEOJ.GT.0) GOTO 109
32500=C**       FOUND CSDR
32600=      IF(ISYM.EQ.2) GOTO 104
32700=      IF(ISYM.EQ.1) GOTO 105
32800=      IF(ISYM.EQ.0) GOTO 107
32900=C**       EXPECTED CSDR
33000= 104  FLAC=CLAS(JD+1,NCSDR)
33100=      GOTO 10
33200=C**       NO CSDR EXPECTED
33300= 105  CONTINUE
33400=      CALL DEL(NAMR,CLAS(1,NCIX),NENT,NPOS,IEOJ)
33500=      DO 106 J=1,JDX
33600= 106  CLAS(J,NPOS)=0
33700=      GOTO 10
33800=C**       SETUP FOR FANDER
33900= 107  CONTINUE
34000=      ISYM=-2
34100=      GOTO 10
34200=C**       FOUND NO CSDR
34300= 109  CONTINUE
34400=      IF(ISYM.GT.1) GOTO 113
34500=C**       NONE EXPECTED
34600=      GOTO 10
34700=C**               SETUP NCSDR
34800= 113  CALL ADD(NAMR,CLAS(1,NCIX),NENT,NCSDR,IEOJ)
34900=      IF(IEOJ.GT.0) GOTO 300
35000=      CLAS(JD+2,NCSDR)=TWO
35100=      CLAS(JD+3,NCSDR)=FEXC
35200= 110  CONTINUE
```

```
35360=C**
35400= 10   CONTINUE
35500=      IF(ILOG.NE.0.AND.ILOG.LE.2)
35600=     1   WRITE(ITRACE,957)NAMC,NAML,NAMR,NCAV,NCSDL,NCSDR
35700= 957  FORMAT(8H  CLASS ,6I9)
35800=C**      REINITIALIZE FOR EA CLASS
35900=      DO 1 J=1,JD
36000=      DO 2 N=1,NHX
36100= 2    HISC(N,J)=0.0
36200= 1    CONTINUE
36300=      IEOJ=0
36400=C**      READ AND PROCESS FEAT RECORD
36500= 5    CALL RFEAT(FEAT,JDX,INIT,IEOJ)
36600=      IF(IEOJ.GT.0) GOTO 400
36700=      IF(ILOG.EQ.1)WRITE(ILOGF,978)NEXC,NBPR,LASV,
36800=     1   (FMARK,(FEAT(M,N),M=1,JD),N=1,LASV)
36900= 978  FORMAT(1X,6H CLAS ,I4,10H FEAT BLK ,I4,I5/((((10(1XE9.3)))))
37000=      IF(IEOJ.GT.0) GOTO 400
37100=      IF(IPIC.NE.0.AND.IALL.NE.1)
37200=     1   CALL FANDER(CLAS,JDX,FEAT,IEOJ)
37300=C#    IF(ICAL.NE.0) CALL SHUCK(FEAT,CLAS,JDX,IEOJ)
37500=      IF(ISYM.GT.0) CALL CDEFI(FEAT,CLAS,HISC,JDX,NHX,IEOJ)
37600=      IF(IEOFR.EQ.0) GOTO 5
37700=C**      COMPLETE PROCESSING THIS CLASS
37800=C#    IF(IPIC.NE.0.AND.IALL.NE.1) CALL PICKER(FEAT,CLAS,JDX,IEOJ)
37900=C**   ***        INSERT CAV,CSDL,CSDR INTO HISC 2,3,4
37950=C**   ***        FIRC,FLAC ARE CLASS 'OVERALL' MIN,MAX VALS
38000=C**   ***        FIND FIRC,FLAC,IN HISC AND SAVE IN CLAS
38100=      IF(ISYM.LE.0) GOTO 400
38200=      JL=6
38300=      JR=KTR+5
38400=      FRL=1E10
38500=      FLL=1E10
38600=      FRH=-1E10
38700=      FLH=-1E10
38800=C**      NUM,CAV,CSDL,CSDR(MBUC),MIN,MAX,MODE,VMODE
38900=C**      (FLR,FLH),(FRL,FRH) BOUND LEFT/RIGHT DISTRIBS
39000=      DO 30 J=1,JD
39100=      HISC(2,J)=CLAS(J,NCAV)
39200=      HISC(3,J)=CLAS(J,NCSDL)
39400=        HISC(4,J)=CLAS(J,NCSDR)
39500=      IF(HISC(6,J).GT.FLH)FLH=HISC(6,J)
39600=      IF(HISC(5,J).LT.FLL)FLL=HISC(5,J)
39800=      IF(ISYM.NE.2) GOTO 30
40000=      IF(HISC(JR,J).GT.FRH)FRH=HISC(JR,J)
40100=      IF(HISC(JR-1,J).LT.FRL)FRL=HISC(JR-1,J)
40200=        HISC(JR-2,J)=CLAS(J,NCSDR)
40300=      HISC(JR-3,J)=CLAS(J,NCSDL)
40400=      HISC(JR-4,J)=CLAS(J,NCAV)
40600= 30   CONTINUE
40650=C**      SAVE CLASS-LOCAL MIN,MAX IN CAV,CSD(L AND R)
40700=      CLAS(JD+1,NCAV)=FLH
40800=      FLAC=FLH
40900=      CLAS(JD+1,NCSDL)=FLL
41000=      FIRC=FLL
41100=      IF(ISYM.LE.1) GOTO 33
41200=      CLAS(JD+1,NCSDR)=FRH
41300=      FLAC=FRH
41400= 33   CONTINUE
41500=C**          CHECK THAT RL(--->)LH HERE
41600=C**
41700=      IF(IH.EQ.1)
41800=     1  CALL WRHIS(HISC,NEXC,ISYM,LABLC,NBUC,KTR,NI,NHX,JDX)
41900=      GOTO 400
42000=C**         NO  MORE ROOM ERROR
42100= 300  CONTINUE
42200=      IEOJ=5
42300=C**                  NORMAL EXIT
42400= 400  CONTINUE
42500=      IF(ILOG.NE.0.AND.ILOG.LE.2)
42600=     1  WRITE(ITRACE,958)LASV,IEOFR,NBPR,LB,NCAV,NCSDL,NCSDR,
42700=     1 NAMC,NAML,FIRC,FLAC,FRL,FRH,FLL,FLH
42800= 958  FORMAT(9H  CLASSX ,1015,2X,6E9.3)
42900=      RETURN
43000=      END
```

```
49400=
49500=
49600=      SUBROUTINE CDEFI(FEAT,CLAS,HISC,JDX,NHX,IEOJ)
49700=C
49800=      COMMON/NAMES/INPF,IOUTF,IFEAT,IOLDC,INEWC,ILOGF,IHISC
49900=      COMMON/TRACE/ILOG,ITRACE
50000=      COMMON/CNUM/NEXC,NTC
50100=      COMMON/CONTRL/ IMOD,IALL,ISYM,ICAL,IPIC,IH,IS,
50200=     1    IUKER,LABLC,LABLF,SIZE,IPK
50300=      COMMON/DIST/FIRS,FLAS,FIRC,FLAC,NBUC,KTR,NI
50400=      COMMON/CREC/  LASV,IEOFR,NBPR,LB,MAXV
50500=      COMMON/CLID/NCAV,NCSDL,NCSDR,NCMAX,NCMIN
50600=      DIMENSION FEAT(JDX,1),CLAS(JDX,1),HISC(NHX,1)
50700=      EQUIVALENCE(AVEC,NVEC)
50800=C**     SET UP CONSTANTS ON FIRST CALL
50900=      IF(ICO.EQ.1) GOTO 90
51000=      NI=(NBUC-8)/2
51100=      KTR=NHX/2  +1
51200=      ICO=1
51250=      JD=JDX-3
51300= 90   CONTINUE
51400=C**           INIT TO BEGIN NEW CLASS
51550=      LAS=0
51575=      IF(IUKER.GT.0.AND.NBPR.EQ.1)
51600=     1   NKER=KERGET(CLAS,JDX,NEXC,IEOJ)
51700=      IEOJ=0
51800=      L=0
52000=C**           UPDATES CLAS(NEXC) I WITH
52100=C**           FEATURE VECTORS FROM THIS BLOCK
52200= 100  CONTINUE
52300=      L=L+1
52400=      IF(L.GT.LASV) GOTO 200
52500=      AVEC=FEAT(JD+1,L)
52600=      IF(NVEC.GT.NKER.AND.NKER.GT.0)
52700=     1 NKER=KERGET(CLAS,JDX,NEXC,IEOJ)
52800=C**           SKIP VECTOR LISTED IN  HUSK
52900=      IF(NVEC.EQ.NKER.AND.IUKER.EQ.1) GOTO 100
53000=C
53100=C**     CONTROLS ASSYMETRIC SDEV CALCS
53200=C**     COMPUTES VECTOR STATS AND HISTOGRAM
53300=C**     MAY BE USED FOR MODE BASED STATS
53400=C**
53450=      IF(L.EQ.LASV.AND.IEOFR.EQ.1) LAS=1
53500=          NB=NBUC
53600=          IF(ISYM.GT.1) NB=NI
53700=          DO 190 J=1,JD
53800=          KT=1
53900=          I1=9
54000=          BLO=FIRS
54100=          BHI=FLAS
54200=          IF(ISYM.LT.2) GOTO 120
54300=C**                 SETUP FOR ASSYMETRIC BOX OPTION
54400=          IF(FEAT(J,L).GT.CLAS(J,NCAV)) GOTO 110
54500=C**                 NEGATIVE IS LEFT SIDE
54600=          BHI=FLAC
54700=          CALL STATX(0,FEAT(J,L),HISC(1,J),CLAS(J,NCAV),
54800=     1       CLAS(J,NCSDL),1)
54900=          GOTO 130
55000=C**                 POSITIVE IS RIGHT SIDE
55100= 110      CONTINUE
55200=          BLO=FIRC
55300=          I1=KTR+8
55400=          KT=KTR
55500=          CALL STATX(0,FEAT(J,L),HISC(KT,J),CLAS(J,NCAV),
55600=     1       CLAS(J,NCSDR),1     )
55700=          GOTO 130
55800=C**                 DEFINE SYMMETRIC CLASS
55900= 120      CONTINUE
56000=          CALL STATX(0,FEAT(J,L),HISC(KT,J),CLAS(J,NCAV),
56100=     1         CLAS(J,NCSDL),0)
56200= 130      CONTINUE
56210=          IF(LAS.NE.1) GOTO 140
```

```
56212=C**                LAST VECTOR OF CLASS
56215=          IF(ISYM.GT.1) GOTO 135
56220=          CALL STATX(1,0.0,HISC(1,J),CLAS(J,NCAV),
56225=     1        CLAS(J,NCSDL),0)
56227=          GOTO 140
56228=C**                  ASSYMETRIC LAST VEC
56230= 135      CONTINUE
56240=          CALL S ATX(1,0.0,HISC(KTR,J),CLAS(J,NCAV),
56245=     1        CLAS(J,NCSDR),1)
56247=          CALL STATX(1,0.3,HISC(1,J),CLAS(J,NCAV),
56248=     1       CLAS(J,NCSDL),1)
56250= 140      CONTINUE
56270=          IF(IH.NE.1) GOTO 190
56300=          CALL STATH(0,FEAT(J,L),HISC(KT,J),
56400=     1         HISC(I1,J),NB,BLO,BHI)
56410=          IF(LAS.NE.1) GOTO 190
56412=C**                LAST VECTOR OF CLASS
56415=          IF(ISYM.GT.1) GOTO 185
56420=          CALL STATH(1,0.0,HISC(1,J),HISC(9,J),NB,BLO,BHI)
56430=          GOTO 190
56431=C**               ASSYMETRIC LAST VEC
56435= 185       CONTINUE
56437=          CALL STATH(1,0.0,HISC(1,J),HISC(9,J),NB,FIRS,FLAC)
56439=          CALL STATH(1,0.0,HISC(9,J),HISC(KTR+8,J),NB,FIRC,FLAS)
56500= 190      CONTINUE
56600=C**         NOTE THAT AV,SDL,SDR, ARE NOT YET IN HISC ARRAY
56700=      GOTO 100
56800=C**               NORMAL EXIT
56900=C**
57000= 200  CONTINUE
57100=      IF(ILOG.NE.0.AND.ILOG.LE.2)
57200=     1  WRITE(ITRACE,957)LASV,NVEC,NKER,NEXC,IEOJ
57300= 957  FORMAT(8H  CDEFI ,8I5)
57400=      IF(IEOFR.NE.1) GOTO 300
57500=      FMARK=.999E99
57600=      NR=NCSDR
57700=      IF(ISYM.LT.2)NR=NCSDL
57800=      IF(ILOG.GE.2)WRITE(ILOGF,983)NEIC,ISYM,
57900=     1 ((CLAS(J,NCAV),J=1,JD),FMARK,
58000=     2 (CLAS(J,NCSDL),J=1,JD),FMARK,
58100=     3 (CLAS(J,NR),J=1,JD)  ,FMARK)
58200= 983  FORMAT(1X,15H CLAS DEFN FOR ,I4,6H SYM= ,I2/
58300=     1 (((10(1XE9.3)))) )
58400= 300  CONTINUE
58500=      RETURN
58600=      END
```

```
98200=
98300=
98400=      SUBROUTINE FANDER(CLAS,JDX,FEAT,IEOJ)
98500=C**       PRODUCES COMPONENT CLUSTER PLOT
98600=      COMMON/TRACE/ILOG,ITRACE
98700=      COMMON/CNUM/NEXC,NTC
98750=      COMMON/PCLUS/ICPS,IQUIK,ISCF
98800=      COMMON/CONTRL/ IMOD,IALL,ISYM,ICAL,IPIC,IH,IS,
99000=     1   IUKER,LABLC,LABLF,SIZE,IPK
99000=      COMMON/DIST/FIRS,FLAS,FIRC,FLAC,KBUC,KTR,NI
99100=      COMMON/KFILE/ MAXKV,NCIX,NENT,MSCAL,MZAPD
99200=      COMMON/NAMES/INPF,IOUTF,IFEAT,IOLDC,INEWC,ILOGF,IHISC
99300=      COMMON/CREC/  LASV,IEOFR,NDPR,LB,MAXV
99400=      COMMON/CLID/NCAV,NCSDL,NCSDR,NCMAX,NCMIN
99500=      DIMENSION CLAS(JDX,1),FEAT(JDX,1)
99600=      DIMENSION KHD(9)
99650=      EQUIVALENCE(AVEC,NVEC)
99700=      DATA KHD(1),KHD(2),KHD(3),KHD(4),KHD(5),KHD(6),
99800=     1  KHD(7),KHD(8),KHD(9)/
99900=     2  4HUNIV,4HERSE,4H    ,4HCLAS,4HS   ,4H    ,
100000=    3  4HMANA,4HLANA,4HOBIS/
100100=     IF(IGO.EQ.1) GOTO 1
100200=     IGO=1
100300=     CALL INITT(ICPS)
100400= 1   CONTINUE
100500=     JD=JDX-3
100600=     IF(NEXC.EQ.LASC) GOTO 100
100700=C**      INITIALIZE NEW CLASS
100800=     CALL ERASE
101300= 900 FORMAT(2X,32H CLASS FEATURES CLUSTER ,CLASS= ,I4/
101400=    1  2X,10H SCALE IS , 3A4/
101500=    2  2X,6H MIN= ,E9.3,3X,6H MAX= ,E9.3)
101600=C**      DRAW MASTER Y-AXIS
101700=     IX=100
101800=     IY=650
101900=     CALL MOVABS(IX,IY)
102000=     CALL MARK(IX,IY,3,0)
102100=     DO 10 J=1,11
102200=     IY=IY-50
102300=     CALL DRWABS(IX,IY)
102400=     CALL MARK(IX,IY,3,0)
102500= 10  CONTINUE
102600=C**      DRAW X-AXIS
102700=     IX=100
102800=     CALL MOVABS(IX,IY)
102900=     CALL MARK(IX,IY,0,3)
103000=     INCX=600/JD
103050=     MOD=5
103100=     DO 20 J=1,JD
103200=     IX=IX+INCX
103300=     CALL MOVABS(IX,IY)
103400=     CALL MARK(IX,IY,0,3)
103500=     IF((J/MOD).NE.1) GOTO 20
103510=     MOD=MOD+5
103520=     CALL MARK(IX,IY,3,0)
103600= 20  CONTINUE
103700=     CALL MOVABS(100,60)
103800=     CALL MOVABS(100,60)
103810=C**      MARK MEAN POSITION
103820=     IF(ISCF.NE.2) GOTO 25
103830=     CALL MOVABS(90,400)
103840=     CALL MARK(90,400,3,3)
103850=     CALL MARK(90,400,3,0)
103860= 25  CONTINUE
103862=     CALL MOVABS(100,60)
103864=     CALL MOVABS(100,60)
103866=     CALL PNTABS(100,60)
103900=     CALL ANMODE
104020=     WRITE(IOUTF,901)
104100= 901 FORMAT(2X,19H FEATURE COMPONENTS)
104200=C**      SETUP KERNEL
104300=     IF(IUKER.EQ.1)NKER=KERGET(CLAS,JDX,NEXC,IEOJ)
```

```
104400=C**   SETUP CLASS PROCESS
104500=      LASC=NEXC
104600=C**      SETUP CONSTANTS
104700=      IF(ISCF.NE.0) GOTO 30
104800=C**      GLOBAL SCALES
104900=      BLO=FIRS
104920=      K1=3*ISCF+1
104930=      K2=K1+2
104950=      BHI=FLAS
105000=      BSPAN=FLAS-FIRS
105100=      GOTO 50
105200= 30   IF(ISCF.NE.1) GOTO 40
105300=C**      LOCAL SCALES
105400=      BLO=FIRC
105450=      BHI=FLAC
105500=      BSPAN=FLAC-FIRC
105600=      GOTO 50
105700= 40   IF(ISCF.NE.2)GOTO 50
105800=      BLO=-6.
105850=      BHI=6.
105900=      BSPAN=12.
106000= 50   CONTINUE
106050=C**      WRITE PAGE HEADER
106052=      CALL MOVABS(100,740)
106053=      CALL MOVABS(100,740)
106054=      K1=3*ISCF+1
106055=      K2=K1+2
106056=      CALL ANMODE
106058=      WRITE(IOUTF,900) NEXC,(KHD(K),K=K1,K2),BLO,BHI
106100=C***     PLOT VECTORS
106200=C**
106300= 100  CONTINUE
106400=      LK=.3*FLOAT(LASV)
106500=      DO 350 L=1,LASV
106600=      L1=L/LK
106700=      L1=L-L1*LK
106800=      IF(L1.NE.0) GOTO 60
106900=C**       EARLY EXIT TEST
107000=      CALL MOVABS(800,800)
107100=      CALL DCURSR(ICH,IX,IY)
107200=      CALL ANMODE
107400=      IF(ICH.NE.32) GOTO 400
107500= 60   CONTINUE
107600=      AVEC=FEAT(JD+1,L)
107700=      IF(NVEC.GT.NKER.AND.NKER.GT.0)
107800=     1  NKER=KERGET(CLAS,JDX,NEXC,IEOJ)
107900=      IF(NVEC.EQ.NKER.AND.IUKER.EQ.1) GOTO 300
```

```
108000=C**
108100=C**      PLOT COMPONENTS
108110=      IX=100
108120=      IY=100
108200=      DO 200 J=1,JD
108300=         F=FEAT(J,L)
108400=         IF(ISCF.NE.2) GOTO 120
108500=            NCS=NCSDL
108600=            T=F-CLAS(J,NCAV)
108700=            IF((T.GT.0.).AND.IABS(ISYM).EQ.2) NCS=NCSDR
108800=            F=T/CLAS(J,NCS)
108900= 120     CONTINUE
109000=         IY=((F-BLO)/BSPAN)*600.+100.
109100=         IX=IX+INCX
109200=         IF(J.EQ.1) GOTO 190
109300=         IF(IQUIK.EQ.1) CALL MARK(IX,IY,2,0)
109400=         IF(IQUIK.EQ.0) CALL DRWABS(IX,IY)
109500= 190     CONTINUE
109600=         CALL MOVABS(IX,IY)
109700= 200     CONTINUE
109810=C**      PAUSE FOR COPY
109812= 300  CONTINUE
109815=      CALL MOVABS(800,100)
109820=      CALL DCURSR(ICH,IX,IY)
109900= 400  CONTINUE
110000=      IF(IEOFR.NE.1) GOTO 600
110100=      LASC=0
110150=      CALL ERASE
110152=C**      WAIT LOOP
110153=      DO 444 J=1,10000
110154= 444  CONTINUE
110156=      CALL ANMODE
110200=C**      EXIT ROUTINE
110300= 600  CONTINUE
110400=      IEOJ=0
110500=      IF(ILOG.NE.0.AND.ILOG.LE.2)
110600=     1   WRITE(ITRACE,957)LASC,IEOJ,NKER,NEXC,ICH
110700= 957  FORMAT(10H FANDER  ,8I5)
110800=      RETURN
110900=      END
```

```
151700=
151800=      SUBROUTINE SETLIM(CLAS,FEAT,JDX,IEOJ)
151850=      COMMON/NAMES/INPF,IOUTF,IFEAT,IOLDC,INEWC,ILOGF,IHIST
151900=      COMMON/CLID/NCAV,NCSDL,NCSDR,NCMIN,NCMAX
152000=      COMMON/KFILE/MAXKV,NCIX,NENT,MSCAL,MZAPD
152050=      DIMENSION CLAS(JDX,1),FEAT(JDX,1)
152100=      EQUIVALENCE (M1,FM1),(M2,FM2),(N98,F98),(N99,F99)
152200=      DATA M1,M2,N98,N99,F0/-1,-2,98,99,0./
152300=      INIT=2
152310=      JD=JDX-3
154500=C**      INSERT MIN AND MAX VECTOR INTO CLAS FILE
154600=      CALL ADD(98000,CLAS(1,NCIX),NENT,NCMIN,IEOJ)
154700=      IF(IEOJ.GT.0) GOTO 300
154702=      CALL ADD(99000,CLAS(1,NCIX),NENT,NCMAX,IEOJ)
154704=      IF(IEOJ.GT.0) GOTO 300
154710=C**   READ VECTOR COMPONENTS INTO CLAS FILE VECTORS
154720=      CALL RFEAT(CLAS(1,NCMIN),JDX,INIT,IEOJ)
154800=      CLAS(JD+1,NCMIN)=F0
154900=      CLAS(JD+2,NCMIN)=FM2
155000=      CLAS(JD+3,NCMIN)=F98
155400=      CLAS(JD+1,NCMAX)=F0
155500=      CLAS(JD+2,NCMAX)=FM1
155600=      CLAS(JD+3,NCMAX)=F99
155700=C**      NORMAL EXIT
155800= 200  CONTINUE
155900=      RETURN
156000=C**      NO MORE ROOM ERROR
156100= 300  CONTINUE
156200=      IEOJ=5
156300=      GOTO 200
156400=      END
```

```
24300=
24400=
24500=      SUBROUTINE PHUSK(CLAS,JDX,IEOJ)
24600=      COMMON/KVEC/KK,ILIS(15),ISA
24700=      COMMON/KFILE/MAXKV,NCIX,NENT,MSCAL,MZAPD
24800=      COMMON/NAMES/INPF,IOUTF,IFEAT,IOLDC,INEWC,IHISC
24900=      COMMON/CNUM/NEXC,NTC
25000=      INTEGER CLAS(JDX,1)
25100=C**      PRINTS HUSK FOR CURRENT CLASS
25200=      KIX=1
25300=      WRITE(IOUTF,901) NEXC
25400=      WRITE(ILOGF,901) NEXC
25500= 901  FORMAT(2X,14HHUSK FOR CLAS=,I3 )
25600= 5    NUMH=KERGET(CLAS,JDX,NEXC,IEOJ)
25700=      IF(IEOJ.GT.0) GOTO 20
25800=      IF(NUMH.LE.0) GOTO 20
25900=      ILIS(KIX)=NUMH
26000=      KIX=KIX+1
26100=      IF(KIX.LE.15) GOTO 5
26200=      WRITE(IOUTF,900) ILIS
26300=      WRITE(ILOGF,900) ILIS
26400= 900  FORMAT(2X,2HP= ,15(I3,1X)  )
26500=      GOTO 5
26900=C**      NORMAL EXIT
27000= 20   CONTINUE
27100=      RETURN
27200=      END
```

APPENDIX G

TRYOUT Module

```
  100=      PROGRAM TRYOUT(FEAT,OLDC,NEWC,LOGF=64,INPUT=64,OUTPUT=64)
  200=C***              27 OCT 78 VERSION JRL -- TRY00
  300=      COMMON WO(8200)
  400=      COMMON/TRACE/ILOG,ITRACE
  500=      COMMON/PARAM/MEMC,MEM,NAX,JDX,ICX,NC1,NF1,NA1,IC
  600=      COMMON/NAMES/INPF,IOUTF,IFEAT,IOLDC,INEWC,ILOGF,IHISC
  700=      COMMON/CONX/LCID(51,3),SCALE(2),ICONF,ANS(3),IMAG
  800=      COMMON/TRY/IFIG,ISUB,NT1,ND1,ND2,LABLC
  900=      MEM=8200
 1000=      DO 5 M=1,MEM
 1100=      WO(M)=0.
 1200= 5    CONTINUE
 1300=C**       ***** EVALUATE CLASS PROTOTYPES
 1400=      CALL DEFT(IEOJ)
 1500=      IF(IEOJ.GT.0) GOTO 800
 1600=      CALL LOADC(WO(NC1),JDX,ICX)
 1700=      CALL INDEX(WO(NC1),JDX,ICX,IEOJ)
 1800= 10   CONTINUE
 1900=      IF(IFIG.NE.1) GOTO 12
 2000=      CALL MERIT(WO(NT1),WO(ND1),WO(ND2),WO(NC1),JDX)
 2100= 12   CONTINUE
 2200=      IF(ISUB.EQ.1)CALL SUBSET(WO(NC1),JDX,ICX,IEOJ)
 2300=      IF(IEOJ.GT.0) GOTO 800
 2400=      IF(IFIG.GT.0)CALL FIGM(WO(NT1),JDX,IEOJ)
 2500=      IF(IEOJ.GT.0) GOTO 800
 2600=      CALL EVAL(WO(NF1),WO(NC1),WO(NA1),JDX,NAX,IC,IEOJ)
 2700=      IF(IEOJ.GT.6) GOTO 800
 2800=      CALL DOCU(WO(NA1),WO(NC1),JDX,NAX,IC)
 2900=C***              DUMP CLAS FILE IF NOT YET DONE
 3000=      IF(ILOG.GT.2)
 3100=     1  CALL PRCLAS(WO(NC1),WO(NC1),WO(NC1),JDX,ICX)
 3150=      IF(ISUB+IFIG.LT.1)GOTO 850
 3200=C***              RESET FOR NEXT TRY
 3300=      INIT=1
 3400=      CALL RFEAT(WO(NF1),JDX,INIT,IEOJ)
 3500=      GOTO 12
 3600=C**       ***** PRINT ERROR MESSAGES
 3700= 800  CONTINUE
 3800=      IF(ISUB.EQ.1) CALL WRCLAS(WO(NC1),JDX,ICX,0,LABLC)
 3900=      IF(IEOJ.EQ.1)WRITE(IOUTF,900)
 4000= 900  FORMAT(2X,1H )
 4100=      IF(IEOJ.EQ.2)WRITE(IOUTF,901)
 4200= 901  FORMAT(2X,13H  QUIT TRYOUT )
 4300=      IF(IEOJ.EQ.3)WRITE(ILOGF,902)
 4400= 902  FORMAT(2X,20H  TOO LITTLE MEMORY    )
 4500=      IF(IEOJ.EQ.4) WRITE(IOUTF,903)
 4600= 903  FORMAT(27H  CLAS AND FEAT NOT MATCHED)
 4610=C**           END JOB
 4620= 850  CONTINUE
 4700=      STOP
 4800=      END
```

```
5100=      SUBROUTINE DEFT(IEOJ)
5200=      COMMON/CNUM/NEXC,NTC
5300=      COMMON/NAMES/INPF,IOUTF,IFEAT,IOLDC,INEWC,ILOGF,IHISC
5400=      COMMON/TRACE/ILOG,ITRACE
5500=      COMMON/CONX/LCID(51,3),SCALE(2),ICONF,ANS(3),IMAG
5600=      COMMON/PARAM/MEMC,MEM,NAX,JDX,ICX,NC1,NF1,NA1,IC
5700=      COMMON/CREC/LASV,IEOFR,NBPR,LB,MAXV
5750=      COMMON/KFILE/MAXKV,NCIX,NENT,MSCAL,MZAPD
5800=      COMMON/TRY/IFIG,ISUB,NT1,ND1,ND2,LABLC
5900=      DIMENSION IOPNS(7)
6000=      DATA IHP,IHC,IHA,IHM,IHN,IHB,IHW,IHL,IHY,IHF,IH1,IH2,IHT,IHS
6100=     1 /1HP,1HC,1HA,1HM,1HN,1HB,1HW,1HL,1HY,1HF,1H1,1H2,1HT,1HS/
6200=
6300=      INPF=5LINPUT
6400=      IOUTF=6LOUTPUT
6500=      IFEAT=4LFEAT
6600=      INEWC=4LNEWC
6700=      IOLDC=4LOLDC
6800=      ILOGF=4LLOGF
6900=      ITRACE=4LLOGF
7000=      IEOJ=-1
7100=      NN=3
7200=C**           GET USER OPTIONS
7300=      WRITE(IOUTF,901)
7500= 901  FORMAT(9H  TRYOUT /15H  ENTER OPTIONS/2H *)
7600= 5    READ(INPF,903) IOPNS
7800= 904  FORMAT(1X,7A1)
7900= 903  FORMAT(7A1)
8000=      IFIG=0
8100=      ISUB=0
8200=      ICONF=-1
8300=      SCALE(1)=-1.
8400=      SCALE(2)=0.
8500=      DO 10 I=1,7
8600=      IF(IOPNS(I).EQ.IHL)ILOG=3
8700=      IF(IOPNS(I).EQ.IHT)ILOG=1
8800=      IF(IOPNS(I).EQ.IHY)ILOG=2
8900=      IF(IOPNS(I).EQ.IHP)ICONF=1
9000=      IF(IOPNS(I).EQ.IHC)ICONF=2
9100=      IF(IOPNS(I).EQ.IHA)ICONF=0
9200=      IF(IOPNS(I).EQ.IHN)SCALE(1)=1.
9300=      IF(IOPNS(I).EQ.IHB)SCALE(1)=256.
9400=      IF(IOPNS(I).EQ.IHW)SCALE(1)=32768.
9500=      IF(IOPNS(I).EQ.IHM)IMAG=1
9600=      IF(IOPNS(I).EQ.IHF) IFIG=1
9700=      IF(IOPNS(I).EQ.IHS)ISUB=1
9800= 10   CONTINUE
9810=      IF(ILOG.NE.0)WRITE(ILOGF,901)
9820=      IF(ILOG.NE.0)WRITE(ILOGF,904)IOPNS
9850=      IF(SCALE(1).GT.1.)MSCAL=1
9900=      IF(ICONF.LT.0.OR.SCALE(1).LT.0.) GOTO 30
10000=C**          OPEN DATA FILES
10100=     CALL OPENI(0,0,LABLC,LABLF,IEOJ)
10200=C**          SET MEMORY CONTROL PARAMETERS
10300=     NAX=NTC+1
10400=     NF1=1
10500=     NC1=NF1+JDX*LB
10600=     NA1=NC1+MEMC
10700=     NT1=NA1+NAX*IC
10800=     ND1=NT1+5*JDX
10900=     ND2=ND1+JDX
11000=     MEMK=MEM-(NAX*IC)-JDX*LB - JDX*ICX - JDX*7
11100=     IF(MEMK.LT.0) GOTO 25
11200=C**          NORMAL EXIT
11300= 20  CONTINUE
11400=     IF(ILOG.NE.0.AND.ILOG.LE.2)
11500=    1 WRITE(ITRACE,957)SCALE,NTC,IC,NAX,NF1,NC1,NT1,NA1,ILOG,ICONF
11600= 957 FORMAT(7H  DEFT ,2E9.3,9I6)
11700=     RETURN
11800=C**          TOO LITTLE MEMORY
11900= 25  CONTINUE
12000=     IEOJ=3
12100=     GOTO 20
12200=C**          ERROR RETRY
12300= 30  CONTINUE
12400=     CALL ERR(IOUTF,NN,IEOJ)
12500=     IF(IEOJ.LT.0) GOTO 5
12600=     GOTO 20
12700=     END
```

```
55200=
55300=
55400=      SUBROUTINE MERIT(FT,DISI,DISN,CLAS,JDX)
55500=      COMMON/NAMES/INPF,IOUTF,IFEAT,IOLDC,INEWC,ILOGF,IHISC
55600=      COMMON/TRACE/ILOG,ITRACE
55700=      COMMON/CONX/LCID(51,3),SCALE(2),ICONF,ANS(3),IMAG
55800=      COMMON/CNUM/NEXC,NTC
55900=      COMMON/SRCH/ISX,IKNO,NIC,JD,ITAG(50),DIS(50),RATE(2,50)
56000=      DIMENSION FT(JDX,5),CLAS(JDX,1)
56100=      DIMENSION DISN(JDX),DISI(JDX)
56200=      DATA LABLI,LABLU/4HICU ,4HUCI /
56300=C**
56400=C**          COMPUTES A FIGURE OF MERIT FOR EACH DIMENSION
56500=C**          PRINTS INTERMEDIATE MEASURES OF CLASS WORTH
56600=C**
56700=C**          FT(J,4)--> PRODUCT OF ICU VECTORS FOR CLASS I
56800=C**          FT(J,5)--> PRODUCT OF UCI VECTORS FOR CLASS I
56900=C**          FT(J,1)--> GRAND PRODUCT OF UCI*ICU VECTORS
57000=C**          FT(J,2)--> GRAND SUM OF FT(J,4)*FT(J,5)
57100=C**          FT(J,3)--> GRAND SUM OF ALL ICU AND UCI VECTORS
57200=C**          FOR ALL CLASS PAIRS
57300=C**
57400=      JD=JDX-3
57500=      DO 10 J=1,JD
57600=      FT(J,1)=1.
57700=      FT(J,4)=1.
57800=      FT(J,5)=1.
57900=      FT(J,2)=1.0
58000= 10   FT(J,3)=0.0
```

```
58100=C**           PICK EACH MEMBER OF CLAS FILE AS AN UNKNOWN
58200=      DO 200 I=1,NTC
58300=      ICAV=LCID(I,1)
58400=      ICSDL=LCID(I,2)
58500=      ICSDR=LCID(I,3)
58600=      ISYM=1
58700=      IF(ICSDR.GT.0) ISYM=2
58800=C**           LOOP THRU CLAS FILE DOING RECOGNITION
58900=      DO 150 N=1,NTC
59000=      IF(N.EQ.I) GOTO 150
59100=      NCAV=LCID(N,1)
59200=      NCSDL=LCID(N,2)
59300=      NCSDR=LCID(N,3)
59400=      NSYM=1
59500=      IF(NCSDR.GT.1) NSYM=2
59600=C**           CLAS I EVAL --> SUM: (DISI*DISN), N=1,NTC
59700=C**           DIM EVAL --> (PROD: DISI + PROD: DISN) N=I+1,NTC
59800=      DO 100 J=1,JD
59900=C**           CLEAR AND INITIALIZE ON FIRST N
60000=      IF(N.NE.1) GOTO 80
60100=      FT(J,4)=1.0
60200=      FT(J,5)=0.0
60300= 80   CONTINUE
60400=      DFEA=ABS(CLAS(J,ICAV)-CLAS(N,NCAV))
60500=C**           CLAS - UNKNOWN
60600=      IF(CLAS(J,ICAV).LT.CLAS(J,NCAV).OR.NSYM.EQ.1)
60700=     1 DISN(J)=DFEA/CLAS(J,NCSDL)
60800=      IF(CLAS(J,ICAV).GE.CLAS(J,NCAV).AND.NSYM.EQ.2)
60900=     1 DISN(J)=DFEA/CLAS(J,NCSDR)
61000=      IF(CLAS(J,NCAV).LT.CLAS(J,ICAV).OR.ISYM.EQ.1)
61100=     1 DISI(J)=DFEA/CLAS(J,ICSDL)
61200=      IF(CLAS(J,NCAV).GE.CLAS(J,ICAV).AND.ISYM.EQ.2)
61300=     1 DISI(J)=DFEA/CLAS(J,ICSDR)
61400=C**           GENERATE CLASS FERSPECTIVE FIGURES
61600=      FT(J,3)=FT(J,3)+DISI(J)+DISN(J)
61700=      FT(J,4)=FT(J,4)*DISI(J)
61800=      FT(J,5)=FT(J,5)+DISI(J)+DISN(J)
61900= 95   CONTINUE
62000= 100  CONTINUE
```

```
62100=C**           PROCESS EACH CLASS PAIR IF OPTED
62200=      IF(ILOG.NE.1) GOTO 110
62300=      VNMAG=SQRT(ENER(DISN,JD))
62400=      VIMAG=SQRT(ENER(DISI,JD))
62500=      IF(ILOG.NE.0)WRITE(ILOGF,900)LABLI,I,N,VNMAG,(DISN(J),J=1,JD)
62600=      IF(ILOG.NE.0)WRITE(ILOGF,900)LABLU,N,I,VIMAG,(DISI(J),J=1,JD)
62700= 900  FORMAT(2X,A4,2XI3,4H TO ,I3,6HVMAG= ,E8.2/((((10(1XE9.3)))))
62800= 110  CONTINUE
62900= 150  CONTINUE
63000=C**           PRINT MERIT FIGURES FOR EACH CLASS I
63050=      IF(ILOG.NE.3) GOTO 152
63100=      VNMAG=SQRT(ENER(FT(1,5),JD))
63200=      VIMAG=SQRT(ENER(FT(1,4),JD))
63300=      VKMAG=SQRT(ENER(FT(1,3),JD))
63400=      IF(ILOG.NE.0)WRITE(ILOGF,904) I,VKMAG,(FT(J,3),J=1,JD)
63500=      IF(ILOG.NE.0)WRITE(ILOGF,901)I,VNMAG,(FT(J,4),J=1,JD)
63600=      IF(ILOG.NE.0)WRITE(ILOGF,902)I,VIMAG,(FT(J,5),J=1,JD)
63700= 901  FORMAT(2X,6HCLASS ,I2,18H  AS SEEN BY REST ,8H  VMAG= ,E8.2/
63800=     1 ((((10(1XE9.3))) ) )
63900= 902  FORMAT(2X,23H  REST AS SEEN BY CLASS ,I2,8H  VMAG= ,E8.2/
64000=     1 ((((10(1XE9.3))) ) )
64100= 904  FORMAT(2X,I2,15H  CLASS MERIT= , 8H  VMAG= ,E8.2/
64200=     1 ((((10(1XE9.3)))) )
64300=C**           COMPUTE GRAND MERIT MATRIX
64350= 152  CONTINUE
64500=      DO 160 J=1,JD
64600=      T1=ALOG(FT(J,4))
64700=      FT(J,1)=FT(J,1)+T1
65100=      FT(J,2)=FT(J,2)*FT(J,5)
65200= 160  CONTINUE
65300=      IF(ILOG.NE.1) GOTO 200
65400=      DO 199 K=1,5
65500=      IF(ILOG.NE.0)WRITE(ILOGF,905) I,(FT(J,K),J=1,JD)
65600= 905  FORMAT(28H  INTERMEDIATE MERIT FIGURES,I4/((((10(1XE9.3)))) )
65700= 199  CONTINUE
65800= 200  CONTINUE
65900=C**           RETURNS,WITH FT(J,1)-->PLOG F/M
66000=C**           RETURNS WITH FT(J,2)-->PSUM F/M
66100= 300  CONTINUE
66200=      RETURN
66300=      END
```

```
66600=      SUBROUTINE FIGM(FT,JDX,IEOJ)
66700=      COMMON/NAMES/INPF,IOUTF,IFEAT,IOLDC,INEWC,ILOGF,IHISC
66800=      COMMON/SRCH/ISX,IKNO,NIC,JD,ITAG(50),DIS(50),RATE(2,50)
66900=      DIMENSION FT(JDX,5),MNAM(2)
67000=      DIMENSION NTAG(50)
67100=      DATA IHY/1HY/
67200=      DATA MNAM(1),MNAM(2)/4HILOG,4HZSUM/
68000=      IF(ISW.EQ.12345) GOTO 20
68100=      ISW=12345
68200=C**           OUTPUT FIGURE OF MERIT FOR EACH DIMENSION
68300=      DO 10 M=1,2
68400=      CALL FDSORT(FT(1,M),ITAG,DIS,JD)
68500=      IF(ILOG.NE.0)WRITE(ILOGF,901)MNAM(M),(ITAG(J),DIS(J),J=1,JD)
68600=      WRITE(IOUTF,901)MNAM(M),(ITAG(J),DIS(J),J=1,JD)
68700= 901  FORMAT(2X,A4,34H  FIGURES OF MERIT FOR DIMENSIONS    /
68800=     1 (((8( 2X,I2,1H=,E10.4) )))  )
68960= 10   CONTINUE
69000=C**           SUBSET FIG/MERIT-->SUBSPACE
69100= 20   CONTINUE
69200= 905  FORMAT(22H  ENTER F/M SET NUMBER,/2H *)
69300= 906  FORMAT(I1)
69400= 907  FORMAT(1X,I3,24H =FIG-MERIT SUBSPACE SET)
69500=      WRITE(IOUTF,905)
69600=      READ(INPF,906) N
69700=C***              N=  1,2--> SELECT F/M SUBSPACE
69800=C***                  0  --> USER SETS SUBSPACES
69900=C***                  3  --> END OF TRYOUT
69710=C***              KK=  1-JD  SUBSPACE TAG
69920=C***                   99    END OF INPUT LIST
69930=C***                   ELSE  ERROR
70000=      IF(N.EQ.3) GOTO 25
70100=      IF(N.LT.0.OR.N.GT.3) N=1
70200=      IF(N.EQ.0) GOTO 31
70300=      IF(ILOG.EQ.1.OR.ILOG.EQ.2)WRITE(ILOGF,907)N
70400=      CALL FDSORT(FT(1,N),ITAG,DIS,JD)
70500=      GOTO 50
70600=C***      END OF TRYOUT
70700= 25   CONTINUE
70800=      IEOJ=2
70900=      GOTO 50
```

```
71000=C**           USER RESET OF SUBSPACE TAGS LIST
71100= 31   CONTINUE
71200=      DO 32 I=1,50
71300=      NTAG(I)=I
71400= 32   ITAG(I)=0
71500=      KK=1
71600= 902  FORMAT(27H  SPECIFY SUBSPACE TAGS     /
71700=     1 11H  KK,KK,.../2H *)
71800= 903  FORMAT(20(I2,1X))
71900= 904  FORMAT(1X,20(I2,1X))
72000=C***            READS TAG-LIST 'TIL 99 ENTRY
72100= 33   CONTINUE
72200=      KN=KK+19
72300=      IF(KN.GT.50) KN=50
72400=      WRITE(IOUTF,902)
72500=      IF(ILOG.EQ.1.OR.ILOG.EQ.2)WRITE(ILOGF,902)
72600=      READ(INPF,903)(ITAG(K),K=KK,KN)
72700=      IF(ILOG.EQ.1.OR.ILOG.EQ.2)WRITE(ILOGF,904)(ITAG(K),K=KK,KN)
72800=C***            QUIT AT TAG=99, OR ENTRY NO > (JD.OR.50)
72900=C***            ZERO NTAG(I) IF 'SLOT' SPECIFIED
73000=      DO 35 I=1,20
73100=      IF(KK.GT.JD) GOTO 50
73200=      IF(KK.GT.50) GOTO 50
73300=      IF(ITAG(I).EQ.0) GOTO 33
73400=      IF(ITAG(I).EQ.99) GOTO 42
73500=      KX=ITAG(I)
73600=      NTAG(KX)=0
73700=      KK=KK+1
73800= 35   CONTINUE
73900=      GOTO 33
74000=C**         ERROR RETRY
74100= 40   CONTINUE
74200=      CALL ERR(IOUTF,6,IEOJ)
74300=      IF(IEOJ.GT.0) GOTO 50
74400=      GOTO 31
74500=C***        SLOT NON-ZERO NTAGS
74600= 42   CONTINUE
74700=      II=0
74800=      DO 44 J=KK,JD
74900= 43   II=II+1
75000=      IF(NTAG(II).EQ.0) GOTO 43
75100=      ITAG(J)=NTAG(II)
75200= 44   CONTINUE
75300=C**           NORMAL EXIT
75400= 50   CONTINUE
75500= 908  FORMAT(15H  SUBSPACE TAGS/,(((20I3))) )
75600=      IF(ILOG.NE.0)WRITE(ILOGF,908)(ITAG(J),J=1,JD)
75700=      RETURN
75800=      END
```

```
76900=
77000=
77100=      SUBROUTINE SUBSET(CLAS,JDX,ICX,IEOJ)
77200=      COMMON/NAMES/INPF,IOUTF,IFEAT,IOLDC,INEWC,ILOGF,IHISC
77300=      COMMON/TRACE/ILOG,ITRACE
77400=      COMMON/CONX/LCID(51,3),SCALE(2),ICONF,ANS(3),IMAG
77500=      COMMON/CNUM/NEXC,NTC
77550=      COMMON/KFILE/MAXKV,NCIX,NENT,MSCAL,MZAPD
77600=      DIMENSION CLAS(JDX,1),LISD(50)
77700=C***     ACCEPTS USER INPUTS AND SUBSETS OR 'ZAPS' COMPONENTS
77800=C***     ICLX=98 --> INITIALIZE CLAS FILE
77900=C***          88 --> QUIT SUBSET, CONTINUE JOB
78000=C***          00 --> QUIT SUBSET,  END JOB
78100=C***         +99 --> 'ZAP' ALL BUT SPECIFIED DIMS FOR ALL CLASSES
78200=C***         -99 --> 'ZAP' SPECIFIED DIMENSIONS FOR ALL CLASSES
78300=C***     +-01 TO 50 --> 'ZAP' DIMENSIONS AS ABOVE FOR SPECIFIED CLA
78400=C***     LISD=<0 --> ERROR
78500=C***          00 --> CONTINUE LIST FOR THIS CLASS
78600=C***          >0 --> INDICATES DIMENSION TO NULLIFY
78700=C***          99 --> END LIST FOR THIS CLASS
78800= 1    CONTINUE
78900=      NN=3
79000=      JD=JDX-3
79100=      DO 2 L=1,50
79200= 2    LISD(L)=0
79300=      IEOJ=-1
79400=      WRITE(IOUTF,900)
79500= 900  FORMAT(8H  SUBSET/28H  ENTER CLASS AND DIMENSIONS)
79600=C***              INPUT CLASS/CONTROL VALUE
79700= 3    CONTINUE
79800=      WRITE(IOUTF,904)
79900= 904  FORMAT(5H  +CL/2H *)
80000=      READ(INPF,905)ICLX
80100= 905  FORMAT(I3)
80200=      IF(ILOG.EQ.1)WRITE(ILOGF,906)ICLX
80300= 906  FORMAT(17H  SUBSET  CLASS= ,I3)
80400=      IF(ICLX.EQ.98) GOTO 8
80500=      IF(ICLX.EQ.88) GOTO 800
80600=      IF(ICLX.EQ.0) GOTO 888
80700=      IF(IABS(ICLX).GT.NTC.AND.IABS(ICLX).NE.99) GOTO 25
80800=      KK=1
```

```
80900=C***              INPUT DIMENSIONS LIST
81000= 5    CONTINUE
81100=      WRITE(IOUTF,902)
81200= 902  FORMAT(12H  DD,DD,...  /2H *)
81300=      KN=19+KK
81400=      IF(KN.GT.50) KN=50
81500=      READ(INPF,901)(LISD(K),K=KK,KN)
81600=      IF(ILOG.EQ.1)WRITE(ILOGF,902)
81700=      IF(ILOG.EQ.1)WRITE(ILOGF,907)(LISD(K),K=KK,KN)
81800= 907  FORMAT(2H+ ,20(I2,1X))
81900= 901  FORMAT(20(I2,1X))
82000=      GOTO 20
82100=C***              REINITIALIZE CLAS FILE BEFORE SUBSET
82200= 8    CONTINUE
82300=      BACKSPACE IOLDC
82400=      CALL LOADC(CLAS,JDX,ICX,IEOJ)
82500=      CALL INDEX(CLAS,JDX,ICX,IEOJ)
82550=      MZAPD=0
82600=      IF(IEOJ.GT.0)GOTO 800
82700=      GOTO 1
82800=C***              DECODE SUBSET CONTROLS
82900= 20   CONTINUE
83000=      DO 24 K=1,20
83100=      IF(LISD(KK).GE.99) GOTO 30
83200=      IF(LISD(KK).EQ.0) GOTO 5
83300=      KK=KK+1
83400= 24   CONTINUE
83500=      IF(KN.LT.50) GOTO 5.
83600=      GOTO 30
83700=C**           ERROR RETRY
83800= 25   CONTINUE
83900=      CALL ERR(IOUTF,KN,IEOJ)
84000=      IF(IEOJ.LT.0) GOTO 5
84100=      GOTO 800
84200=C**           'ZAP' SPECIFIED DIMS IF ICLX < 0
84300=C**           'ZAP' UNSPECIFIED DIMS IF ICLX > 0
84400= 30   CONTINUE
84500=      CALL IASORT(LISD,KK-1)
84600= 903  FORMAT(2X,9H  SUBSET ,I3,/2X,8H  ZAPS= ,(((20(1XI2) ))) )
84700=      IF(ICLX.LT.0) GOTO 40
```

```
84800=C***              COMPLEMENT LISD IF ICLX = +99
84900= 33   CONTINUE
85000=      KN=KK-1
85100=      LISD(KK)=0
85200=      JN=JD
85300= 35   IF(JN.EQ.0) GOTO 40
85400=      IF(LISD(KN).EQ.JN) GOTO 37
85500=C***              SET LISD WHEN LISD NOT SET
85600=      LISD(JN)=JN
85700=      JN=JN-1
85800=      GOTO 35
85900=C***              CLEAR LISD WHEN LISD HAS AN ENTRY
86000= 37   CONTINUE
86100=      LISD(JN)=0
86200=      JN=JN-1
86300=      IF(KN.GT.1)KN=KN-1
86400=      GOTO 35
86500=C***              'ZAP' CLASSES AND DIMENSIONS
86600= 40   CONTINUE
86700=      IF(ILOG.EQ.3)WRITE(ILOGF,903)ICLX,(LISD(J),J=1,JD)
86750=      MZAPD=1
86800=      NCL=1
86900=      IF(IABS(ICLX).NE.99) NCL=ICLX
87000= 45   CONTINUE
87100=      NCSDL=LCID(NCL,2)
87200=      NCSDR=LCID(NCL,3)
87300=      DO 50 J=1,JD
87400=      IF(LISD(J).EQ.0) GOTO 50
87500=      JZAP=LISD(J)
87600=      CLAS(JZAP,NCSDL)=1E20
87700=      IF(NCSDR.GT.0)CLAS(JZAP,NCSDR)=1E20
87800= 50   CONTINUE
87900= 55   CONTINUE
88000=      IF(IABS(ICLX).NE.99) GOTO 800
88100=      IF(NCL.GE.NTC) GOTO 800
88200=      NCL=NCL+1
88300=      GOTO 45
88400=C***              QUIT JOB EXIT
88500= 888  CONTINUE
88600=      IEOJ=2
88700=C***              NORMAL EXIT
88800= 800  CONTINUE
88900=C**           TRACE DUMP CLAS FILE
89000=      IF(ILOG.EQ.1.OR.ILOG.EQ.2)
89100=     1 CALL PRCLAS(CLAS,CLAS,CLAS,JDX,ICX)
89200=      RETURN
89300=      END
```

```
19200=
19300=
19400=      SUBROUTINE EVAL(FEAT,CLAS,CMX,JDX,NAX,IC,IEOJ)
19500=      COMMON/NAMES/INPF,IOUTF,IFEAT,IOLDC,INEWC,ILOGF,IHISC
19600=      COMMON/TRACE/ILOG,ITRACE
19700=      COMMON/CONX/LCID(51,3),SCALE(2),ICONF,ANS(3),IMAG
19800=      COMMON/CREC/LASV,IEOFR,NBPR,LB,MAXV
19900=      COMMON/CNUM/NEXC,NTC
20000=      COMMON/SRCH/ISX,IKNO,NIC,JD,ITAG(50),DIS(50),RATE(2,50)
20100=      COMMON/TRY/IFIG,ISUB,NT1,ND1,ND2,LABLC
20200=      DIMENSION FEAT(JDX,1),CLAS(JDX,1),CMX(IC,1)
20300=      DIMENSION FDIS(50),NTAG(50)
20400=      DIMENSION IRATE(2,50)
20500=      EQUIVALENCE (FNIC,NIC),(ICME,CME),(NVEC,FVEC),(IKNO,FIKNO)
20600=      EQUIVALENCE(IRATE(1,1),RATE(1,1))
20700=      EQUIVALENCE (FDIS(1),NTAG(1))
20800=C**
20900=      NCMIN=LCID(51,1)
21000=      NCMAX=LCID(51,2)
21100=      JD=JDX-3
21200=      INIT=9
21300=      IEOJ=-1
21400=      DO 25 I=1,IC
21500=      DO 20 N=1,NAX
21600=      CMX(I,N)=6.0
21700= 20   CONTINUE
21800= 25   CONTINUE
21900=      DO 30 K=1,3
22000= 30   ANS(K)=0.0
22100=      NUMV=0
22200=      NUMT=0
22300=      NEXC=1
22400=C**           INPUT FEATURE VECTORS, COUNT PER CLASS
22500= 50   CONTINUE
22600=      L=0
22700=      CALL RFEAT(FEAT,JDX,INIT,IEOJ)
22800=      IF(IEOJ.GT.0) GOTO 800
22900=      FIKNO=FEAT(JDX-1,1)
23000=      IKNO=IABS(IKNO)
23100=C**           PROCESS FVEC BLOCK
23200=C**           FMIN,FMAX ARE GLOBAL BOUNDS
23300=C**           SCALING IS OK-MAHDIS,NG EUCDIS
23400= 100  CONTINUE
23500=      L=L+1
23600=      IF(SCALE(1).GT.1.)
23700=     1 CALL XSCAL(CLAS(1,NCMIN),CLAS(1,NCMAX),FEAT(1,L),JD,SCALE,1)
23800=      FVEC=FEAT(JDX-2,L)
23900=      NUMV=NUMV+1
24000=C**           COMPUTE MAHALANOBIS' DISTANCE 2 WAYS
24100=C**           TO EACH PROTO; PICK CLOSEST CLASS
24200=      DMIN=.99E99
24300=      ISX=0
24400=      DO 200 N=1,NTC
24500=      NCAV=LCID(N,1)
24600=      FNIC=CLAS(JD+3,NCAV)
24700=      NCSDL=LCID(N,2)
24800=      NCSDR=LCID(N,3)
24900=      KSYM=1
25000=      IF(NCSDR.GT.0) KSYM=2
25100=C**           COMPUTE MAHALANOBIS' DISTANCE 2 WAYS
25200=C**           TO EACH PROTO; PICK CLOSEST CLASS
25300=      DMAG=0.
25400=      DMAX=0.
25500=      IF(N.EQ.NTC) ISX=2
25600=      IF(SCALE(1).GT.1.) GOTO 160
25700=C**           REAL DISTANCE PROCESS
25800=        DO 150 J=1,JD
25900=        DFEAT=FEAT(J,L)-CLAS(J,NCAV)
26000=        IF(DFEAT.LT.0.0.OR.KSYM.EQ.1)
26100=     1    DIS(J)=DFEAT/CLAS(J,NCSDL)
26200=        IF(DFEAT.GE.0.0.AND.KSYM.EQ.2)
26300=     1    DIS(J)=DFEAT/CLAS(J,NCSDR)
26400=        DIS(J)=ABS(DIS(J))
26500=        IF(DIS(J).GT.DMAX) DMAX=DIS(J)
26600=        DMAG=DMAG+DFEAT**2
26700= 150    CONTINUE
```

```
26800=      GOTO 190
26900=C**           SCALED DISTANCE PROCESS
27000= 160  CONTINUE
27100=        DO 180 J=1,JD
27200=        KFEAT=(FEAT(J,L)-CLAS(J,NCAV))
27300=        KFEAT=   KFEAT*2**8
27400=        KCSDL=CLAS(J,NCSDL)*2.
27500=        KCSDR=CLAS(J,NCSDR)*2.
27600=        IF(KFEAT.LT.0.OR.KSYM.EQ.1)
27700=     1     DIS(J)=KFEAT/KCSDL
27800=        IF(KFEAT.GE.0.AND.KSYM.EQ.2)
27900=     1     DIS(J)=KFEAT/KCSDR
28000=        DIS(J)=ABS(DIS(J))
28100=        IF(DIS(J).GT.DMAX) DMAX=DIS(J)
28200= 180    CONTINUE
28300=C**           RESUME DECISION RULE
28400= 190  CONTINUE
28500=      IF(IFIG.EQ.1) CALL LOOK
28600=      ISX=1
28700=      DMAG=SQRT(DMAG)
28800=C**           CHOSE CLASS
28900=      DVAL=DMAX
29000=      IF(IMAG.EQ.1)DVAL=DMAG
29100=      PDIS(N)=DVAL
29200=      IF(DVAL.GT.DMIN) GOTO 200
29300=      ISAY=NIC
29400=      DMIN=DVAL
29500= 200  CONTINUE
```

```
29600=C**           STORE FVEC RECOGNITION DECISION
29700=      CMX(IKND,ISAY)=CMX(IKND,ISAY)+1.
29800=      IF(IKND.EQ.ISAY)ANS(1)=ANS(1)+1.
29900=      IF(IKND.NE.ISAY)ANS(2)=ANS(2)+1.
30000=      IF(ILOG.EQ.1.AND.L.LT.5)
30100=     1  WRITE(ILOGF,900)NVEC,IKND,ISAY,(PDIS(N),N=1,NTC)
30200= 900  FORMAT(2X,7HDEC VEC,I5,6H CLASS,I4,4H IS ,I4/(((10(1XE9.3) ))) )
30300=      IF(L.LT.5.AND.(ILOG.EQ.1.OR.ILOG.EQ.2))
30400=     1 WRITE(ILOGF,901)IKND,NUMV,(FEAT(J,L),J=1,JD)
30500= 901  FORMAT(6H  CLAS= ,I3,14H  FEAT VEC NO= ,I4/(((10(1XE9.3)))) )
30600=      IF(L.LT.LASV) GOTO 100
30700=C**           GET NEXT RECORD BLOCK
30800=      IF(IEOFR.NE.1) GOTO 50
30900=C**           FINISHED A PRIORI CLASS
31000= 300  CONTINUE
31100=      CMX(IKND,NAX)=NUMV
31200=      NUMT=NUMT+NUMV
31300=      NUMV=0
31400= 957  FORMAT(5H  EV ,11I6)
31500=      NEXC=NEXC+1
31600=      IF(NEXC.LE.IC) GOTO 50
31700=C**           FINISHED ALL FEAT FILE
31800=C**           UNITIZE CONFUSION MATRIX
31900= 400  CONTINUE
32000=      IEOJ=0
32100=      DO 600 N=1,NTC
32200=      DO 500 I=1,IC
32300=      ICME=(CMX(I,N)/CMX(I,NAX)) *100.
32400= 500  CMX(I,N)=CME
32500= 600  CONTINUE
32600=      DO 700 J=1,3
32700= 700  ANS(J)=ANS(J)/FLOAT(NUMT)
32800=      DO 750 J=1,JD
32900=      IRATE(1,J)=RATE(1,J)/FLOAT(NUMT)*100.
33000=      IRATE(2,J)=RATE(2,J)/FLOAT(NUMT)*100.
33100= 750  CONTINUE
33200=C**           ERROR EXIT
33300= 800  CONTINUE
33400=      RETURN
33500=      END
```

```
33900=      SUBROUTINE DOCU(CMX,CLAS,JDX,NAX,IC)
33950=      COMMON/CNUM/NEXC,NTC
34000=      COMMON/NAMES/INPF,IOUTF,IFEAT,IOLDC,INEWC,ILOGF,IHISC
34100=      COMMON/TRACE/ILOG,ITRACE
34200=      COMMON/CONX/LCID(51,3),SCALE(2),ICONF,ANS(3),IMAG
34300=      COMMON/SRCH/ISX,IKNO,NIC,JD,ITAG(50),DIS(50),RATE(2,50)
34400=      COMMON/TRY/IFIG,ISUB,NT1,ND1,ND2,LABLC
34500=      INTEGER CLAS(JDX,1),CMX(IC,1)
34600=      DIMENSION IRATE(2,50)
34700=      EQUIVALENCE(FNUM,NUM)
34800=      EQUIVALENCE(IRATE(1,1),RATE(1,1))
34900=C**           PRINT CONCLUSION
35100=      WRITE(IOUTF,902) (ANS(I),I=1,3)
35200=      IF(ILOG.NE.0)WRITE(ILOGF,902)(ANS(I),I=1,3)
35300= 902  FORMAT(20H  SUMMARY CONCLUSION/3F7.4)
35400= 901  FORMAT(2X,I4,13,(((37(I3)))) )
35500= 900  FORMAT(16H  CONFUSION MATRIX)
35600=      IF(ICONF.LT.1) GOTO 20
35700=C**           PRINT CONFUSION MATRIX
35800=      IF(ILOG.NE.0)WRITE(ILOGF,900)
35900=      DO 10 I=1,IC
36000=      NUM=CMX(I,NAX)
36100=      NUM=FNUM
36200=      IF(ILOG.NE.0)WRITE(ILOGF,901)NUM,I,(CMX(I,N),N=1,NTC)
36300= 10   CONTINUE
36400=C**           PRINT SUBSPACE ERROR RATES IF OPTED
36500= 20   CONTINUE
36600=      IF(IFIG.NE.1) GOTO 30
36700=      IF(ILOG.NE.0)WRITE(ILOGF,904)(ITAG(J),J=1,JD)
36800=      WRITE(IOUTF,904)(ITAG(J),J=1,JD)
36900=      WRITE(IOUTF,903) (J,IRATE(1,J),IRATE(2,J),J=1,JD)
37000=      IF(ILOG.NE.0)WRITE(ILOGF,903) (J,IRATE(1,J),IRATE(2,J),J=1,JD)
37100= 904  FORMAT(15H  SUBSPACE TAGS/(((20(1X,I2))) ) )
37200= 903  FORMAT(22H  SUBSPACE ERROR RATES/
37300=     1 (((12(1X,I2,1H=,I2,1H/,I2)))) )
37400=C**           NORMAL EXIT
37500= 30   CONTINUE
37600=      RETURN
37700=      END
37800=
```

```
98300=      SUBROUTINE LOOK
98400=      COMMON/TRACE/ILOG,ITRACE
98500=      COMMON/SRCH/ISX,IKNO,NIC,JD,ITAG(50),DIS(50),RATE(2,50)
98600=      DIMENSION CLOSE(50),IPICK(50)
98700=      DIMENSION WORK(50)
98800=C**
98900=C**          CALCULATES ERROR CLASSIFICATION RATES FOR THE JD
99000=C**          SUBSETS OF THE JD DIMENSIONS OF FEATURE SPACE.
99100=C**          THESE SUBSETS ARE THE JD SETS OF INCREASING NUMBERS
99200=C**          OF COMPONENTS ORDERED BY DECREASING 'FIGURES OF MERIT'.
99300=C**
99400=      IF(ISX.NE.0) GOTO 40
99500=C**          RESET CLOSE AND IPICK DECISION ARRAYS
99600=      DO 20 I=1,JD
99700=      CLOSE(I)=1E9
99800=      IPICK(I)=0
99900= 20   CONTINUE
100000=C**          REORDER DISTANCE VECTOR PER MERIT TAGS
100100= 40  CONTINUE
100200=     DO 45 J=1,JD
100300=     K=ITAG(J)
100400=     WORK(J)=DIS(K)
100500= 45  CONTINUE
100600=C**          COMPUTE AN R-INFINITY MAG FOR EACH SUBSET VECTOR
100700=     DO 100 J=1,JD
100800=     TMAG=-1E20
100900=     DO 90 K=1,J
101000=     IF(WORK(K).GE.TMAG) TMAG=WORK(K)
101100= 90  CONTINUE
101200=     IF(TMAG.GT.CLOSE(J)) GOTO 95
101300=C**          SAVE TENTATIVE DECISION FOR EACH SUBSPACE
101400=     IPICK(J)=NIC
101500=     CLOSE(J)=TMAG
101600= 95  CONTINUE
101700=     IF(ISX.NE.2) GOTO 100
101800=C**          HAVING CHECKED ALL CLASSES,RECORD DECISION
101900=     IF(IPICK(J).EQ.IKNO)RATE(1,J)=RATE(1,J)+1.0
102000=     IF(IPICK(J).NE.IKNO)RATE(2,J)=RATE(2,J)+1.0
102100= 100 CONTINUE
102200=C**          NORMAL EXIT
102300= 800 CONTINUE
102400=     RETURN
102500=     END
```

APPENDIX H

FORMAT Module

```
100=
200=      PROGRAM FORMAT(FEAT,OLDC,HIST,BITE,LOGF=64,
300=     1   INPUT=64,OUTPUT=64)
400=C***          3@ OCT VERSION 1400
500=      COMMON W0(8200)
600=      COMMON/CNUM/NEXC,NTC
700=      COMMON/PARAM/MEMC,MEM,NHX,JDX,ICX,NC1,NL1,NB1,IC
800=      COMMON/NAMES/INPF,IOUTF,IFEAT,IOLDC,INEWC,ILOGF,IHISC
900=      COMMON/FORM/IFIL,IFN,IGRIN,NBX,NBY,NDSENT
1000=     NX1=1
1100=     MEM=8200
1200=     DO 5 M=1,MEM
1300=     W0(M)=0.
1400= 5   CONTINUE
1500=     CALL DEFF(IEOJ)
1600=     IF(IEOJ.GT.0) GOTO 800
1700=     CALL LOADC(W0(NC1),JDX,ICX,IEOJ)
1800=     CALL INDEX(W0(NC1),JDX,ICX,IEOJ)
1900=     IF(IEOJ.GT.0) GOTO 800
2000=C*****    PROCESS EACH CLASS
2100= 10  CONTINUE
2200=     CALL NEXREC(IEOJ)
2300=     IF(IEOJ.GT.0)GOTO 700
2400=     IF(IFIL.EQ.1)
2500=    1 CALL XFEAT(W0(NL1),W0(NB1),W0(NX1),W0(NC1),JDX,NBX,IEOJ)
2600=     IF(IFIL.EQ.2)
2700=    2 CALL XCLAS(W0(NC1),W0(NB1),W0(NX1),JDX,NBX,IEOJ)
2800=     IF(IFIL.GE.3)
2900=    3 CALL XHIST(W0(NL1),W0(NB1),W0(NX1),NHX,NBX,IEOJ)
3000=     IF(IEOJ.EQ.-9) GOTO 700
3100=     IF(IEOJ.GT.0) GOTO 800
3200=     IF(NEXC.LE.NTC) GOTO 10
3300=C*****    NORMAL END OF JOB
3400= 700 CONTINUE
3500=     IF(IFN.EQ.3)WRITE(IOUTF,950)NDSENT
3600= 950 FORMAT(14H  XMIT OUTPUT ,I3,11H DIMENSIONS)
3700=     STOP
3800=C*******    PRINT ERROR MESSAGES
3900= 800 CONTINUE
4000=     IF(IEOJ.EQ.1)WRITE(IOUTF,900)
4100= 900 FORMAT(2X,18H  MISSING  RECORD  )
4200=     IF(IEOJ.EQ.2)WRITE(IOUTF,901)
4300= 901 FORMAT(2X,13H  WRONG FILE   )
4400=     IF(IEOJ.EQ.3)WRITE(IOUTF,902)
4500= 902 FORMAT(2X,20H  TOO LITTLE MEMORY   )
4600=     IF(IEOJ.EQ.4) WRITE(IOUTF,903)
4700= 903 FORMAT(27H  CLAS AND FEAT NOT MATCHED)
4800=     IF(IEOJ.EQ.5) WRITE(IOUTF,904)
4900= 904 FORMAT(18H  ILLEGAL FUNCTION)
5000=     IF(IEOJ.EQ.10) WRITE(IOUTF,905)
5100= 905 FORMAT(39H  CLAS FILE HAS INCONSISTENT DIMENSIONS)
5200=     IF(IEOJ.EQ.6)WRITE(IOUTF,906)
5300= 906 FORMAT(15H  ILLEGAL INPUT)
5400=     STOP
5500=     END
5600=
```

```
13400=
13500=      SUBROUTINE DEFF(IEOJ)
13600=      COMMON/CNUM/NEXC,NTC
13700=      COMMON/NAMES/INPF,IOUTF,IFEAT,IOLDC,INEWC,ILOGF,IHISC
13800=      COMMON/TRACE/ILOG,ITRACE
13900=      COMMON/CONX/LCID(51,3),SCALE(2),ICONF,ANS(3),IMAG
14000=      COMMON/PARAM/MEMC,MEM,NHX,JDX,ICX,NC1,NL1,NB1,IC
14100=      COMMON/FORM/IFIL,IFN,IGRIN,NBX,NBY,NDSENT
14200=      COMMON/KFILE/MAXKV,NCIX,NENT,MSCAL,MZAPD
14300=      COMMON/CREC/LASV,IEOFR,NBPR,LB,MAXV
14400=      COMMON/HREC/ISYM,KTR,LABLH,NI,NBUC,JD
14500=      COMMON/TRANS/IBUF(16),IVAL,NVAL,LTYP,LADR(4),LINE(43),IBYTE
14600=      COMMON/PL3D/BSC,ZSC,EL,ICPS,IGRID
14700=      DIMENSION KOPNS(3)
14800=      DATA IHF,IHH,IHC,IHD,IHP,IHX,IHS,IHG
14900=     1    /1HF,1HH,1HC,1HD,1HP,1HX,1HS,1HG/
15000=      DATA IHL,IHT,IHY/1HL,1HT,1HY/
15100=C
15200=      INPF=5LINPUT
15300=      IOUTF=6LOUTPUT
15400=      IHISC=4LHIST
15500=      IFEAT=4LFEAT
15600=      IOLDC=4LOLDC
15700=      ILOGF=4LLOGF
15800=      ITRACE=6LOUTPUT
15900=      IBYTE=4LBYTE
16000=      SCALE(1)=1.
16100=      IFIL=0
16200=      IFN=0
16300=      ILOG=0
16400=      IEOJ=-1
16500=      NN=3
```

```
16600=C**          GET FILE  AND FUNCTION
16700=C***       F-->FEAT,     X-->TRANSMIT      T--> TRACE EXECUTION
16800=C***       H-->HIST,     P-->PICTURE(3D)   Y--> TRACE AND LOG
16900=C***       C-->CLAS,     S-->STRIP CHART   L--> LOG OUTPUT
17000=C*         D-->DIST,
17100= 5    CONTINUE
17200=      WRITE(IOUTF,900)
17300= 900  FORMAT(8H  FORMAT/25H  ENTER FILE AND FUNCTION/2H *)
17400=      IF(ILOG.NE.0)WRITE(ILOGF,900)
17500=      READ(INPF,901) KOPNS
17600=      IF(ILOG.NE.0)WRITE(ILOGF,902)KOPNS
17700= 902  FORMAT(2X,3A1)
17800= 901  FORMAT(3A1)
17900=      DO 6 I=1,3
18000=      IF(KOPNS(I).EQ.IHF)IFIL=1
18100=      IF(KOPNS(I).EQ.IHC)IFIL=2
18200=      IF(KOPNS(I).EQ.IHH)IFIL=3
18300=      IF(KOPNS(I).EQ.IHD)IFIL=4
18400=      IF(KOPNS(I).EQ.IHS)IFN=1
18500=      IF(KOPNS(I).EQ.IHP)IFN=2
18600=      IF(KOPNS(I).EQ.IHX)IFN=3
18700=      IF(KOPNS(I).EQ.IHT)ILOG=1
18800=      IF(KOPNS(I).EQ.IHY)ILOG=2
18900=      IF(KOPNS(I).EQ.IHL)ILOG=3
19000= 6    CONTINUE
19100=      ITEM=IFIL*IFN
19200=      IF(ITEM.EQ.0) GOTO 30
19300=      IF(IFN.NE.2) GOTO 7
19400=C***            GET SPECIAL PLOT PARAMETERS
19500=      WRITE(IOUTF,903)
19600=      IF(ILOG.NE.0)WRITE(ILOGF,903)
19700= 903  FORMAT(23H  ENTER PLOT PARAMETERS/
19800=     1 8H  ICPS,G/2H *)
19900=      READ(INPF,904)ICPS,IGRIN
20000= 904  FORMAT(I4,1X,A1)
20100=      IF(ILOG.NE.0)WRITE(ILOGF,904)ICPS,IGRIN
20200= 7    CONTINUE
20300=C***            GET SPECIAL XMIT PARAMETERS
20400=      IF(IFN.NE.3) GOTO 9
20500=      IF(MSCAL.NE.1)SCALE(1)=256.
20600=      WRITE(IOUTF,906)
20700=      IF(ILOG.NE.0)WRITE(ILOGF,906)
20800= 906  FORMAT(23H  ENTER XMIT PARAMETERS/
20900=     1 6H  AAAA/2H *)
21000= 907  FORMAT(4A1)
21100=      READ(INPF,907) LADR
21200= 9    CONTINUE
21300=C**      OPEN DATA FILES
21400=      CALL OPENX(0,IFIL+1,LABLC,LABLF,IEOJ)
21500=      IF(IFIL.GT.2) CALL OPENH(LABLH,IEOJ)
21600=      JD=JDX-3
21700=      IF(IEOJ.GT.0) GOTO 20
22000=C***            MEMORY CONTROL PARAMETERS
22100=C***            MEMX-->3DPLOT(HID),MEMB-->PLOT BUFFER
22200=C***            MEMC, MEMF, MEMH  --->CLAS,FEAT,HIST BUFFERS
22300=C***            BUFFER(NBX,NBY) MEMORY REQUIREMENTS
22400=C***  HBUF(NBUC,JD),CBUF(JD,ICX),FBUF(JD,LB)
22500=C***  STRIP-->NBY.LE.9  ,PLOT-->NBY.LE.50
22600=      IF(IFIL.EQ.1)NTC=IC
22700=      MEMC=JDX*ICX
22800=      IF(IFIL.EQ.2)NBY=5*ICX
22900=      IF(IFIL.EQ.1)NBY=LB
23000=      IF(IFIL.GE.3)NBY=JD
23100=      IF(IFN.GE.3)NBY=MIN0(NBY,0)
23200=      IF(IFN.EQ.1)NBY=MIN0(NBY,9)
23300=      IF(IFN.EQ.2)NBY=MIN0(NBY,50)
23400=      IF(IFIL.LE.2)NBX=JD
23500=      IF(IFIL.GE.3)NBX=NHX-8
23600=      IF(IFN.EQ.3)NBX=1
23700=      MEMB=NBX*NBY
23800=      MEMX=2*(NBX+NBY)
23900=      NC1=1+MEMX
24000=      NB1=NC1+MEMC
24100=      NL1=NB1+MEMB
24200=      IF(IFIL.EQ.1) MEML=JDX*LB
24300=      IF(IFIL.GE.3)MEML=NHX*JD
24400=C**         CHECK THAT MEM IS BIG ENOUGH
24500=      MEMK=MEM-MEML-MEMC-MEMB-MEMX
24600=      IF(MEMK.LT.0) GOTO 25
24700=C**           NORMAL EXIT
24800= 20   CONTINUE
24810=      IF(ILOG.NE.0)WRITE(ITRACE,957)IFIL,IFN,ILOG,NEXC,NTC,NC1,NB1,NL1
24820= 957  FORMAT(6H DEFF ,8I4)
24900=      IEOJ=0
25000=C***           INPUT ERROR EXIT
25100= 22   CONTINUE
25200=      RETURN
25300=C**           TOO LITTLE MEMORY
25400= 25   CONTINUE
25500=      IEOJ=3
25600=      GOTO 20
25700=C**           ERROR RETRY
25800= 30   CONTINUE
25900=      CALL ERR(IOUTF,NN,IEOJ)
26000=      IF(IEOJ.LT.0) GOTO 5
26100=      GOTO 22
26200=      END
26300=
```

```
26400=
26500=      SUBROUTINE XCLAS(CLAS,BUF,HID,JDX,NBX,IEOJ)
26600=      COMMON/NAMES/INPF,IOUTF,IFEAT,IOLDC,INEWC,ILOGF,IHISC
26700=      COMMON/CNUM/NEXC,NTC
26800=      COMMON/TRACE/ILOG,ITRACE
26900=      COMMON/CONX/LCID(51,3),SCALE(2),ICONF,ANS(3),IMAG
27000=      COMMON/FORM/IFIL,IFN,IGRIN,DUM,NBY,NDSENT
27100=      COMMON/TRANS/IBUF(16),IVAL,NVAL,LTYP,LADR(4),LINE(43),IBYTE
27200=      DIMENSION CLAS(JDX,1),BUF(NBX,1),HID(1)
27300=C***              INITIALIZE ON FIRST ENTRY
27400=      IF(KIN.EQ.12345) GOTO 10
27500=      KIN=12345
27600=      KBY=NBY-5
27700=      NVAL=0
27800=      NPV=1
27900=      JD=JDX-3
28000=      IEOJ=-1
28100=      LTYP=1
28200=C***              PROCESS NEXT VECTOR (NPV)
28300= 10   CONTINUE
28400=      NDSENT=KDSENT
28500=      NCAV=LCID(NEXC,1)
28600=      NCSDL=LCID(NEXC,2)
28700=      NCSDR=LCID(NEXC,3)
28800=      NCSD=NCSDR
28900=      IF(NCSDR.LT.0)NCSD=NCSDL
29000=      IF(IFN.NE.3) GOTO 100
29100=C***              TRANSMIT PROTOTYPE VALUES
29200=C***          SKIP 'ZAPPED' COMPONENTS
29300=      IF(NEXC.GT.NTC)GOTO 20
29400=      KDSENT=0
29500=      IVAL=NEXC
29600=      DO 12 J=1,JD
29700=      IF(CLAS(J,NCSDL).GT.1E6) GOTO 12
29800=      KDSENT=KDSENT+1
29900=      IVAL=CLAS(J,NCAV)
30000=      CALL XMIT
30100=      IVAL=CLAS(J,NCSDL)
30200=      CALL XMIT
30300=      IVAL=CLAS(J,NCSD)
30400=      CALL XMIT
30500= 12   CONTINUE
30600=      GOTO 800
30700=C***              FLUSH BYTE FILE AT END
30800= 20   CONTINUE
30900=      NVAL=16
31000=      CALL XMIT
31100=      LTYP=3
31200=      NVAL=16
31300=      CALL XMIT
31400=      GOTO 800
31500=C***              LOAD CLAS INTO STRIP/PLOT BUF
31600= 100  CONTINUE
31700=      IF(IFN.EQ.2.AND.NPV.EQ.1) NPV=2
31800=      IF(NEXC.GT.NTC) GOTO 190
31900=      DO 110 J=1,JD
32000=     BUF(J,NPV+1)=CLAS(J,NCAV)-CLAS(J,NCSDL)
32100=      BUF(J,NPV)=CLAS(J,NCAV)
32200=      BUF(J,NPV+2)=CLAS(J,NCAV)+CLAS(J,NCSD)
32300= 110  CONTINUE
32400=      NPV=NPV+5
32500=      IF(NPV.LE.KBY) GOTO 800
32600=C***              FLUSH BUFFER WHEN FULL
32700= 190  CONTINUE
32800=      IF(IFN.EQ.2)CALL PICT(BUF,HID,NBX,NPV)
32900=      IF(IFN.EQ.3)CALL STRIP(BUF,NBX,NPV)
33000=      NPV=1
33100=C***              NORMAL EXIT
33200= 800  CONTINUE
33300=      IF(NDSENT.EQ.0) GOTO 810
33400=      IF(NDSENT.NE.KDSENT) IEOJ=10
33500= 810  CONTINUE
33600=      RETURN
33700=      END
```

```
5700=
5800=      SUBROUTINE XHIST(HIST,BUF,HID,NHX,NBX,IEOJ)
5900=      COMMON/CNUM/NEXC,NTC
6000=      COMMON/FORM/IFIL,IFN,IGRIN,DUM,NBY,NDSENT
6100=      COMMON/HREC/ISYM,KTR,LABLH,NI,NBUC,JD
6200=      DIMENSION HIST(NHX,1),BUF(NBX,1),HID(1)
6300=C***                PROCESS HISTOGRAMS FROM CREATE AND DEFINE
6400=      IEOJ=0
6500= 5    CONTINUE
6600=      CALL RHIST(HIST,NHX,IEOJ)
6700=      IF(IEOJ.EQ.-9) GOTO 85
6800=      IF(IEOJ.GT.0) GOTO 85
6900=      IF(IFN.EQ.1) GOTO 80
7000=      IF(ISYM.EQ.1) NX=NBUC
7100=      IF(ISYM.EQ.2) NX=NI
7200=      NY=JD
7300=      J1=1
7400=      JN=JD
7500=      IF(JN.GT.NBY)JN=NBY
7600= 10   CONTINUE
7700=      ICON=ISYM
7800=      DO 15 I=1,ISYM
7900=      CALL FILBUF(HIST,BUF,NHX,J1,JN,NX,NY,ICON)
7950=      IF(IFN.EQ.IFN) STOP "TEST"
8000=      IF(IFN.EQ.2) CALL PICT(BUF,HID,NX,NY)
8100=      IF(IFN.EQ.3) CALL STRIP(BUF,NX,NY)
8200=      ICON=ISYM+1
8300= 15   CONTINUE
8400=      IF(JN.GE.JD) GOTO 20
8500=         J1=JN+1
8600=         JN=JN+30
8700=         IF(JN.GT.NBY) JN=NBY
8800=         GOTO 10
8900= 20   CONTINUE
9000=      GOTO 85
9100=C***                ERROR EXIT
9200= 80   CONTINUE
9300=      IEOJ=5
9400=C***                NORMAL EXIT
9500= 85   CONTINUE
9600=      IF(IFIL.EQ.4)NEXC=99
9700=      RETURN
9800=      END
```

```
35600=
35700=      SUBROUTINE XFEAT(FEAT,BUF,HID,CLAS,JDX,NBX,IEOJ)
35800=      COMMON/NAMES/INPF,IOUTF,IFEAT,KDUM,IDUM,ILOGF,IHIST
35900=      COMMON/FORM/IFIL,IFN,IGRIN,DUM,NBY,NDSENT
36000=      COMMON/CNUM/NEXC,NTC
36100=      COMMON/CREC/LASV,IEOFR,NBPR,LB,MAXV
36200=      COMMON/TRACE/ILOG,ITRACE
36300=      COMMON/CONX/LCID(51,3),SCALE(2),ICONF,ANS(3),IMAG
36400=      COMMON/TRANS/IBUF(16),IVAL,NVAL,LTYP,LADR(4),LINE(43),IBYTE
36500=      DIMENSION FEAT(JDX,1),BUF(NBX,1),HID(1)
36600=      DIMENSION CLAS(JDX,1)
36700=      EQUIVALENCE(FIKNO,IKNO),(NVEC,FVEC)
36800=      IB=0
36900=      JD=JDX-3
37000=      INIT=0
37100=      ISEQ=0
37200=      NUMV=0
37300=      NPV=1
37400=      NRB=0
37500=      NCMIN=LCID(51,1)
37600=      NCMAX=LCID(51,2)
37700=C***                READ REQUESTED RECORD
37800= 5    CONTINUE
37900=      IF(NEXC.GT.NTC.AND.IFN.EQ.3)GOTO 200
38000=      IF(NEXC.GT.NTC)GOTO 800
38100=      CALL RFEAT(FEAT,JDX,INIT,IEOJ)
38200=      IF(IEOJ.GT.0)GOTO 800
38300=      NRB=NRB+1
38400=      ISEQ=0
38500=C***                PROCESS FEATURE VECTORS
38600=      DO 90 L=1,LASV
38700=      CALL NEXVEC(ISEQ,NUMV,NRB,IEOJ)
38800=C     IF(L.NE.NUMV) GOTO 90
38900=      IF(IFN.EQ.3) GOTO 50
39000=C***                FOR STRIP/PLOT, LOAD VECTOR
39100= 7    CONTINUE
39200=      IF(IEOJ.LT.0) GOTO 15
39300=      IF(L.NE.NUMV) GOTO 12
39400=      IF(NPV.EQ.1.AND.IFN.EQ.2)NPV=2
39500=      DO 10 J=1,JD
39600= 10   BUF(J,NPV)=FEAT(J,L)
39700=      NPV=NPV+1
39800= 12   CONTINUE
39900=      IF(IEOFR.NE.0.AND.L.EQ.LASV)GOTO 15
40000=      IF(NPV.LE.NBY) GOTO 90
40100=C***                FLUSH BUFFER WHEN FULL
40200=C***                AND AT END RECORD
40300= 15   CONTINUE
40400=      IF(IFN.EQ.2)CALL PICT(BUF,HID,NBX,NPV)
40500=      IF(IFN.EQ.1)CALL STRIP(BUF,NBX,NPV)
40600=      NPV=1
40700=      IF(IEOJ.LT.0) GOTO 800
40800=      GOTO 90
40900=C***                TRANSMIT FEAT VECTOR
41000= 50   CONTINUE
41100=      IF(NEXC.GT.NTC) GOTO 200
41200=      IF(IEOJ.EQ.-9)  GOTO 200
41300=      IF(IEOJ.EQ.-5)GOTO 800
41400=      IF(L.NE.NUMV) GOTO 90
41500=      FIKNO=FEAT(JDX-1,L)
41600=      IKNO=IABS(IKNO)
41700=      FVEC=FEAT(JDX-2,L)
41800=      NVEC=MOD(NVEC,255)
41900=C***                SKIP 'ZAPPED' COMPONENTS
42000=      CALL XSCAL(CLAS(1,NCMIN),CLAS(1,NCMAX),FEAT(1,L),JD,SCALE,1)
42100=      IVAL=IKNO
42200=      CALL XMIT
42300=      IVAL=NVEC
42400=      CALL XMIT
42500=      KDSENT=0
42600=      DO 60 J=1,JD
42700=      IF(CLAS(1,NCSD).GT.1E6)GOTO 60
42800=      KDSENT=KDSENT+1
42900=      IVAL=FEAT(J,L)
43000=      CALL XMIT
43100= 60   CONTINUE
43200= 90   CONTINUE
43300=C***                END OF RECORD
43400= 95   CONTINUE
43500=      IF(IEOFR.EQ.0) GOTO 5
43600=      GOTO 800
43700=C***                FLUSH BYTE FILE AT END JOB
43800= 200  CONTINUE
43900=      NVAL=16
44000=      CALL XMIT
44100=      LTYP=3
44200=      NVAL=16
44300=      CALL XMIT
44400=      GOTO 800
44500=C***                ERROR EXIT
44600= 850  CONTINUE
44700=      IEOJ=1
44800=C***                NORMAL EXIT
44900= 800  CONTINUE
45000=      NDSENT=KDSENT
45100=      RETURN
45200=      END
```

```
52200=
52300=      SUBROUTINE NEXVEC(ISEQ,NVEC,NRB,IEOJ)
52400=      COMMON/NAMES/INPF,IOUTF,IFEAT,IUSER,IDUM,ILOGF,IHISG
52500=      COMMON/CREC/LASV,IEOFR,NBPR,LB,MAXV
52600=      COMMON/TRACE/ILOG,ITRACE
52700=      COMMON/CNUM/NEXC,NTC
52800=      DIMENSION LISV(15)
52900=      DATA IHI,IHL,IHS,IHC,IHE,IHQ/1HI,1HL,1HS,1HC,1HE,1HQ/
53000=C***
53100=C***              ALLOWS USER TO SPECIFY VECTOR NUMBERS
53200=C***              OPTIONS ARE AS FOLLOWS
53300=C***              I-->WITHIN INTERVALS  (VVV,VVV)...
53400=C***              L-->FROM LIST: VVV,VVV,...(15)
53500=C***              C-->CHOOSE CONSECUTIVE  TO LISV(1)
53600=C***              S-->SKIP CONSECUTIVE  TO LISV(1)
53700=C***              E-->EXIT XFEAT FOR THIS CLASS
53800=C***              Q-->QUIT FORMAT
53900=C***
54000=      IEOJ=0
54100=      NN=3
54200=      IF(ISEQ.NE.0) GOTO 100
54300=C***              INITIALIZE FOR NEW RECORD-SET/NEW CONTROLS
54400= 10   CONTINUE
54500=      NOW=0
54600=      IF(ILOG.GE.0)WRITE(IOUTF,900)NEXC,NRB
54700= 900  FORMAT(16H  SELECT  CLASS=,I3,5H RB= ,I2)
54800= 902  FORMAT(20H  PICK VECTORS FROM ,I3,4H TO ,I3/
54900=     1  19H  K,VVV,VVV,...(15)/2H $)
55000=C***              OPTION EXPIRED, RENEW IT
55100= 20   CONTINUE
55200=      WRITE(IOUTF,902)NOW+1,LASV
55300= 901  FORMAT(A1,15(1X,I3))
55400=      READ(INPF,901)KOPT,LISV
55500=C***              PROCESS FOLLOW-ON ENTRY
55600= 100  CONTINUE
55700=      IF(KOPT.EQ.IHE)IEOJ=-5
55800=      IF(KOPT.EQ.IHQ)IEOJ=-9
55900=      IF(KOPT.EQ.IHI) GOTO 200
56000=      IF(KOPT.EQ.IHL) GOTO 300
56100=      IF(KOPT.EQ.IHS.OR.KOPT.EQ.IHC) GOTO 400
56200=      IF(IEOJ.NE.0) GOTO 800
56300=C***              ERROR INPUT
56400=      IEOJ=-6
56500=      CALL ERR(IOUTF,NN,IEOJ)
56600=      IF(IEOJ.LT.0) GOTO 20
56700=      GOTO 800
56800=C***              GENERATES NEXT NVEC WITHIN GIVEN INTERVAL
56900=C***              STUB FOR EXPANSION
57000= 200  CONTINUE
57100=      GOTO 800
57200=C***              GENERATES NEXT NVEC FROM LIST
57300=C***              STUB FOR EXPANSION
57400= 300  CONTINUE
57500=      GOTO 800
57600=C***              GENERATES NEXT CONSECUTIVE NVEC UP TO LISV(1)
57700= 400  CONTINUE
57800=      NOW=NOW+1
57900=      IF(NOW.LE.LISV(1)) GOTO 480
58000=      NOW=NOW-1
58100=      GOTO 20
58200= 480  CONTINUE
58300=      NVEC=0
58400=      IF(KOPT.EQ.IHC)NVEC=NOW
58500=C***              NORMAL EXIT
58600= 800  CONTINUE
58700=      ISEQ=1
58800=      RETURN
58900=      END
```

```
59000=
59100=      SUBROUTINE NEXREC(IEOJ)
59200=      COMMON/FORM/IFIL,IFN,IGRIN,DUM,NBY,NDSENT
59300=      COMMON/CNUM/NEXC,NTC
59310=      COMMON/TRACE/ILOG,ITRACE
59400=      COMMON/NAMES/INPF,IOUTF,IFEAT,IOLDC,INEWC,ILOGF,IHISC
59500=C***              USER INPUT RECORD-ID=NEXC
59600=C***              NEXC=0--> NEXC=NEXC+1...
59700=C***              NEXC>LASC-->NORMAL EXIT
59800= 900  FORMAT(13H  NEXT-RECORD/15H  NN= RECORD-ID/2H *)
59900= 901  FORMAT(I2)
60000= 902  FORMAT(15H  NEXT RECORD= ,I2)
60100=      IF(IALL.EQ.1)GOTO 10
60200= 1    CONTINUE
60300=      WRITE(IOUTF,900)
60400=      READ (INPF,901) NEXC
60500=      IF(ILOG.NE.0)WRITE(ILOGF,902)NEXC
60600=      IF(NEXC.GT.LASC) GOTO 20
60700=      IF(NEXC.EQ.LASC.AND.IFIL.EQ.1) GOTO 20
60800=      IF(NEXC.EQ.0) GOTO 5
60900=      IEOJ=-6
61000=      CALL ERR(IOUTF,NN,IEOJ)
61100=      IF(IEOJ.GT.0) GOTO 30
61200=      GOTO 1
61300=C***              NEXC=0-->ALL
61400= 5    CONTINUE
61500=      IALL=1
61600=C***              INCREMENT NEXC
61700= 10   CONTINUE
61800=      NEXC=NEXC+1
61900=C***              NORMAL EXIT
62000= 20   CONTINUE
62100=      IEOJ=0
62200=      LASC=NEXC
62300=C***              ERROR EXIT
62400= 30   CONTINUE
62410=      IF(ILOG.EQ.1)WRITE(ITRACE,957)
62420=     1  NTC,NEXC,INEXC,LASC,IFIL,IEOJ
62430= 957  FORMAT(8H NEXREC ,6I6)
62500=      RETURN
62600=      END
62700=
62800=
```

```
33900=
34000=      SUBROUTINE XMIT
34100=      COMMON/TRANS/IBUF(16),IVAL,NVAL,LTYP,LADR(4),LINE(43),IBYTE
34200=C***     BUILDS BUFFER FOR INTEL FORMATTER(ILINE)
34300=      IF(NVAL.LT.16) GOTO 10
34400=        CALL ILINE
34500=        WRITE(IBYTE,900) LINE
34600= 900    FORMAT(43A1)
34700=      DO 5 I=1,16
34800= 5    IBUF(I)=0
34900=        LTYP=2
35000=        NVAL=0
35100= 10   CONTINUE
35200=      NVAL=NVAL+1
35300=      IBUF(NVAL)=IVAL
35400=      RETURN
35500=      END
```

```
51000=
51100=      SUBROUTINE STRIP(BUF,NX,NY)
51200=      DIMENSION BUF(NX,1)
51300=C***            STUB FOR STRIP CHART PLOTTING
51400=C
51500=C***            CLEAR BUF AT EXIT
51600=      DO 20 I=1,NX
51700=      DO 19 J=1,NY
51800= 19   BUF(I,J)=0
51900= 20   CONTINUE
52000=      RETURN
52100=      END
```

```
45400=
45500=      SUBROUTINE PICT(BUF,HID,NX,NY)
45600=      COMMON/FORM/IFIL,IFN,IGRIN,DUM,NBY,NDSENT
45700=      COMMON/CNUM/NEXC,NTC
45800=      COMMON/PL3D/BSC,ZSC,EL,ICPS,IGRID
45900=      COMMON/NAMES/INPF,IOUTF,IFEAT,IOLDC,INEWC,ILOGF,IHIST
45950=      COMMON/TRACE/ILOG,ITRACE
46000=      DIMENSION BUF(NX,1),HID(1),LNAM(3)
46100=      DATA LNAM(1),LNAM(2),LNAM(3)/4HFEAT,4HCLAS,4HHIST/
46200=      DATA IHG/1HG/
46300=C***              PLOT DRIVER-->INIT,PLOT,CLEAR BUFFER
46400=C***              PAUSE BEFORE PROCESS
46500= 904  FORMAT(14H  PLOT FOLLOWS,
46600=     1   31H  TYPE G TO GO ON NOW AND LATER/2H *)
46700= 905  FORMAT(A1)
46800=      WRITE(IOUTF,904)
46900=      READ(INPF,905) ICH
47000=      IF(ICH.NE.IHG) GOTO 30
47100=      IF(INO.EQ.999) GOTO 10
47200=C***              INITIALIZE ON FIRST CALL
47300=      INO=999
47400=      IGRID=2
47500=      IF(IGRIN.EQ.IHG)IGRID=3
47600=      CALL INITT(ICPS)
47700=C***              PLOT PICTURE
47800= 10   CONTINUE
47900=      CALL ERASE
48000=      CALL MOVABS(50,700)
48100=      CALL MOVABS(50,700)
48200=C***              LABEL PICTURE
48300=      CALL ANMODE
48400=      WRITE(IOUTF,901) LNAM(IFIL),NEXC
48500= 901  FORMAT(2X,A4,8H CLASS= ,I3
48600=     1 /  22H  ENTER SCALE CONTROLS,
48700=     2 /  15H  BB.B,ZZ.Z,EL./2H *)
48800=      READ(INPF,902)BSC,ZSC,EL
48900= 902  FORMAT(F4.1,1X,F4.1,1X,F3.0)
49000= 903  FORMAT(2X,F4.1,1X,F4.1,1X,F3.0)
49100=      IF(ILOG.NE.0)WRITE(ILOGF,901)NEXC,LNAM(IFIL)
49200=      IF(ILOG.NE.0)WRITE(ILOGF,903)BSC,ZSC,EL
49300=      CALL PLOT3D(BUF,HID,NX,NY)
49400=C***              PAUSE AND CHECK FOR EXIT(NON BLANK ICH)
49500=      CALL DCURSR(ICH,IX,IY)
49600=      IF(ICH.EQ.32) GOTO 10
49700=C***              CLEAR BUFFER AND RETURN
49800=      DO 20 N1=1,NX
49900=      DO 19 N2=1,NY
50000=      BUF(N1,N2)=0.
50100= 19   CONTINUE
50200= 20   CONTINUE
50300=      CALL ERASE
50400=      CALL MOVABS(10,750)
50500=      CALL ANMODE
50600=C***              EXIT
50700= 30   CONTINUE
50800=      RETURN
50900=      END

 9900=
10000=
10100=      SUBROUTINE FILBUF(HIST,BUF,NHX,J1,JN,NX,NY,ICON)
10200=      COMMON/TRACE/ILOG,ITRACE
10300=      DIMENSION HIST(NHX,1),BUF(NX,NY)
10400=      COMMON/NAMES/INPF,IOUTF,IFEAT,IOLDC,INEWC,ILOGF,IHISC
10500=      COMMON/HREC/ISYM,KTR,LABLH,NI,NBUC,JD
10600=C***              FILLS PLOT BUFFER WITH HISTOGRAM; PRINTS
10700=      N1=1
10800=      N2=8
10900=      K1=9
11000=      K2=NHX
11100=      IF(ICON.EQ.1) GOTO 20
11200=      IF(ICON.EQ.2) K2=KTR-1
11300=      IF(ICON.NE.2) GOTO 20
11400=      N1=KTR
11500=      N2=KTR+7
11600=      K1=KTR+8
11700=      K2=NHX
11800= 20   CONTINUE
11900=      DO 30 J=J1,JN
12000=      IF(ILOG.LE.1) GOTO 25
12100=      IF(ILOG.NE.0)WRITE(ILOGF,901) J,(HIST(N,J),N=N1,N2)
12200= 901  FORMAT(38H COMPONENT  NUM        AV      CSDL   ,
12300=     * 2X4HCSDR,7X3HMIN,7X3HMAX,6X4HMODE,5X5HVMODE/
12400=     1  1XI2,2H= ,8(1XE9.3))
12500=      IF(ILOG.NE.2) GOTO 25
12600=      IF(ILOG.NE.0)WRITE(ILOGF,902)(HIST(K,J),K=K1,K2)
12700= 902  FORMAT(  (((  20(1XF5.0)  )))   )
12800= 25   CONTINUE
12900=      DO 22 K=K1,K2
13000= 22   BUF(K,J)=HIST(K,J)
13100= 30   CONTINUE
13200=      RETURN
13300=      END
```

```
                        ;*************************TAPEIN*************************************000001
                        ;***************CASSETTE TAPE READ UTILITY FOR BOX80 SYSTEM*************000002
                        ;************     EQUATE STATEMENTS      ********************        000003
0487                         NMOUT  EQU   0487H         ;80/20 OUTPUTS 2-CHARS FOR BYTE      000380
003C                         GETCM  EQU   003CH         ;80/20 CMD PROCESSOR                 000390
0348                         GETCH  EQU   0348H         ;CHARACTER INPUT                     000391
02FE                         CNVBN  EQU   02FEH         ;CONVERTS TO BINARY                  000392
033A                         ERROR  EQU   033AH         ;ERROR HANDLER                       000393
3850                    ORG   3850H                     ;    CASSETTE TAPE READ              006770
3850  CD4803       XC05: CALL GETCH     ;READ A CHAR            000140                       000780
3853  FE3A          CPI ':'             ;START LINE PROCESSING  000150                       000790
3855  C25038        JNZ XC05            ;   AT THE FIRST ':'    000160                       000800
3858  AF            XRA A               ;SETUP                  000170                       000810
3859  57            MOV D,A             ;   FOR CHECKSUM        000180                       000820
385A  CD7F38        CALL BYTEX          ;READ PAIR OF HEX CHARS 000190                       000830
385D  CA3C00        JZ GETCM            ;QUIT A 0 RECORD LEN    000200                       000840
3860  5F            MOV E,A             ;SAVE RECLEN            000210                       000850
3861  CD7F38        CALL BYTEX          ;GET HI LOAD ADDRESS    000220                       000860
3864  67            MOV H,A             ;   AND SAVE IN H       000230                       000870
3865  CD7F38        CALL BYTEX          ;GET LO LOAD ADDRESS    000240                       000880
3868  6F            MOV L,A             ;   AND SAVE IN L       000250                       000890
3869  CD7F38        CALL BYTEX          ;GET RECORD TYPE        000260                       000900
386C  4B            MOV C,E             ;   AND SAVE IN C       000270                       000910
386D  CD7F38        XC10: CALL BYTEX    ;READ TAPE DATA, AND    000280                       000920
3870  77            MOV M,A             ;   STORE AT MEMORY LOC 000290                       000930
3871  23            INX H               ;GET NEXT MEM-LOC       000300                       000940
3872  1D            DCR E               ;DECR REC-LEN           000310                       000950
3873  C26D38        JNZ XC10            ;   AND LOOP            000320                       000960
3876  CD7F38        CALL BYTEX          ;GET CHECKSUM           000330                       000970
3879  C23A03        JNZ ERROR           ;  AND TEST LINE        000340                       000980
387C  C35038        JMP XC05            ;IF OK, GET NEXT LINE   000350                       000990
387F  C5      BYTEX: PUSH B             ;ASCII BYTE INPUT,      000360                       001000
3880  CD4803        CALL GETCH          ;READ CHAR              000370                       001010
3883  4F            MOV C,A             ;                       000380                       001020
3884  CDFE02        CALL CNVBN          ;CHAR-->HEXADECIMAL     000390                       001030
3887  07            RLC                 ; MOVE                  000400                       001040
3888  07            RLC                 ;   TO                  000410                       001050
3889  07            RLC                 ;     UPPER             000420                       001060
388A  07            RLC                 ;       4-BITS          000430                       001070
388B  47            MOV B,A             ;SAVE RESULT IN B       000440                       001080
388C  CD4803        CALL GETCH          ;READ CHARACTER         000450                       001090
388F  4F            MOV C,A             ;                       000460                       001100
3890  CDFE02        CALL CNVBN          ;CONVERT                000470                       001110
3893  B0            ORA B               ;OR IN UPPER 4 BITS     000480                       001120
3894  4F            MOV C,A             ;SAVE                   000490                       001130
3895  82            ADD D               ;INCREMENT CHECKSUM     000500                       001140
3896  57            MOV D,A             ;                       000510                       001150
3897  79            MOV A,C             ;RESTORE HEX TO A       000520                       001160
3898  C1            POP B               ;RESTORE B              000530                       001170
3899  C9            RET                 ;                       000540                       001180
                                        ;80/20 SYSTEM CODE      000550                       001190
```

```
8080 MACRO ASSEMBLER, VER 2.2  ERRORS = 0 PAGE 2

                ;  GETCH:  CALL CI      ;READS CHAR              000560              001200
                ;  ANI  7FH             ;TURNS OFF PARITY BIT    000570              001210
                ;  MOV C,A              ;RETURN IN CREG          000580              001220
                ;  RET                  ;                        000590              001230
                ;  CI: IN 0EDH          ;GET CONSOLE STATUS      000600              001240
                ;  ANI 02H              ;CHECK BUFFER RDY        000610              001250
                ;  JZ CI                ;   LOOP IF NOT          000620              001260
                ;  IN 0ECH              ;    INPUT IF RDY        000630              001270
                ;  RET                  ;                        000640              001280
                ;  CNVBN: MOV A,C       ; CONVERT TO BINARY      000650              001290
                ;  SUI '0'              ;SUBTR '0' CODE          000660              001300
                ;  CPI 10               ;CHECK 0-9 RESULT        000670              001310
                ;  RM                   ;IF SO, DONE             000680              001320
                ;  SUI 7                ;ELSE RESULT--> 7-23     000690              001330
                ;  RET                  ;                        000700              001340
                   END                  ;                        000710              001341
NO PROGRAM ERRORS


8080 MACRO ASSEMBLER, VER 2.2  ERRORS = 0 PAGE 3

                        SYMBOL TABLE

* 01

A       0007    B       0000    BYTEX   387F    C       0001
CNVBN   02FE    D       0002    E       0003    ERROR   033A
GETCH   0348    GETCM   003C    H       0004    L       0005
M       0006    NMOUT   0487 *  PSW     0006    SP      0006
XC05    3850    XC10    386D

    STOP
     2.078 CP SECONDS EXECUTION TIME
..
```

```
001 BLOCK01 0
002 A       0000007Q
003 B       0000000Q
004 BYTEX   034177Q
005 C       0000001Q
006 CNVBN   001376Q
007 D       0000002Q
008 E       0000003Q
009 ERROR   001472Q
010 GETCH   001510Q
011 GETCM   0000740
012 H       0000004Q
013 L       0000005Q
014 M       0000006Q
015 NMOUT   002207Q
016 PSW     0000006Q
017 SP      0000006Q
018 XC05    034120Q
019 XC10    034155Q
$
:10385000CD4803FE3AC25038AF57CD7F38CA3C003E
:10386000E5CD7F3867CD7F386FCD7F384BCD7F38C8
:10387000772310C26D38CD7F38C23A003C350038C597
:10388000CD48034FCDFE02D707070747CD48034F3A
:0A389000CDFE02B04F825779C1C986
:00000000000
$
```

APPENDIX J

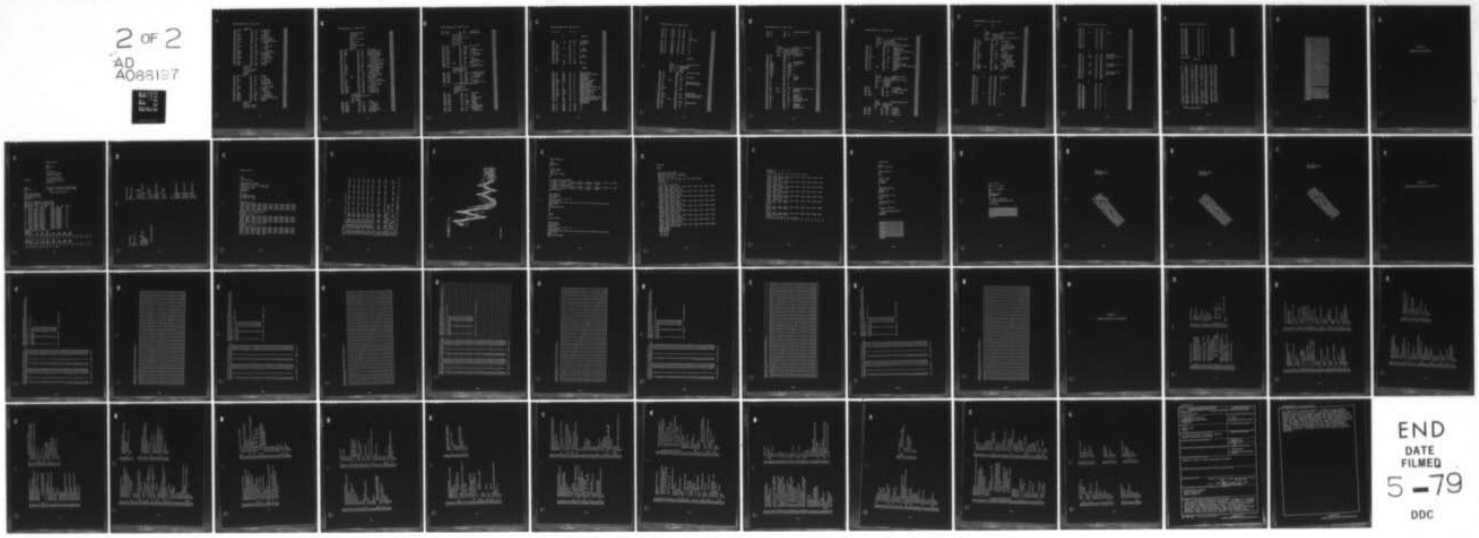
DECIDE Module

```
                        ;************************DECIDE*********************************  000100
                        ;*****************CLASSIFIER FOR 'BOX80' SYSTEM*****************  000200
                        ;*****************DESIGNED FOR INTEL 80/20 USE *****************  000300
                        ;************************************************************  000400
                        ;*************DATA STRUCTURES FOR DECIDE*************           000500
                        ;CLAS(J,I)  ICLAS(1ST)=1,CAV(1,I),CSDL(1,I),CSDR(1,I),...      000600
                        ;           ...,CAV(JD,I),CSDL(JD,I),CSDR(JD,I),...            000700
                        ;           ICLAS(LAS)=IC,CAV(1,IC),CSDL(1,IC),CSDR(1,IC),...  000800
                        ;           ...,CAV(JD,IC),CSDL(JD,IC),CSDR(JD,IC)   #         000900
                        ;FEAT(J,LL)  IKNO=1,IVEC=1,FEAT(1,LL),...FEAT(JD,LL),...       001000
                        ;           IKNO=1,IVEC=NV,FEAT(1,LL),...FEAT(JD,LL),...       001100
                        ;           IKNO=IC,IVEC=NV,FEAT(1,LL),...FEAT(JD,LL)   #      001200
                        ;                                                              001210
                        ;                                                              001220
                        ;                                                              001230
                        ;*******       EQUATE STATEMENTS                               001300
0000                    START:      DS      0           ;BEGIN PROGRAM LABLE           001310
                        ;***THESE EQUATES MUST BE RESET FOR EACH DATA SET               001320
3F80                    STACK       EQU     3F80H       ;INITIAL STACK ADDRESS(POINTER)001400
3F40                    PILE        EQU     3F40H       ;SPACE FOR STACK               001450
3D00                    FEAT        EQU     3D00H       ;BASE-A FOR FEAT FILE BLOCK    001500
3B80                    CLAS        EQU     3B80H       ;BASE-A FOR CLAS FILE DATA AREA001600
00ED                    USTAT       EQU     0EDH        ;USART STATUS PORT ADDRESS     001700
0001                    OK2GO       EQU     01H         ;READY STATUS                  001800
00EC                    UPORT       EQU     0ECH        ;USART PORT ADDRESS            001900
3800                    CODE        EQU     3800H       ;ADDRESS OF CODE BLOCK         001910
3940                    DATA        EQU     3940H       ;ADDRESS OF DATA BLOCK         001920
3980                    SUBS        EQU     3980H       ;ADDRESS OF SUBROUTINE BLOCK   001930
0003                    EJD         EQU     03H         ;NUMBER OF DIMENSIONS          001932
0003                    ELB         EQU     03H         ;NUMBER OF FEATURE VECTORS PER BLOCK 001934
0003                    EIC EQU     03H     ;NUMBER OF CLASSES                         001936
                        ;                                                              001940
                        ;                                                              001950
                        ;                                                              001960
3800                            ORG     CODE    ;                                      002000
                        ;***BEGIN PROCESS                                              002200
                        ;***GOTO IF7                                                   002300
3800  31803F    BP0:          LXI     SP,STACK    ;INITIALIZE STACK POINTER          002400
3803  3E03                    MVI     A,EJD       ;SET UP                            002410
3805  325E39                  STA     JD          ;  JD= NUMBER OF DIMENSIONS        002412
3808  3E03                    MVI     A,ELB       ;SET UP                            002414
380A  326739                  STA     LB          ;  NUMBER OF VECTORS PER BLOCK     002416
380D  3E03                    MVI     A,EIC       ;SET UP                            002418
380F  325F39                  STA     IC          ;  NUMBER OF CLASSES               002420
3812  C3F038                  JMP     IF7         ;GET NEXT FEAT BLOCK               002500
                        ;***NEXT FVEC                                                 002600
                        ;***LL=LL+1                                                   002700
                        ;***IF((LB-LL).LT.0) GOTO OS6                                 002800
                        ;***IKNO=FVEC(1,LL)                                           002900
                        ;***IVEC=FVEC(2,LL)                                           003000
                        ;***I=IC                                                      003100
```

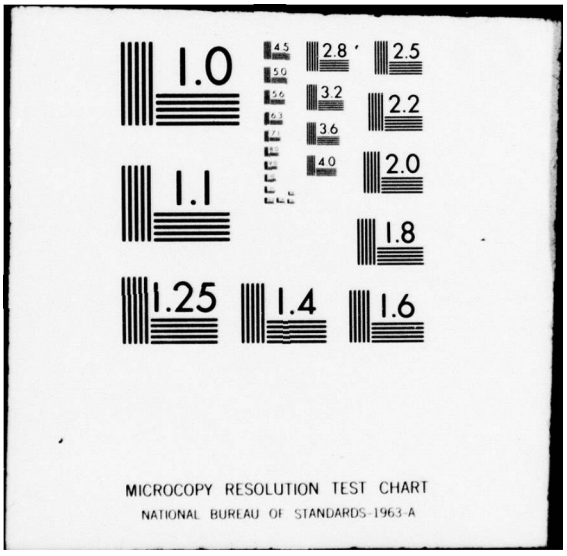
AD-A066 197 AIR FORCE INST OF TECH WRIGHT-PATTERSON AFB OHIO SCH--ETC F/G 5/8
A DEVELOPMENT SYSTEM FOR MICROPROCESSOR BASED PATTERN RECOGNIZE--ETC(U)
DEC 78 J R LEARY

UNCLASSIFIED AFIT/GCS/EE/78-12-VOL-2 NL

2 OF 2
AD
A066197



END
DATE
FILMED
5-79
DDC


1.0
1.1
1.25
1.4
1.6
1.8
2.0
2.2
2.5
2.8
3.2
3.6
4.0
4.5
5.0
5.6
6.3
MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

```
                        ;***MND=255                                                  003150
3815  215339   NV1:     LXI     H,LL    ;GET FEAT VECTOR                            003200
3818  46                MOV     B,M     ;   INDEX=LL,                               003300
3819  04                INR     B       ;     INCREMENT                             003400
381A  3A6739            LDA     LB      ;GET LB AND                                 003420
381D  90                SUB     B       ;   CALC (LB-LL)                            003430
381E  FADC38            JM      OS6     ;GOTO OS6 IF LT ZERO                        003500
3821  70                MOV     M,B     ;   ELSE SAVE NEW LL                        003600
3822  2A5839            LHLD    AFL     ;GET FEAT COMPONENT POINTER                 003700
3825  23                INX     H       ;   INCREMENT                               003800
3826  7E                MOV     A,M     ;      AND                                  003900
3827  325439            STA     IKNO    ;         SAVE IKNO                         004000
382A  23                INX     H       ;   INCREMENT                               004100
382B  7E                MOV     A,M     ;      AND                                  004200
382C  325739            STA     IVEC    ;         SAVE IVEC                         004300
382F  225A39            SHLD    AFL0    ;SAVE FEAT VECTOR BASE A  -1                004400
3832  21803B            LXI     H,CLAS  ;GET CLAS BASE-A                            004500
3835  2B                DCX     H       ;  AND DECREMENT                            004550
3836  226039            SHLD    ACLI    ;    TO SAVE A(CLAS)-1                      004600
3839  215F39            LXI     H,IC    ;GET NO. CLASSES(IC)                        004700
383C  7E                MOV     A,M     ;   AND SET                                 004800
383D  325D39            STA     I       ;      CLAS INDEX=I                         004900
3840  3EFF              MVI     A,0FFH  ;PRESET MND                                 004950
3842  326539            STA     MND     ;   TO 255                                  004960
                        ;***NEXT CLASS                                              005000
                        ;***I=I-1                                                   005100
                        ;***IF(I.EQ.0) GOTO OA5                                     005200
                        ;***ICLAS=CLAS(1,I)                                         005300
                        ;***J=JD                                                    005400
                        ;***MXF=0                                                   005450
3845  215D39   NC2:     LXI     H,I     ;GET CLAS                                   005500
3848  7E                MOV     A,M     ;   INDEX=I                                 005600
3849  3D                DCR     A       ;      DECREMENT                            005700
384A  FAB938            JM      OA5     ;GOTO OA5 IF ZERO                           005800
384D  77                MOV     M,A     ;   ELSE SAVE NEW I                         005900
384E  23                INX     H       ;GET NO DIMENSIONS(JD)                      006000
384F  7E                MOV     A,M     ;   AND SET                                 006100
3850  325C39            STA     J       ;      J-LIMIT ON FEATS                     006150
3853  2A6039            LHLD    ACLI    ;GET CLAS BASE-A  -1                        006200
3856  23                INX     H       ;  INCREMENT                                006250
3857  7E                MOV     A,M     ;   TO GET ICLAS                            006300
3858  326239            STA     ICLAS   ;      AND TO SAVE                          006400
385B  226039            SHLD    ACLI    ;      A(NEXT CAV)-1                        006500
385E  2A5A39            LHLD    AFL0    ;GET FEAT VECTOR BASE  -1                   006600
3861  225839            SHLD    AFL     ;   AND RESET COMPONENT POINTER             006700
3864  3E00              MVI     A,0D    ;PRESET MXF                                 006750
3866  326339            STA     MXF     ;   TO MIN                                  006760
                        ;***NEXT FEAT                                               006800
                        ;***J=J-1                                                   006900
                        ;***IF(J.EQ.0) GOTO DC4                                     0070000
                        ;***X=CAV(J,I),  X(=A)                                      007100
```

```
                        ;***Y=CSDL(J,I),  Y(=C)                                  007200
                        ;***Z=CSDR(J,I),  Z(=D)                                  007300
                        ;***F=FEAT(J,LL)                                         007400
                        ;***X=X-F                                                007500
                        ;***IF(X GE 0) GOTO NF3A                                 007600
                        ;***Y=Z                                                  007700
                        ;***NF3A: T=X/Y                                          007800
                        ;***T=X/Y                                                007900
                        ;***IF(MXF.GE.T) GOTO NF3                                008000
                        ;***  MXF=T                                              008100
                       ;***  GOTO NF3                                           008150
3869  215C39    NF3:        LXI     H,J       ;GET A(J-LIMIT)                   008200
386C  35                    DCR     M         ;  AND DECREMENT MEMORY           008250
386D  FAA038                JM      DC4       ;GOTO DC4 IF LT ZERO              008300
3870  2A6039                LHLD    ACLI      ;GET BASE-A FOR PROTO(J,I)        008400
3873  23                    INX     H         ;  BUMP TO NEXT ENTRY             008500
3874  7E                    MOV     A,M       ;     GET CAV(J,I)                008600
3875  23                    INX     H         ;  BUMP TO NEXT ENTRY             008700
3876  56                    MOV     D,M       ;     GET CSDL(J,I), DIVISOR      008800
3877  23                    INX     H         ;  BUMP TO NEXT ENTRY             008900
3878  4E                    MOV     C,M       ;     GET CSDR(J,I), DIVISOR      009000
3879  226039                SHLD    ACLI      ;SAVE BASE-A FOR PROTO(J,I)       009100
387C  2A5839                LHLD    AFL       ;GET FEAT VECTOR BASE             009200
387F  23                    INX     H         ;  BUMP TO NEXT COMPONENT         009300
3880  225839                SHLD    AFL       ;     SAVE NEW BASE               009400
3883  96                    SUB     M         ;X=CAV(J,I)-FEAT(J,L)             009500
3884  F28838                JP      NF3A      ;  GOTO NF3A IF POS               009600
3887  51                    MOV     D,C       ;SET Y=CSDR(J,I), DIVISOR         009700
3888  4F          NF3A:     MOV     C,A       ;ELSE Y=CSDL(J,I)                 009800
3889  AF                    XRA     A         ;THEN ZERO B TO                   009900
388A  47                    MOV     B,A       ;  SET BC = X, DIVIDEND           010000
388B  CD7F39                CALL    DIV       ;LET T=X/Y , QUOTIENT IN C-REG    010100
388E  1600                  MVI     D,0D      ;  AND SETUP                      010110
3890  59                    MOV     E,C       ;     IN DE FOR COMPARE           010120
3891  2A6339                LHLD    MXF       ;GET MXF INTO HL                  010200
3894  CDDA39                CALL    HILO      ;IF MXF GE T                      010300
3897  DA6938                JC      NF3       ;  GOTO NEXT FEAT                 010400
389A  226339                SHLD    MXF       ;ELSE SAVE T AT MXF               010500
389D  C36938                JMP     NF3       ;  GOTO NEXT FEAT                 010600
                        ;***DECIDE CLASS                                         010700
                        ;***IF(MXF.GT.MND) GOTO NEXT CLAS                        010800
                        ;***   MND=MXF                                           010900
                        ;***   ANS=ICLAS                                         011000
                        ;***   GOTO NC2                                          011100
38A0  2A6539    DC4:        LHLD    MND       ;GET MND                          011200
38A3  EB                    XCHG              ;  INTO DE AND                    011300
38A4  2A6339                LHLD    MXF       ;     MXF INTO HL                 011400
38A7  CDDA39                CALL    HILO      ;IF MXF GT MND                    011500
38AA  DA4538                JC      NC2       ;  GOTO NEXT CLAS                 011600
38AD  226539                SHLD    MND       ;ELSE SAVE MXF AT MND             011700
38B0  3A6239                LDA     ICLAS     ;GET ICLAS AND                    011800
```

```
38B3  325539              STA     ANS       ;   SAVE AT ANS                  011900
38B6  C34538              JMP     NC2       ;THEN GOTO NEXT CLAS             012000
                ;***OUTPUT ANSWER                                             012100
                ;***-->LL,IKNO,ANS                                            012200
                ;***IF(IKNO.EQ.ANS) GOTO OA5A                                  012300
                ;***   IOK=IOK+1                                               012400
                ;***   GOTO NV1                                                012500
                ;***OA5A:  ING=ING+1                                           012600
                ;***   GOTO NV1                                                012700
38B9  215339    OA5:      LXI     H,LL      ;GET A(LL) INTO HL                012800
38BC  114039              LXI     D,LINE    ;                                 012900
38BF  0613                MVI     B,19D     ;LINE LENGTH                      012950
38C1  CD013A              CALL    OUTB      ;    AND OUTPUT 3 VALUES          013000
38C4  3A5439              LDA     IKNO      ;GET IKNO                         013100
38C7  215539              LXI     H,ANS     ;   IF(ANS EQ IKNO)               013200
38CA  BE                  CMP     M         ;      GOTO OA5A                  013300
38CB  CAD538              JZ      OA5A      ;ELSE                             013400
38CE  216939              LXI     H,ING     ;   BUMP                          013500
38D1  34                  INR     M         ;   ERROR COUNT                   013600
38D2  C31538              JMP     NV1       ;AND GOTO NEXT F-VECTOR           013700
38D5  216839    OA5A:     LXI     H,IOK     ;BUMP                             013800
38D8  34                  INR     M         ;   CORRECT COUNT                 013900
38D9  C31538              JMP     NV1       ;AND GOTO NEXT F-VECTOR           014000
                ;***OUTPUT STATISTICS                                         014100
                ;***OUTPUT SUMMARY HEADER                                     014200
                ;***-->LB,IOK,ING                                             014300
38DC  217639    OS6:      LXI     H,SH01    ;GET A(SUM HEADER) INTO HL        014400
38DF  0609                MVI     B,9D      ;   AND NO. CHARS INTO B          014500
38E1  CDEC39              CALL    CLOOP     ;      THEN OUTPUT                014600
38E4  216739              LXI     H,LB      ;GET A(LB) INTO HL                014700
38E7  114039              LXI     D,LINE    ;                                 014800
38EA  0613                MVI     B,19D     ;LINE LENGTH                      014850
38EC  CD013A              CALL    OUTB      ;    AND OUTPUT 3 VALUES          014900
                ;**********************QUIT AFTER ONE BLOCK                   015000
38EF  CF                  RST     1         ;                                 015100
                ;***INITIALIZE FBLOCK                                          015200
                ;***IOK=ING=0                                                 015300
                ;***OUTPUT DECISION HEADER                                     015400
                ;***LL=LB                                                     015500
38F0  AF        IF7:      XRA     A         ;CLEAR A AND                      015600
38F1  326839              STA     IOK       ; ZERO                            015700
38F4  326939              STA     ING       ; VALUES                          015800
38F7  AF                  XRA     A         ;INITIALIZE VECTOR                016000
38F8  325339              STA     LL        ;   INDEX,LL                      016100
38FB  216B39              LXI     H,DH01    ;OUTPUT                           016200
38FE  060B                MVI     B,11D     ; DECISION                        016300
3900  CDEC39              CALL    CLOOP     ;  HEADER                         016400
                                            ;GET NEXT FEAT BLOCK              016450
3903  21003D              LXI     H,FEAT    ;RESET FEAT VECTOR                016500
3906  2B                  DCX     H         ; BASE ADDRESS                    016550
3907  225839              SHLD    AFL       ;  TO A(FEAT BLOCK)-1             016600
```

```
390A  C31538              JMP     NV1     ;                                   016800
                          ;                                                   016900
                          ;                                                   017000
                          ;                   WORK AREA                       017100
                          ;                                                   017200
                          ;                                                   017300
3940                      ORG     DATA    ; OUTPUT AREA                       017400
0000                LINE: DS      0       ;OUTPUT WORK AREA                   017500
3940  00                  DB      0       ;FIRST VALUE                        017600
3941  0000                DW      0       ;                                   017650
3943  0000                DW      0       ;                                   017660
3945  20            SP1:  DB      ' '     ;   SPACE                           017700
3946  00                  DB      0       ;SECOND VALUE                       017800
3947  0000                DW      0       ;                                   017850
3949  0000                DW      0       ;                                   017860
394B  20            SP2:  DB      ' '     ;   SPACE                           017900
394C  00                  DB      0       ;THIRD VALUE                        018000
394D  0000                DW      0       ;                                   018050
394F  0000                DW      0       ;                                   018060
3951  0A            SP3:  DB      0AH     ;CR                                 018100
3952  0D            SP4:  DB      0DH     ;LINE FEED                          018200
                          ;                                                   018300
                          ;                                                   018400
                          ;                   VARIABLES                       018500
                          ;                                                   018600
                          ;                                                   018700
3953  00            LL:    DB     0       ;POINTS TO FEAT VEC                 018800
3954  00            IKNO:  DB     0       ;A PRIORI CLASS OF F FEAT BLOCK     018900
3955  00            ANS:   DB     0       ;DECISION CLASS                     019000
3956  3D            EDEC:  DB     '='     ;END DECISION LINE                  019100
3957  00            IVEC:  DB     0       ;VECTOR NUMBER                      019200
3958  0000          AFL:   DW     0       ;POINTS TO F-VEC COMPONENT          019300
395A  0000          AFL0:  DW     0       ;FEATURE VECTOR BASE-A              019400
395C  00            J:     DB     0       ;FEATURE PROCESS LIMIT              019450
395D  00            I:     DB     0       ;POINTER SHOWS PROTO I              019600
395E  00            JD:    DB     0       ;NUMBER OF FEATURE DIMENSIONS  ***  019650
395F  00            IC:    DB     0       ;NUMBEROF CLASSES              ***  019660
3960  0000          ACLI:  DW     0       ;POINTS TO PROTOTYPE COMPONENT      019700
3962  00            ICLAS: DB     0       ;LABELS EACH PROTOTYPE              019800
3963  0000          MXF:   DW     0       ;R-INF MAG FOR DIS VECTOR           019900
3965  0000          MND:   DW     0       ;MINIMUM R-INF MAG                  020000
3967  00            LB:    DB     0       ;NUMBER F-VECS IN FEAT BLOCK   ***  020100
3968  00            IOK:   DB     0       ;COUNT OF CORRECT DECISIONS THIS BLOCK 020200
3969  00            INC:   DB     0       ;COUNT OF ERROR DECISIONS THIS B.   020300
396A  3D            ESUM:  DB     '='     ;END SUMMARY LINE                   020400
                          ;                DECISION HEADER                    020500
396B  44            DH01:  DB     44H     ; D                                 020600
396C  45            DH02:  DB     45H     ; E                                 020700
396D  43            DH03:  DB     43H     ; C                                 020800
396E  49            DH04:  DB     49H     ; I                                 020900
396F  53            DH05:  DB     53H     ; S                                 021000
```

```
3970  49        DH06:   DB      49H     ; I                             021100
3971  4F        DH07:   DB      4FH     ; O                             021200
3972  4E        DH08:   DB      4EH     ; N                             021300
3973  53        DH10:   DB      53H     ; S                             021400
3974  0D        DH11:   DB      0DH     ; LF                            021500
3975  0A        DH12:   DB      0AH     ; CR                            021600
                ;                       SUMMARY HEADER                  021700
3976  53        SH01:   DB      53H     ; S                             021800
3977  55        SH02:   DB      55H     ; U                             021900
3978  4D        SH03:   DB      4DH     ; M                             022000
3979  4D        SH04:   DB      4DH     ; M                             022100
397A  41        SH05:   DB      41H     ; A                             022200
397B  52        SH06:   DB      52H     ; R                             022300
397C  54        SH07:   DB      54H     ; Y                             022400
397D  0D        SH08:   DB      0DH     ; LF                            022500
397E  0A        SH09:   DB      0AH     ; CR                            022600
                ;                                                       022700
                ;                                                       022800
                ;                       SUBROUTINES                     022900
                ;                                                       023000
                ;                                                       023100
                ;                                                       028415
                ;***DIV                 ;DIVIDE (INTRP80, P18)          028420
                ;***INPUT   BC-  DIVIDEND UNSIGNED                      028430
                ;***        D -  DIVISOR UNSIGNED                       028435
                ;***OUTPUT  C -  QUOTIENT                               028440
                ;***        B -  REMAINDER                              028445
397F  F5        DIV:    PUSH    PSW     ;SAVE CALLER'S REGISTERS        028506
3980  D5                PUSH    D       ;                               028508
3981  E5                PUSH    H       ;                               028510
3982  1E09              MVI     E,9     ;LOOP COUNTER                   028512
3984  78                MOV     A,B     ;                               028514
3985  47        DIV0:   MOV     B,A     ;                               028516
3986  79                MOV     A,C     ;ROTATE CARRY INTO CREG         028518
3987  17                RAL             ;ROTATE NEXT MSB INTO CARRY     028520
3988  4F                MOV     C,A     ;                               028522
3989  1D                DCR     E       ;                               028524
398A  CA9739              JZ      DIV1    ;                               028526
398D  78                MOV     A,B     ;                               028528
398E  17                RAL             ;                               028530
398F  92                SUB     D       ;SUBTRACT DIVISOR.              028532
3990  D28539            JNC     DIV0    ;IF LT HI-QUO-BYTE, GOTO DIV0   028534
3993  82                ADD     D       ;OTHERWISE ADD DIVISOR BACK     028536
3994  C38539            JMP     DIV0    ;                               028538
3997  17        DIV1:   RAL             ;                               028540
3998  5F                MOV     E,A     ;                               028542
3999  3EFF              MVI     A,0FFH  ;COMPLEMENT QUOTIENT            028544
399B  A9                XRA     C       ;                               028546
399C  4F                MOVV     C,A    ;                               028548
399D  7B                MOV     A,E     ;                               028550
399E  1F                 RAR            ;                               028552
```

```
399F  E1              POP   H       ;RESTORE CALLER'S REGISTERS          028554
39A0  D1              POP   D       ;                                    028556
39A1  F1              POP   PSW     ;                                    028558
39A2  C9              RET           ;                                    028560
                ;                                                        028562
                ;                                                        028564
                ;                                                        028566
                ;                                                        028600
                ;                                                        028700
                ;                                                        028800
                ;***BNBCD                  ;BINARY TO BCD   (BB18-INTEL U.L.)  028900
                ;***INPUT:  UNSIGNED BINARY NUMBER IN DE                  029000
                ;***        POINTER TO LOWEST BUFFER LOC IN HL            029100
                ;***OUTPUT: 5 BCD DIGITS, ONE PER BYTE                    029200
                ;***        HL POINTS TO MSB IN LOWEST LOCATION           029300
39A3  F5        BNBCD:  PUSH  PSW     ;SAVE VARIABLES                    029400
39A4  C5              PUSH  B         ;                                  029500
39A5  D5              PUSH  D         ;                                  029600
39A6  E5              PUSH  H         ;                                  029700
39A7  EB              XCHG            ;GET NUMBER INTO HL                029800
39A8  01F0D8          LXI   B,-10000  ;  ADDR INTO DE                    029900
39AB  CDC939          CALL  DECNO     ;GET MSD                           030000
39AE  0118FC          LXI   B,-1000   ;                                  030100
39B1  CDC939          CALL  DECNO     ;                                  030200
39B4  019CFF          LXI   B,-100    ;                                  030300
39B7  CDC939          CALL  DECNO     ;                                  030400
39BA  01F6FF          LXI   B,-10     ;                                  030500
39BD  CDC939          CALL  DECNO     ;                                  030600
39C0  7D              MOV   A,L       ;GET LSD                           030700
39C1  F630            ORI   30H       ;FOR ASCII                         030750
39C3  12              STAX  D         ;STORE IT                          030800
39C4  E1              POP   H         ;                                  030900
39C5  D1              POP   D         ;                                  031000
39C6  C1              POP   B         ;                                  031100
39C7  F1              POP   PSW       ;                                  031200
39C8  C9              RET             ;                                  031300
39C9  3E30      DECNO:  MVI   A,30H   ;30H CONSTANT FOR ASCII, 0 FOR BCD  031500
39CB  D5              PUSH  D         ;SAVE ADDR                         031600
39CC  5D        DECN1:  MOV   E,L     ;SAVE BINARY                       031700
39CD  54              MOV   D,H       ;                                  031800
39CE  3C              INR   A         ;INCREMENT DIGIT                   031900
39CF  09              DAD   B         ;SUBTRACT                          032000
39D0  DACC39          JC    DECN1     ;RESULT NEGATIVE                   032100
39D3  3D              DCR   A         ;YES-->RESTORE DIGIT COUNT         032200
39D4  6B              MOV   L,E       ;BINARY NUMBER                     032300
39D5  62              MOV   H,D       ;                                  032400
39D6  D1              POP   D         ;AND ADDRESS                       032500
39D7  12              STAX  D         ;STORE DIGIT                       032600
39D8  13              INX   D         ;INCREMENT POINTER                 032700
39D9  C9              RET             ;                                  032800
                ;                                                        032900
```

```
                        ;                                                       033000
                        ;                                                       033100
                        ;***HILO                    ;16-BIT COMPARE(80/20 SYS)  033200
                        ;***INPUT:   DE- 16 BIT INTEGER                         033300
                        ;***         HL- 16 BIT INTEGER                         033400
                        ;***OUTPUT:  CARRY --)0 IF(HL<DE),ELSE--)1               033500
                        ;***         DESTROYS A AND F/F'S,  UNSIGNED NUMBERS ONLY 033600
39DA  C5        HILO:   PUSH    B           ;SAVE BC                            033700
39DB  47                MOV     B,A         ;SAVE A REGISTER                    033800
39DC  23                INX     H           ;INCREMENT HL BY ONE                033900
39DD  7C                MOV     A,H         ;WANT TO TEST FOR ZERO RESULT AFTER 034000
39DE  B5                ORA     L           ;/INCREMENTING                      034100
39DF  2B                DCX     H           ;RESTORE HL                         034200
39E0  37                STC                 ;SET CARRY                          034300
39E1  CAE939            JZ      HIL05       ;IF SO CARRY IS SET PROPERLY        034400
39E4  7D                MOV     A,L         ;IF NOT MOVE L TO A                 034500
39E5  93                SUB     E           ;SUBTRACT E                         034600
39E6  7C                MOV     A,H         ;MOV H TO A                         034700
39E7  9A                SBB     D           ;SUBTRACT D WITH BORROW             034800
39E8  3F                CMC                 ;COMPLEMENT CARRY FOR OK CARRY BIT VALUE034900
39E9  78        HIL05:  MOV     A,B         ;RESTORE A                          035000
39EA  C1                POP     B           ;RESTORE BC                         035100
39EB  C9                RET                 ;EXIT                               035200
                        ;                                                       035300
                        ;                                                       035400
                        ;                                                       035500
                        ;***CLOOP                   ;CHARACTER STRING OUTPUT    035600
                        ;***INPUT:   B- NUMBER OF CHARACTERS                    035700
                        ;***         HL- POINTER TO LOWEST CHARACTER IN STRING   035800
                        ;***OUTPUT:  TYPEWRITER OUTPUT                          035900
                        ;***          HL-DESTROED                               035950
                        ;***          B-DESTROYED                               035952
                        ;***          A-DESTROYED                               035954
39EC  05        CLOOP:  DCR     B           ;FEED COUT UNTIL                    036000
39ED  F8                RM                  ;  LAST CHARACTER, THEN EXIT        036100
39EE  4E                MOV     C,M         ;GET CHAR                           036200
39EF  CDF639            CALL    COUT        ;  TO OUTPUT ROUTINE                036300
39F2  23                INX     H           ;                                   036400
39F3  C3EC39            JMP     CLOOP       ;                                   036500
                        ;                                                       036600
                        ;                                                       036700
                        ;                                                       036800
                        ;***COUT                    ;CHARACTER OUTPUT(80/20  SYS ) 036900
                        ;***INPUT:   C-BYTE TO OUTPUT                           037000
                        ;***OUTPUT:  TYPEWRITER OUTPUT                          037100
                        ;***         A-DESTROYED                                037150
39F6  DBED      COUT:   IN      USTAT       ;CHECK READY                        037200
39F8  E601              ANI     OK2GO       ;  STATUS UNTIL                     037300
39FA  C2F639            JNZ     COUT        ;  OK TO WRITE                      037400
39FD  79                MOV     A,C         ;SEND BYTE                          037500
39FE  D3EC              OUT     UPORT       ;  TO OUTPUT PORT                   037600
```

```
3A00  C9                  RET             ;                                   037700
                          ;                                                   037800
                          ;                                                   037900
                          ;                                                   038000
                          ;***OUTB                  ;OUTPUT 1-BYTE INTEGERS   038100
                          ;***INPUT:   HL=A(VALUES LIST)                      038200
                          ;***         DE=ADDR(OUTPUT LINE)                    038300
                          ;***         B =LENGTH (IN BYTES) OF  LINE ENDING IN CR LF  038350
                          ;***OUTPUT:  TYPEWRITER OUTPUT                      038400
3A01  7E          OUTB:   MOV     A,M     ;GET NEXT VALUE                     038500
3A02  FE3D                CPI     '='     ;   IF EQ '='                       038600
3A04  CA1B3A              JZ      OUTBA   ;      GOTO OUTBA                   038700
3A07  E5                  PUSH    H       ;SAVE A(VALUES) ON STACK            038800
3A08  D5                  PUSH    D       ;SAVE A(BUFFER) ON STACK            038900
3A09  EB                  XCHG            ;HL NOW HAS A(BUFFER)               039000
3A0A  5F                  MOV     E,A     ;SETUP                              039100
3A0B  AF                  XRA     A       ;   DE WITH                         039200
3A0C  57                  MOV     D,A     ;      VALUE                        039300
3A0D  CDA339              CALL    BNBCD   ;CONVERT BIN TO ASCII               039400
3A10  D1                  POP     D       ;GET A(BUFFER) INTO D               039500
3A11  210600              LXI     H,06H   ;  BUMP BY 6 SLOTS                  039600
3A14  19                  DAD     D       ;      TO NEXT FIELD                039700
3A15  EB                  XCHG            ;         AND PUT INTO DE           039800
3A16  E1                  POP     H       ;GET A(VALUES) INTO HL              039900
3A17  23                  INX     H       ;   BUMP TO NEXT VALUE              040000
3A18  C3013A              JMP     OUTB    ;      AND GOTO NEXT VALUE          040100
3A1B  214639      OUTBA:  LXI     H,LINE  ;   OUTPUT                          040300
3A1E  CDEC39              CALL    CLOOP   ;     LINE                          040400
3A21  C9                  RET             ;EXIT                               040500
                          ;                                                   040600
                          ;                                                   040700
                          ;                                                   040800
3B80                      ORG     CLAS    ;CLAS FILE DATA                     040900
3B80  01          CI1:    DB      01D     ;CLAS 1 ID                          040910
3B81  0A                  DB      10D     ;                                   040920
3B82  05                  DB      05D     ;                                   040925
3B83  03                  DB      03D     ;                                   040930
3B84  14                  DB      20D     ;                                   040940
3B85  02                  DB      02D     ;                                   040945
3B86  09                  DB      09D     ;                                   040948
3B87  FF                  DB      255D    ;                                   040950
3B88  0A                  DB      10D     ;                                   040955
3B89  07                  DB      07D     ;                                   040957
3B8A  02          CI2:    DB      02D     ;CLAS 2 ID                          041000
3B8B  28                  DB      40D     ;                                   041010
3B8C  08                  DB      08D     ;                                   041012
3B8D  1B                  DB      27D     ;                                   041014
3B8E  42                  DB      66D     ;                                   041020
3B8F  0F                  DB      15D     ;                                   041022
3B90  08                  DB      08D     ;                                   041024
3B91  27                  DB      39D     ;                                   041036
```

```
3B92  2C                    DB      44D     ;                               041040
3B93  0C                    DB      12D     ;                               041042
3B94  03        C13:        DB      03D     ;CLAS 3 ID                      041100
3B95  FD                    DB      253D    ;                               041120
3B96  03                    DB      03D     ;                               041122
3B97  0F                    DB      15D     ;                               041124
3B98  17                    DB      23D     ;                               041125
3B99  91                    DB      145D    ;                               041140
3B9A  17                    DB      23D     ;                               041142
3B9B  13                    DB      19D     ;                               041144
3B9C  4D                    DB      77D     ;                               041160
3B9D  0C                    DB      12D     ;                               041162
3B9E  05                    DB      05D     ;                               041165
                ;                                                           041200
                ;                                                           041300
                ;                                                           041400
3D00                        ORG     FEAT    ;FEAT BLOCK DATA                041500
3D00  01        KNO1:        DB      01D    ;A PRIORI KNOWN AS 1-CLASS      041600
3D01  01        VEC1:       DB      01D     ;VECTOR NUMBER                  041602
3D02  05                    DB      05D     ;                               041605
3D03  10                    DB      16D     ;                               041607
3D04  C8                    DB      200D    ;                               041609
3D05  02        KNO2:       DB      02D     ;A PRIORI KNOWN AS 2 CLASS      041700
3D06  02        VEC2:       DB      02D     ;VECTOR NUMBER                  041702
3D07  3C                    DB      60D     ;                               041704
3D08  5A                    DB      90D     ;                               041706
3D09  50                    DB      80D     ;                               041708
3D0A  03        KNO3:       DB      03D     ;APRIORI KNOWN AS 3 CLASS       041800
3D0B  03        VEC3:       DB      03D     ;VECTOR NUMBER                  041802
3D0C  C8                    DB      200D    ;                               041900
3D0D  64                    DB      100D    ;                               041902
3D0E  5A                    DB      90D     ;                               041904
                ;                                                           042000
                ;                                                           042100
                ;                                                           042200
3F40                        ORG     PILE    ;                               042300
3F40  0000                  DW      0       ;                               042400
3F42  0000                  DW      0       ;                               042500
3F44  0000                  DW      0       ;                               042600
3F46  0000                  DW      0       ;                               042700
3F48  0000                  DW      0       ;                               042800
3F4A  0000                  DW      0       ;                               042900
3F4C  0000                  DW      0       ;                               043000
3F4E  0000                  DW      0       ;                               043100
3F50  0000                  DW      0       ;                               043200
3F52  0000                  DW      0       ;                               043300
3F54  0000                  DW      0       ;                               043400
3F56  0000                  DW      0       ;                               043500
3F58  0000                  DW      0       ;                               043600
3F5A  0000                  DW      0       ;                               043700
3F5C  0000                  DW      0       ;                               043800
```

```
3F5E  0000              DW      0       ;                       043900
3F60  0000              DW      0       ;                       044000
3F62  0000              DW      0       ;                       044100
3F64  0000              DW      0       ;                       044200
3F66  0000              DW      0       ;                       044300
3F68  0000              DW      0       ;                       044400
3F6A  0000              DW      0       ;                       044500
3F6C  0000              DW      0       ;                       044600
3F6E  0000              DW      0       ;                       044700
3F70  0000              DW      0       ;                       044800
3F72  0000              DW      0       ;                       044900
3F74  0000              DW      0       ;                       045000
3F76  0000              DW      0       ;                       045100
3F78  0000              DW      0       ;                       045200
3F7A  0000              DW      0       ;                       045300
3F7C  0000              DW      0       ;                       045400
3F7E  0000              DW      0       ;                       045500
                        END             ;                       045600
NO PROGRAM ERRORS
```

SYMBOL TABLE

\# 01

A	0007	ACLI	3960	AFL	3958	AFL0	395A
ANS	3955	B	0000	BNBCD	39A3	BP0	3800 #
C	0001	CJ1	3B80 #	CI2	3B8A #	CI3	3B94 #
CLAS	3B80	CLOOP	39EC	CODE	3800	COUT	39F6
D	0002	DATA	3940	DC4	38A0	DECN1	39CC
DECNO	39C9	DH01	396B	DH02	396C #	DH03	396D #
DH04	396E #	DH05	396F #	DH06	3970 #	DH07	3971 #
DH08	3972 #	DH10	3973 #	DH11	3974 #	DH12	3975 #
DIV	397F	DIV0	3985	DIV1	3997	E	0003
EDEC	3956 #	EIC	0003	EJD	0003	ELB	0003
ESUM	396A #	FEAT	3D00	H	0004	HIL05	39E9
HILO	39DA	I	395D	IC	395F	ICLAS	3962
IF7	38F0	IKNO	3954	ING	3969	IOK	3968
IVEC	3957	J	395C	JD	395E	KNO1	3D00 #
KNO2	3D05 #	KNO3	3D0A #	L	0005	LB	3967
LINE	3940	LL	3953	M	0006	MND	3965
MXF	3963	NC2	3845	NF3	3869	NF3A	3888
NV1	3815	OA5	38B9	OA5A	38D5	OKZCO	0001
OS6	38DC	OUTB	3A01	OUTBA	3A1B	PILE	3F40
PSW	0006	SH01	3976	SH02	3977 #	SH03	3978 #
SH04	3979 #	SH05	397A #	SH06	397B #	SH07	397C #
SH08	397D #	SH09	397E #	SP	0006	SP1	3945 #
SP2	394B #	SP3	3951 #	SP4	3952 #	STACK	3F80
START	0000 #	SUBS	3980 #	UPORT	00EC	USTAT	00ED
VEC1	3D01 #	VEC2	3D06 #	VEC3	3D0B #		

STOP
13.744 CP SECONDS EXECUTION TIME

```
$
:1038000031803F3E03325E393E033267393E033238
:1038100045F39C3F038215339460A3A673990FADCEE
:1038200038702A5839237E32543923 7E3257392250
:10383000 5A39218030D2D226039215F397E325D3934
:1038400003EFF326539215D397E3DF4B93877237EF6
:1038500032 5C392A6039237E3262392260392A5A31
:1038600039225839 3E0032639215C3935FAA038A3
:1038700042A6039237E2356234E2260392A58392361
:1038800022583996F28038514FAF47CD7F3916000C
:1038900059 2A6339CDDA39DA6938226339C369368C
:1038A0002A6539EB2A6339CDDA39DA4538226539A8
:1038B0003A623932 5539C3453821533911403906F6
:1038C00013CD013A3A5439215539ECAD538216948
:1038D0003934C31538216839 34C31538217639868F
:1038E00009CDEC392167391140390613CD013ACFA2
:1038F000AF3268393 26939AF325339216B39060B2F
:0D39000CDEC3921003D2B225839C315387C
:1039400000000000003C200000000000200000000037
:1039500000 0A0D0000003D0000000000000000000013
:1039600000000000000000000000003D4445434953B2
:1039700049 4F4E5300A53554D4D41525420DDAF5C2
:1039800D5E51E09704779174F1DCA9739781792E0
:1039900D2853982C38539175F3EFFA94F7B1FE16E
:1039A000D1F1C9F5C5D5E5EB01F0D8CDC93901187C
:1039B000FCCDC939019CFFCDC93901F6FFCDC9390C
:1039C0007DF63012E1D1C1F1C93E30D55D543C09DC
:1039D003DACC393D6B62D11213C9C5472370CB52BB4
:1039E00037CAE9397D937C9A3F78C1C905F84ECD35
:1039F000F63923C3EC39DBEDE601C2F63979D3ECB5
:103A0000C97EFE3DCA1B3AE5D5EB5FAF57CDA33962
:103A1000D12106019EBE123C3013A214039CDEC55
:023A200039C9A2
:103B8000010A050314020 9FF0A07022808 1B420F55
:0F3B900008272C0C03FD030F179117134D0C057D
:0F3D000001010510C80202 3C5A500303C8645A5F
:103F40000000000000000000000000000000000000000071
:103F50000000000000000000000000000000000000000061
:103F60000000000000000000000000000000000000000051
:103F70000000000000000000000000000000000000000041
:0000000000
$
```

APPENDIX K

Sample Execution Output

```
COMMAND- XCR9D,USER

CREATE
ENTER OPTIONS
*LIH1

ENTER PARAMETERS
 LABL,NBUF,JD,IC,NI,NE,MXV
*1000,0100,14,03,50,00,080

 APPROVE ADJUSTED PARAMETERS
OPTS LABL,NBUF,JD,IC,NI,NE,MXV
LIH1  1000  100 14  3 50  0  80
 ENTER Y OR N
*Y

 FEAT FILE HAS  1360 WORDS AND   80 VECTORS PER BLOCK
 DIST FILE HAS   812 WORDS AND 50 STEPS PER HISTOGRAM
     STOP
      5.306 CP SECONDS EXECUTION TIME
```

```
..COPTSBF,LOCF

..

CREATE
ENTER OPTIONS
*
 LIH1
 APPROVE ADJUSTED PARAMETERS
OPTS LABL,NBUF,JD,IC,NI,NE,MXV
LIH1  1000  100 14  3 50  1  80
 ENTER Y OR N
*
 FEAT FILE HAS  1360 WORDS AND   80 VECTORS PER BLOCK
 DIST FILE HAS   812 WORDS AND 50 STEPS PER HISTOGRAM
FIRST PASS SCAN STATISTICS ,LIM=   .500E-02  .919E+00
COMPONENT  NUM       AV      CSDL  CSDR       MIN      MAX     MODE     VMODE
 1=   .471E+03  .228E+03  .327E+03 0.        .539E+00  .726E+00 0.       0.
 2=   .471E+03  .331E+03  .394E+03 0.        .461E+00  .919E+00 0.       0.
 3=   .471E+03  .314E+02  .118E+03 0.        .134E+00  .604E+00 0.       0.
 4=   .471E+03  .282E+03  .362E+03 0.        .264E+00  .888E+00 0.       0.
 5=   .471E+03  .358E+02  .119E+03 0.        .920E-01  .677E+00 0.       0.
 6=   .471E+03  .250E+01  .301E+02 0.        .500E-02  .251E+00 0.       0.
 7=   .471E+03  .528E+01  .449E+02 0.        .220E-01  .374E+00 0.       0.
 8=   .471E+03  .691E+02  .178E+03 0.        .140E+00  .464E+00 0.       0.
 9=   .471E+03  .588E+01  .453E+02 0.        .100E-01  .306E+00 0.       0.
10=   .471E+03  .609E+02  .167E+03 0.        .103E+00  .459E+00 0.       0.
11=   .471E+03  .883E+01  .667E+02 0.        .530E-01  .357E+00 0.       0.
12=   .471E+03  .950E+01  .612E+02 0.        .380E-01  .339E+00 0.       0.
13=   .471E+03  .156E+02  .806E+02 0.        .730E-01  .399E+00 0.       0.
14=   .471E+03  .150E+02  .772E+02 0.        .440E-01  .402E+00 0.       0.
 HISTOGRAM FILE OPEN  HISC    1000   58   14
HISC RECORD     1     1     0     0    50
COMPONENT  NUM       AV      CSDL      CSDR      MIN       MAX      MODE      VMODE
 1=   .471E+03  .139E+01  .981E+00  .380E+02  .539E+00  .726E+00  .709E+00  .410E+00
   0.   0.   0.   0.   0.   0.   0.   0.   0.   0.   0.   0.   0.   0.   0.   0.   0.   0.   0.
   0.   0.   0.   0.   0.   0.   0.   0.   0.   1.   2.   1.   5.   3.  19.  37.  63.  57. 193.  9
   0.   0.   0.   0.   0.   0.   0.   0.   0.   0.
COMPONENT  NUM       AV      CSDL      CSDR      MIN       MAX      MODE      VMODE
 2=   .471E+03  .167E+01  .118E+01  .480E+02  .461E+00  .919E+00  .892E+00  .193E+00
   0.   0.   0.   0.   0.   0.   0.   0.   0.   0.   0.   0.   0.   0.   0.   0.   0.   0.   0.
   0.   0.   0.   0.   1.   0.   0.   0.   0.   0.   0.   1.   1.   2.   0.   4.   8.   4.   2.  1
  15.  25.  30.  36.  29.  32.  51.  64.  91.  58.
```

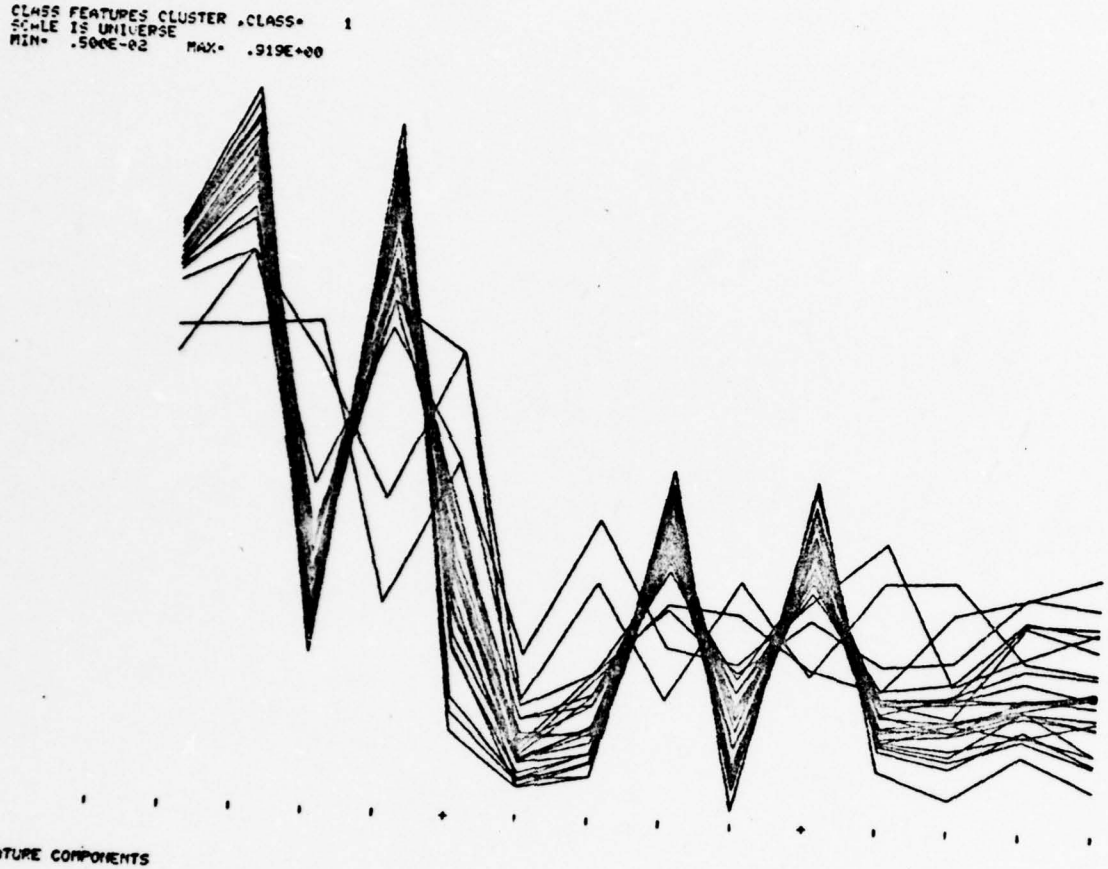
```
COMMAND- XDE9D,FEAT

 DEFINE
 ENTER MODE AND OPTIONS
*
 I,FL

  FEAT FILE    1000
  INITIAL CLAS FILE  100001
  ENTER PARAMETERS
  NB,NE,HUSKR
 *50,00,.0010

      STOP
       1.973 CP SECONDS EXECUTION TIME
```



```
COMMAND- XDE9D,FEAT,NEWC

 DEFINE
 ENTER MODE AND OPTIONS
*
 F,L

  FEAT FILE    1000
  CLAS FILE  100001
  NEXT-CLASS
  ENTER FUNCTION AND CLASS ID
  AAAA,H,+NN
 *KS      01

  SET  HUSK  VECTOR
  A, NV, NV,...(15)
 *A,023,034

  NEXT-CLASS
  ENTER FUNCTION AND CLASS ID
  AAAA,H,+NN
 *KS      01

  SET  HUSK  VECTOR
  A, NV, NV,...(15)
 *A,003

  ?

  A,035

  NEXT-CLASS
  ENTER FUNCTION AND CLASS ID
  AAAA,H,+NN
 *DEFN,H, 01

  NEXT-CLASS
  ENTER FUNCTION AND CLASS ID
  AAAA,H,+NN
 *REFN,H, 02

  NEXT-CLASS
  ENTER FUNCTION AND CLASS ID
  AAAA,H,+NN
 *QUIT
```

```
COMMAND- COPYSBF,LOGF


 I PL
  DEFINE
  ENTER MODE AND OPTIONS
 *
  OPENED FEATURE FILE WITH HEADER
  NAME,LABL,JD, LB,IC, MV,IOPT,IHIS,   FIRS,   FLAS
  FEAT 1000 17  80  3 198    5    1 .50E-02 .92E+00
  INITIAL CLAS FILE  100001
  ENTER PARAMETERS
  NB,NE,HUSKR
 *
  50  0 .0010
 APPROVE ADJUSTED PARAMETERS
  NB,NE,MAXRV,CLAS FILE SIZE
  50  1     0           238

  CLAS DEFN FOR    1 SYM=  1
  .672E+00  .778E+00  .290E+00  .706E+00  .314E+00  .917E-01  .125E+00  .341E+00  .129E+00  .322E+00
  .162E+00  .166E+00  .202E+00  .198E+00  .999E+99  .316E-01  .616E-01  .712E-01  .853E-01  .115E+00
  .339E-01  .458E-01  .559E-01  .562E-01  .593E-01  .445E-01  .561E-01  .638E-01  .689E-01  .999E+99
  .316E-01  .616E-01  .712E-01  .853E-01  .115E+00  .339E-01  .458E-01  .559E-01  .562E-01  .593E-01
  .445E-01  .561E-01  .638E-01  .689E-01  .999E+99
  HISTOGRAM FILE OPEN  HISC  100001   58   14
 HISC RECORD     1     1    30    21    50
  CLAS DEFN FOR    2 SYM=  1
  .715E+00  .884E+00  .211E+00  .825E+00  .197E+00  .359E-01  .623E-01  .408E+00  .629E-01  .379E+00
  .104E+00  .105E+00  .145E+00  .137E+00  .999E+99  .635E-02  .215E-01  .217E-01  .497E-01  .869E-01
  .146E-01  .282E-01  .365E-01  .433E-01  .444E-01  .269E-01  .409E-01  .426E-01  .611E-01  .999E+99
  .635E-02  .215E-01  .217E-01  .497E-01  .869E-01  .146E-01  .282E-01  .365E-01  .433E-01  .444E-01
  .269E-01  .409E-01  .426E-01  .611E-01  .999E+99
 HISC RECORD     2     1    30    21    50
  CLAS DEFN FOR    3 SYM=  1
  .709E+00  .876E+00  .227E+00  .809E+00  .211E+00  .470E-01  .783E-01  .405E+00  .770E-01  .379E+00
  .105E+00  .104E+00  .151E+00  .140E+00  .999E+99  .130E-01  .316E-01  .343E-01  .569E-01  .835E-01
  .186E-01  .315E-01  .390E-01  .443E-01  .451E-01  .292E-01  .435E-01  .498E-01  .598E-01  .999E+99
  .130E-01  .316E-01  .343E-01  .569E-01  .835E-01  .186E-01  .315E-01  .390E-01  .443E-01  .451E-01
  .292E-01  .435E-01  .498E-01  .598E-01  .999E+99
```

```
CLAS VECTOR TAGS
  1/0   1/1   2/0   2/1   3/0   3/1  98/* 99/*  1/3   1/2   0/0   0/0
 AT VEC=    1 CAV  CLAS=   1
.672E+00 .778E+00 .290E+00 .708E+00 .314E+00 .917E-01 .125E+00 .341E+00 .129E+00 .322E+00
.162E+00 .166E+00 .262E+00 .198E+00
 AT VEC=    2 CSDL CLAS=   1
.367E-01 .764E-01 .557E-01 .987E-01 .968E-01 .276E-01 .383E-01 .635E-01 .500E-01 .718E-01
.394E-01 .499E-01 .578E-01 .661E-01
 AT VEC=    3 CAV  CLAS=   2
.715E+00 .884E+00 .211E+00 .825E+00 .197E+00 .359E-01 .623E-01 .408E+00 .629E-01 .379E+00
.104E+00 .105E+00 .145E+00 .137E+00
 AT VEC=    4 CSDL CLAS=   2
.635E-02 .215E-01 .217E-01 .497E-01 .869E-01 .146E-01 .282E-01 .365E-01 .433E-01 .444E-01
.269E-01 .489E-01 .426E-01 .611E-01
 AT VEC=    5 CAV  CLAS=   3
.709E+00 .876E+00 .227E+00 .809E+00 .211E+00 .470E-01 .783E-01 .405E+00 .770E-01 .379E+00
.105E+00 .104E+00 .151E+00 .140E+00
 AT VEC=    6 CSDL CLAS=   3
.130E-01 .316E-01 .343E-01 .569E-01 .835E-01 .186E-01 .315E-01 .390E-01 .443E-01 .451E-01
.292E-01 .435E-01 .498E-01 .598E-01
 AT VEC=    7
CLAS=  98
.539E+00 .461E+00 .134E+00 .264E+00 .920E-01 .500E-02 .220E-01 .140E+00 .100E-01 .103E+00
.530E-01 .380E-01 .730E-01 .440E-01
 AT VEC=    8
CLAS=  99
.726E+00 .919E+00 .604E+00 .898E+00 .677E+00 .251E+00 .374E+00 .464E+00 .386E+00 .459E+00
.357E+00 .339E+00 .399E+00 .402E+00
 HUSK VECTOR=   3 AT  9 FOR CLASS    1 WITH   4 ENTRIES
 23  34  35   0   0   0   0   0   0   0   0   0   0   0
  AT VEC=   10 CSDR CLAS=   1
.264E-01 .481E-01 .877E-01 .717E-01 .133E+00 .404E-01 .533E-01 .480E-01 .624E-01 .481E-01
.493E-01 .620E-01 .698E-01 .717E-01
 LIST.IX FROM      1 OF     17
   13****** 1100  1201  2100  2201  3100  3201 98000 99000  1301  1202     0     0     0     0     0
    3     1     4    12     6     7     8     9    10     2     5    11    14    15    16    17    18
```


CLASS FEATURES CLUSTER .CLASS. 1
SCALE IS UNIVERSE
MIN. .500E-02 MAX. .919E+00
FEATURE COMPONENTS

```
COMMAND- XTR9B,FEAT,NEWC

 TRYOUT
 ENTER OPTIONS
*PNFSL

 FEAT FILE    1000
 CLAS FILE  100001
 SUBSET
 ENTER CLASS AND DIMENSIONS
 +CL
*088

 1LOG  FIGURES OF MERIT FOR DIMENSIONS
  6= .1920E+02   7= .1562E+02   1= .1546E+02  11= .1498E+02   9= .1364E+02  12= .1282E+02   8= .1199E+02   3= .1128E+02
 10= .1112E+02  13= .1065E+02   2= .9852E+01  14= .9692E+01   4= .8223E+01   5= .6761E+01
 2SUM  FIGURES OF MERIT FOR DIMENSIONS
  6= .1186E+07   1= .4927E+06   7= .1974E+06  11= .1784E+06   3= .7843E+05   9= .7090E+05  12= .6069E+05  13= .3495E+05
  8= .2434E+05   2= .2278E+05  14= .1852E+05  10= .1674E+05   4= .3880E+04   5= .3748E+04
 ENTER F/M SET NUMBER
*2

 SUMMARY CONCLUSION
 .6582  .3418 0.0000
 SUBSPACE TAGS
 6  1  7 11  3  9 12 13  8  2 14 10  4  5
 SUBSPACE ERROR RATES
 1=59/40  2=62/37  3=61/38  4=65/34  5=66/33  6=64/35  7=66/33  8=66/33  9=65/34 10=68/31 11=66/33 12=66/33
13=65/34 14=65/34
 SUBSET
 ENTER CLASS AND DIMENSIONS
 +CL
*099

 DD,DD,...
*01,02,99

 ENTER F/M SET NUMBER
*2

 SUMMARY CONCLUSION
 .7134  .2866 0.0000
 SUBSPACE TAGS
 6  1  7 11  3  9 12 13  8  2 14 10  4  5
 SUBSPACE ERROR RATES
 1=59/40  2=61/38  3=61/38  4=61/38  5=61/38  6=61/38  7=61/38  8=61/38  9=61/38 10=71/28 11=71/28 12=71/28
13=71/28 14=71/28
 SUBSET
 ENTER CLASS AND DIMENSIONS
 +CL
```

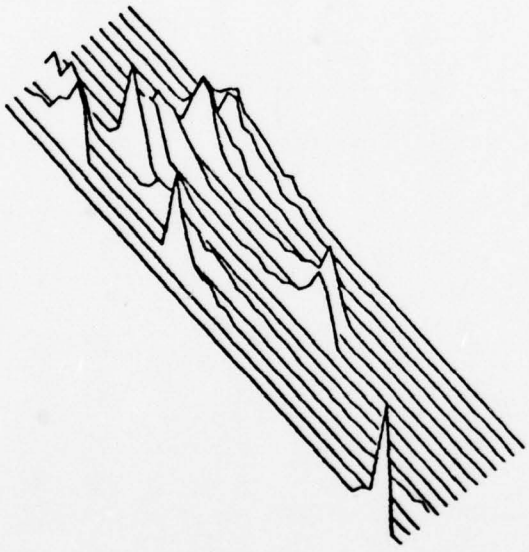
```
COPYSBF,LOCF

COMMAND-


 OPENED FEATURE FILE WITH  HEADER
 NAME,LABL,JD, LB,IC, MV,IOPT,IHIS, FIRS, FLAS
 FEAT 1000 17  80  3 198    5    1 .50E-02 .92E+00
 OPENED CLAS FILE WITH  HEADER
  NAME  LABLC    JDX    ICX    NTC   MBUC    MKV   NENT   NCIX  MSCAL  MZAPD
  CLAS 100001     17     14      3     50      0     17      0      0      0
 CLAS FILE INDEX
1=  1  2 -1, 2=  3  4 -1, 3=  5  6 -1,
  1  CLASS MERIT=   VMAG=  .17E+03
 .884E+02  .330E+02  .384E+02  .197E+02  .133E+02  .888E+02  .489E+02  .295E+02  .349E+02  .258E+02
 .467E+02  .329E+02  .276E+02  .221E+02
 CLASS  1  AS SEEN BY REST   VMAG=  .17E+03
 .942E+02  .154E+02  .739E+01  .116E+02  .377E+01  .934E+02  .368E+02  .198E+02  .236E+02  .151E+02
 .235E+02  .140E+02  .421E+01  .420E+01
   REST AS SEEN BY CLASS 1  VMAG=  .15E+03
 .894E+02  .340E+02  .394E+02  .207E+02  .143E+02  .898E+02  .499E+02  .305E+02  .359E+02  .268E+02
 .477E+02  .339E+02  .286E+02  .231E+02
  2  CLASS MERIT=   VMAG=  .36E+03
 .211E+03  .976E+02  .674E+02  .471E+02  .236E+02  .174E+03  .949E+02  .511E+02  .673E+02  .442E+02
 .893E+02  .626E+02  .518E+02  .415E+02
 CLASS  2  AS SEEN BY REST   VMAG=  .19E+03
 .522E+03  .300E+03  .159E+02  .372E+02  .189E+01  .568E+03  .126E+03  .359E+02  .534E+02  .226E+02
 .962E+02  .414E+02  .238E+02  .128E+02
   REST AS SEEN BY CLASS 2  VMAG=  .85E+03
 .123E+03  .646E+02  .290E+02  .274E+02  .103E+02  .856E+02  .460E+02  .216E+02  .324E+02  .183E+02
 .426E+02  .298E+02  .242E+02  .195E+02
  3  CLASS MERIT=   VMAG=  .58E+03
 .256E+03  .108E+03  .136E+03  .540E+02  .498E+02  .329E+03  .181E+03  .881E+02  .128E+03  .782E+02
 .177E+03  .123E+03  .102E+03  .828E+02
 CLASS  3  AS SEEN BY REST   VMAG=  .25E+03
 .387E+02  .151E+01  .249E+03  .317E+01  .445E+02  .151E+04  .483E+03  .837E+02  .245E+03  .729E+02
 .520E+03  .234E+03  .154E+03  .111E+03
   REST AS SEEN BY CLASS 3  VMAG=  .17E+04
 .450E+02  .104E+02  .687E+02  .684E+01  .253E+02  .154E+03  .860E+02  .370E+02  .609E+02  .340E+02
 .878E+02  .602E+02  .506E+02  .412E+02
 SUMMARY CONCLUSION
 .6582  .3418 0.0000
 CONFUSION MATRIX
  198  1 86  1 12
   82  2  9 47 42
  191  3 24 23 52
```

```
HISC RECORD     3     1    30    21    50
CLAS VECTOR TAGS
  1/0   1/1   2/0   2/1   3/0   3/1  98/*  99/*   0/0   0/0   0/0   0/0
  AT VEC=    1 CAV  CLAS=   1
 .672E+00  .778E+00  .290E+00  .708E+00  .314E+00  .917E-01  .125E+00  .341E+00  .129E+00  .322E+00
 .162E+00  .166E+00  .202E+00  .198E+00
  AT VEC=    2 CSDL CLAS=   1
 .316E-01  .616E-01  .712E-01  .853E-01  .115E+00  .339E-01  .458E-01  .559E-01  .562E-01  .593E-01
 .445E-01  .561E-01  .638E-01  .689E-01
  AT VEC=    3 CAV  CLAS=   2
 .715E+00  .684E+00  .211E+00  .825E+00  .197E+00  .359E-01  .623E-01  .408E+00  .629E-01  .379E+00
 .104E+00  .105E+00  .145E+00  .137E+00
  AT VEC=    4 CSDL CLAS=   2
 .635E-02  .215E-01  .217E-01  .497E-01  .869E-01  .146E-01  .282E-01  .365E-01  .433E-01  .444E-01
 .269E-01  .409E-01  .426E-01  .611E-01
  AT VEC=    5 CAV  CLAS=   3
 .709E+00  .876E+00  .227E+00  .809E+00  .211E+00  .470E-01  .783E-01  .405E+00  .770E-01  .379E+00
 .105E+00  .104E+00  .151E+00  .140E+00
  AT VEC=    6 CSDL CLAS=   3
 .130E-01  .316E-01  .343E-01  .569E-01  .835E-01  .186E-01  .315E-01  .390E-01  .443E-01  .451E-01
 .292E-01  .435E-01  .498E-01  .598E-01
  AT VEC=    7
CLAS=  98
 .539E+00  .461E+00  .134E+00  .264E+00  .920E-01  .500E-02  .220E-01  .140E+00  .100E-01  .103E+00
 .530E-01  .380E-01  .730E-01  .440E-01
  AT VEC=    8
CLAS=  99
 .726E+00  .919E+00  .604E+00  .888E+00  .677E+00  .251E+00  .374E+00  .464E+00  .306E+00  .459E+00
 .357E+00  .339E+00  .399E+00  .402E+00
 LIST.IX FROM      1 OF     17
   11******  1100  1201  2100  2201  3100  3201 98000 99000     0     0     0     0     0     0     0
    3     1     4     5     6     7     8     9    10     2    12    13    14    15    16    17    18
```

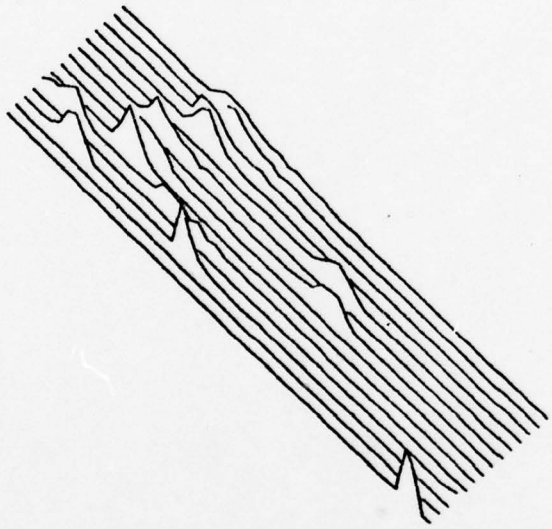
```
..XFR9E,FEAT,NEWC

 FORMAT
 ENTER FILE AND FUNCTION
*FX

 ENTER XMIT PARAMETERS
 AAAA
*D000

 FEAT FILE    1000
 CLAS FILE  100001
 NEXT-RECORD
 NN= RECORD-ID
*01

 SELECT  CLASS=  1 RB=  1
 PICK VECTORS FROM   1 TO  80
 K,VVV,VVV,...(15)
*C,001,009

 PICK VECTORS FROM   2 TO  80
 K,VVV,VVV,...(15)
*C,010

 PICK VECTORS FROM  11 TO  80
 K,VVV,VVV,...(15)
*Q

 XMIT OUTPUT  14 DIMENSIONS
     STOP
       .136 CP SECONDS EXECUTION TIME
..REWIND,BYTE
COPYSBF,BYTE

..


:10D0000000101C7BF44D25A4931A951A1455B4761DB
:10D0100000102C1B866B5464F489C52B25F5E958A0B
:10D0200000103937E94899A927966A0718498B1AE52
:10D0300000104AB9344AB77815DA17D925C6D5B639F
:10D0400000105D9CF3FC049494AB862BB44F444D0D
:10D0500000106F6CE25E11A351CDB74D0373D2C2E12
:10D0600000107DBAC42CD5E6F42B873AB35572B4B7L
:10D0700000108B7B93EC53C4E3AB53DB8524F50580D
:10D0800000109CAC95996D158726ADE476AC9AFC85D
:10D090000010AF2D73CD1503432B950AB3F574D616E
:1000000001FF
```

```
XFR:1,FEAT,NEWC

 FORMAT
 ENTER FILE AND FUNCTION
*CX

 ENTER XMIT PARAMETERS
 AAAA
*C000

 CLAS FILE  100001
 NEXT-RECORD
 NN= RECORD-ID
*00

 XMIT OUTPUT  14 DIMENSIONS
     STOP
          .144 CP SECONDS EXECUTION TIME
..REWIND,BYTE

..COPYSBF,BYTE


:10C000008528288122225526268523236132325A85
:10C01000232348212192E2C2C6630309D2A2A5C25B4
:10C020005602F2F6532326E3131F10808EC0C0C47
:10C030002905056141420262650F0F1D1414D33B
:10C040001C1C2D2525C61F1F2B161632222238215C
:10C05000214228285E91111E8111132121 2DF17171B
:10C06000342424281313281616D11E1E392626CF82
:10C070002020281818372424 3D272744 2A2A000015
:10000001FF

..
```

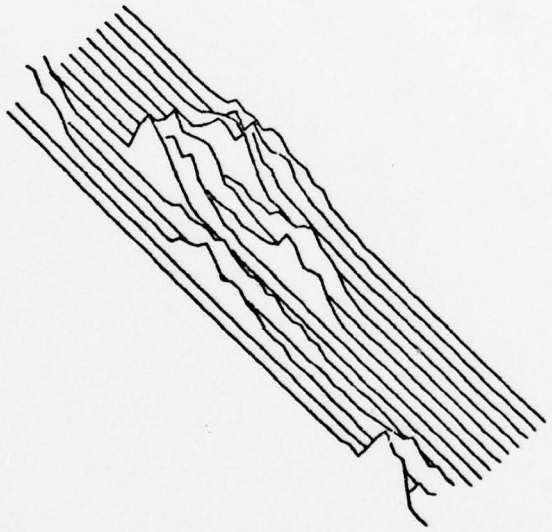
```
HIST CLASS=   3
ENTER SCALE CONTROLS
BB.E,ZZ.Z,EL.
*10.0,40.0,70.
```



```
HIST CLASS=   2
ENTER SCALE CONTROLS
 EE.E,ZZ.Z,EL.
$10.0,40.0,70.
```



```
HIST CLASS=   1
ENTER SCALE CONTROLS
EB.F,ZZ.Z,EL.
*10.0,-0.0,70.
```



APPENDIX L

Alphabet Classification Experiments

SUMMARY OF PROTOTYPE SET PERFORMANCE BY ALPHABET

ALPH	#MISTAKES	%CORRECT	ALPH	#MISTAKES	%CORRECT
1	12	57.4462	2	2	92.3077
3	1	96.1538	4	1	96.1538
5	3	88.4615	6	10	61.5385
7	8	69.2308	8	1	96.1538
9	1	96.1538	10	1	96.1538
11	15	42.3077	12	3	88.4615
13	6	76.9231	14	16	38.4615
15	2	92.3077	16	0	100.0000
17	3	88.4615	18	8	69.2308
19	7	73.0769	20	10	61.5385
21	1	96.1538	22	2	92.3077
23	1	96.1538	25	3	88.4615
25	1	96.1538	27	2	92.3077
28	0	100.0000	29	2	92.3077
30	3	88.4615	31	5	80.7692
32	4	84.6154	33	0	100.0000
34	1	96.1538	35	0	100.0000
36	4	84.6154	37	1	96.1538
38	13	50.0000	39	2	92.3077
41	5	80.7692	41	0	100.0000
42	9	65.3846	43	2	92.3077
44	7	73.0769	45	7	73.0769
46	3	88.4615	47	3	88.4615
48	0	100.0000	49	2	92.3077
50	2	92.3077	51	2	92.3077
52	8	69.2308	53	4	84.6154
54	10	61.5385	55	8	69.2308
56	6	76.9231	57	7	73.0769
58	1	96.1538	59	9	65.3846
60	5	80.7692	62	5	80.7692
63	3	76.9231	64	3	88.4615
65	7	73.0769	66	4	84.6154
67	12	53.8462	68	4	84.6154
69	2	92.3077	70	2	92.3077
71	8	69.2308	72	8	69.2308
73	2	92.3077	74	17	34.5154
75	2	92.3077	76	21	19.2308
77	11	57.6923	78	9	65.3846
79	6	76.9231	80	6	76.9231
81	13	50.0000	83	2	92.3077
84	7	73.0769	85	9	65.3846
86	9	65.3846	87	4	84.6154
88	6	76.9231	90	6	76.9231
91	1	96.1538	92	4	84.6154
93	4	84.6154	94	14	46.1538
95	3	88.4615	96	4	84.6154
97	2	92.3077	98	11	57.6923
99	2	92.3077	100	12	53.8462
102	2	92.3077	103	3	88.4615
104	0	100.0000	105	12	53.8462
107	3	88.4615	108	3	88.4615
109	4	84.6154	110	1	96.1538
111	7	73.0769	112	2	92.3077
113	1	96.1538	114	4	84.6154
115	9	65.3846	116	2	92.3077
117	1	96.1538	118	13	50.0000
119	5	80.7692	120	1	96.1538
121	2	92.3077	122	7	73.0769
123	3	88.4615	124	2	92.3077
125	8	69.2308	126	4	84.6154
127	7	73.0769	128	7	73.0769
129	10	61.5385	130	3	88.4615
131	7	73.0769	132	1	96.1538
133	4	84.6154	134	3	88.4615
135	2	92.3077	136	1	96.1538
137	5	80.7692	138	3	88.4615
139	0	100.0000	140	1	96.1538
141	1	96.1538	142	8	69.2308
143	2	92.3077	144	1	96.1538
145	2	92.3077	146	7	73.0769
147	6	76.9231	148	6	76.9231
149	1	96.1538	150	3	88.4615

TOTAL # MISTAKES= 689 TOTAL % CORRECT= 81.5972

RESULTS OF ISOLATED LETTERS COMPARISON
STANDARD OF COMPARISON IS EUCLIDEAN DISTANCE IN THE PROTOTYPE SPACE
RANGE OF COMPARISON TO ALPHABET 1 TO 150, LETTERS 1 TO 26
ALPHABETS EXCLUDED IN THE RANGE OF INTEREST ARE 24, 61, 82, 89,101,106,
PROTOTYPE FILE #17, LETTERS 1 TO 26
PROTOTYPE SPACE IS 49

SUMMARY OF PROTYPE SET PERFORMANCE BY LETTER

LETTER	#MISTAKES	%CORRECT
A	10	93.0556
B	39	72.9167
C	19	86.8056
D	21	85.4167
E	23	84.0278
F	18	87.5000
G	40	72.2222
H	34	76.3889
I	9	93.7500
J	28	80.5556
K	27	81.2500
L	5	96.5278
M	28	80.5556
N	25	82.6389
O	47	67.3611
P	25	81.9444
Q	49	65.9722
R	41	71.5278
S	19	86.8056
T	22	84.7222
U	35	75.6944
V	37	74.7056
W	25	82.6389
X	21	85.4167
Y	24	83.3333
Z	17	88.1944

TOTAL # MISTAKES= 689 TOTAL % CORRECT= 81.5972

CONFUSION MATRIX. ROWS ARE THE CORRECT LETTER AND THE COLUMNS ARE THE NUMBER OF TIMES THE LETTER WAS IDENTIFIED AS THE COLUMNATED LETTER

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
A	174	0	2	0	0	0	0	0	9	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B	0	135	0	0	13	0	0	5	1	4	2	1	4	0	0	0	0	2	4	0	0	0	1	0	0	1
C	0	1	125	0	0	0	7	0	5	1	0	2	0	0	2	0	1	0	0	0	0	0	0	0	0	0
D	2	5	0	123	2	0	0	2	2	1	0	3	2	0	4	0	1	0	0	0	0	0	0	0	0	0
E	2	1	0	0	121	6	1	0	6	0	0	0	1	0	0	0	0	3	0	0	0	0	2	1	0	2
F	0	0	0	0	0	125	0	3	10	0	0	0	0	0	0	5	0	0	0	0	0	0	2	0	1	0
G	0	4	13	2	0	0	104	0	1	0	0	0	0	0	3	0	6	0	5	0	0	0	1	0	0	0
H	0	5	0	0	3	2	0	110	0	3	0	1	5	0	0	0	0	6	2	0	6	0	0	0	0	0
I	0	0	0	0	0	0	0	0	135	1	1	1	0	0	0	0	0	0	0	2	0	0	0	4	0	0
J	0	0	0	0	0	0	0	0	22	116	0	0	0	0	0	0	1	0	0	1	4	0	0	0	0	0
K	0	1	0	0	4	2	0	1	10	0	117	3	0	0	0	0	0	1	1	0	0	0	1	3	0	1
L	0	0	0	0	0	0	0	0	3	0	2	139	0	0	0	0	0	0	0	0	0	0	0	0	0	0
M	0	2	0	0	0	1	0	5	3	1	0	0	116	0	0	0	0	0	2	0	2	8	1	0	0	0
N	0	1	0	0	0	0	0	4	2	1	0	1	9	119	0	0	0	0	1	0	0	3	1	2	0	0
O	0	3	10	12	0	0	1	2	1	2	0	0	0	0	97	0	13	0	0	0	3	0	0	0	0	0
P	0	0	0	1	0	19	0	0	3	1	0	0	1	0	0	116	0	0	0	0	0	0	1	0	0	0
Q	0	2	7	4	0	0	4	1	3	2	0	0	1	0	18	0	95	1	0	0	5	0	1	0	0	0
R	0	10	0	0	4	3	0	5	10	1	4	0	1	0	0	2	0	103	0	0	0	0	1	0	0	0
S	0	10	0	0	0	0	0	1	4	0	0	0	1	2	0	0	0	0	125	0	0	0	1	0	0	0
T	0	0	0	0	0	0	0	0	21	1	0	0	0	0	0	0	0	0	0	122	0	0	0	0	0	0
U	0	3	0	3	0	2	0	4	4	7	0	6	4	0	0	0	1	0	0	0	109	1	0	0	0	0
V	0	0	0	0	0	2	0	0	12	0	0	0	2	0	0	0	0	0	0	0	1	107	1	0	1	1
W	3	0	0	0	0	1	0	0	9	0	0	0	0	2	0	0	0	0	0	1	0	8	119	1	0	0
X	0	0	0	0	0	0	0	0	14	0	5	0	1	0	0	0	0	0	0	0	0	0	0	123	0	1
Y	0	0	0	0	0	0	0	0	13	0	0	0	0	0	0	0	0	0	0	1	0	8	0	0	120	2
Z	0	0	0	0	0	0	0	0	15	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	127

SUMMARY OF PROTOTYPE SET PERFORMANCE BY ALPHABET

ALPH	#MISTAKES	%CORRECT	ALPH	#MISTAKES	%CORRECT
1	5	80.7692	2	3	88.4615
3	1	96.1538	4	2	92.3077
5	3	88.4615	6	8	69.2308
7	7	73.0769	8	1	96.1538
9	0	100.0000	10	0	100.0000
11	8	69.2308	12	2	92.3077
13	2	92.3077	14	5	76.9231
15	2	92.3077	16	0	100.0000
17	1	96.1538	18	2	92.3077
19	1	96.1538	20	2	92.3077
21	1	96.1538	22	2	92.3077
23	1	96.1538	24	2	92.3077
25	1	96.1538	26	2	92.3077
27	0	100.0000	28	1	96.1538
29	2	92.3077	30	4	84.6154
31	1	96.1538	32	0	100.0000
33	1	96.1538	34	0	100.0000
35	3	88.4615	36	1	96.1538
37	6	76.9231	38	2	92.3077
39	5	73.0769	40	2	92.3077
41	2	92.3077	42	2	92.3077
43	3	88.4615	44	5	80.7692
45	5	80.7692	46	5	80.7692
47	3	88.4615	48	2	92.3077
49	2	92.3077	50	2	92.3077
51	2	92.3077	52	2	92.3077
53	5	80.7692	54	5	80.7692
55	7	73.0769	56	8	69.2308
57	8	69.2308	58	8	69.2308
59	2	92.3077	60	3	88.4615
61	7	73.0769	62	5	80.7692
63	3	88.4615	64	2	92.3077
65	2	92.3077	66	6	76.9231
67	1	96.1538	68	1	96.1538
69	4	84.6154	70	3	88.4615
71	2	92.3077	72	4	84.6154
73	0	100.0000	74	5	80.7692
75	5	76.9231	76	4	84.6154
77	1	96.1538	78	2	92.3077
79	7	73.0769	80	1	96.1538
81	7	73.0769	82	6	76.9231
83	3	88.4615	84	3	88.4615
85	1	96.1538	86	1	96.1538
87	3	88.4615	88	3	88.4615
89	5	80.7692	90	5	80.7692
91	1	96.1538	92	3	88.4615
93	5	80.7692	94	5	80.7692
95	3	88.4615	96	3	88.4615
97	2	92.3077	98	7	73.0769
99	4	84.6154	100	5	80.7692
101	2	92.3077	102	3	88.4615
103	5	80.7692	104	5	80.7692
105	2	92.3077	106	2	92.3077
107	1	96.1538	108	1	96.1538
109	2	92.3077	110	1	96.1538
111	1	96.1538	112	1	96.1538
113	9	65.3846	114	1	96.1538
115	2	92.3077	116	3	88.4615
117	1	96.1538	118	2	92.3077
119	4	84.6154	120	4	84.6154
121	1	96.1538	122	2	92.3077
123	5	76.9231	124	2	92.3077
125	1	96.1538	126	4	84.6154
127	8	69.2308	128	3	88.4615
129	7	73.0769	130	2	92.3077
131	4	84.6154	132	3	88.4615
133	1	96.1538	134	1	96.1538
135	4	84.6154	136	2	92.3077
137	0	100.0000	138	1	96.1538
139	2	92.3077	140	2	92.3077
141	2	92.3077	142	2	92.3077
143	2	92.3077	144	4	84.6154
145	5	76.9231	146	6	76.9231
147	1	96.1538	148	3	88.4615

TOTAL # MISTAKES= 429 TOTAL % CORRECT= 88.5417

RESULTS OF ISOLATED LETTERS COMPARISON
STANDARD OF COMPARISON IS EUCLIDEAN DISTANCE IN THE PROTOTYPE SPACE
RANGE OF COMPARISON IS ALPHABET 1 TO 150, LETTERS 1 TO 26
ALPHABETS EXCLUDED IN THE RANGE OF INTEREST ARE 24, 61, 82, 89,101,106,
PROTOTYPE FILE 417, LETTERS 1 TO 26
PROTOTYPE SPACE IS 49

SUMMARY OF PROTYPE SET PERFORMANCE BY LETTER

LETTER	#MISTAKES	%CORRECT
A	1	99.3056
B	13	87.5000
C	13	91.0722
D	15	89.5833
E	12	91.6667
F	5	96.5278
G	27	81.2500
H	20	86.1111
I	21	85.4167
J	33	77.7778
K	17	88.1944
L	5	96.5278
M	9	93.7500
N	19	86.8056
O	28	82.0556
P	5	96.5278
Q	40	72.2222
R	30	79.1667
S	5	96.5278
T	9	93.7500
U	25	91.9444
V	25	82.6389
W	15	89.5833
X	17	93.0556
Y	17	88.1944
Z	5	96.5278

TOTAL # MISTAKES= 429 TOTAL % CORRECT= 88.5417

CONFUSION MATRIX. ROWS ARE THE CORRECT LETTER AND THE COLUMNS ARE THE NUMBER OF TIMES THE LETTER WAS IDENTIFIED AS THE COLUMNATED LETTER

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
A	133	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B	0	125	0	0	2	0	0	9	0	1	2	0	2	0	0	1	0	1	0	0	0	0	0	0	0	0
C	0	0	131	4	0	0	6	0	0	0	0	0	0	0	2	0	1	0	0	0	0	0	0	0	0	0
D	0	0	0	129	0	0	0	2	0	0	0	0	0	0	3	0	0	0	0	0	2	0	0	0	0	1
E	0	1	1	2	132	4	0	0	3	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	0	0
F	0	0	0	0	0	139	0	0	5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
G	0	3	3	1	0	0	117	0	0	0	0	0	0	0	6	0	3	2	2	0	0	0	1	0	0	0
H	0	7	0	1	0	1	0	124	0	2	0	0	2	0	0	0	0	5	0	0	1	0	0	2	0	0
I	0	3	0	0	0	1	0	0	123	2	1	1	0	0	0	0	0	2	0	10	0	0	1	0	0	0
J	0	6	0	2	0	1	0	1	9	112	0	0	0	0	4	0	1	0	0	10	5	0	0	0	0	0
K	0	2	0	0	1	0	0	2	1	0	127	1	0	1	0	0	0	3	0	0	0	0	1	0	4	0
L	0	0	0	0	0	0	0	0	5	0	0	139	0	0	0	0	0	0	0	0	0	0	0	0	0	0
M	0	1	0	0	0	0	0	2	1	0	0	0	135	1	0	0	0	2	0	0	0	0	1	0	1	0
N	0	0	0	0	0	0	0	3	0	0	0	0	12	125	0	0	0	1	0	0	0	1	1	1	0	0
O	0	4	2	0	0	0	0	3	0	0	0	0	0	0	116	0	11	0	0	0	0	0	0	0	0	0
P	0	0	0	3	0	2	0	0	0	0	0	0	0	0	0	133	0	0	0	0	0	2	0	0	0	0
Q	0	1	3	4	0	0	3	1	0	1	0	0	1	0	23	0	104	1	0	0	0	1	0	1	0	0
R	0	13	0	2	0	1	0	7	2	0	2	0	0	0	0	3	0	114	0	0	0	0	0	0	0	2
S	0	2	0	2	0	0	0	0	0	1	0	0	0	1	0	0	0	0	139	0	0	0	1	0	0	0
T	0	0	0	0	0	0	0	0	8	1	0	0	0	0	0	0	0	0	0	135	0	0	0	0	0	0
U	0	5	0	3	0	1	0	5	2	1	0	2	3	0	1	0	0	1	0	0	118	2	0	0	0	0
V	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	119	3	0	19	1
W	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	8	129	0	0	2
X	3	0	0	0	0	0	0	0	2	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	134	0	1
Y	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	0	7	2	0	127	3
Z	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	139
	1	0	0	0	0	0	0	0	1	0	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

SUMMARY OF PROTOTYPE SET PERFORMANCE BY ALPHABET

ALPH #	#MISTAKES	%CORRECT	ALPH #	#MISTAKES	%CORRECT
1	5	80.7692	2	3	88.4615
3	1	96.1538	4	2	92.3077
5	3	88.4615	6	7	73.0769
7	7	73.0769	8	1	96.1538
9	0	100.0000	10	0	100.0000
11	8	69.2308	12	2	92.3077
13	2	92.3077	14	6	76.9231
15	2	92.3077	16	0	100.0000
17	1	96.1538	18	2	92.3077
19	1	96.1538	20	2	92.3077
21	1	96.1538	22	2	92.3077
23	1	96.1538	25	2	92.3077
26	1	96.1538	27	2	92.3077
28	0	100.0000	29	1	96.1538
30	2	92.3077	31	4	84.6154
32	1	96.1538	33	0	100.0000
34	1	96.1538	35	0	100.0000
36	2	92.3077	37	1	96.1538
38	6	76.9231	39	2	92.3077
40	6	76.9231	41	2	92.3077
42	2	92.3077	43	2	92.3077
44	3	88.4615	45	5	80.7692
46	3	88.4615	47	5	80.7692
48	3	88.4615	49	2	92.3077
51	2	92.3077	51	2	92.3077
52	3	88.4615	53	2	92.3077
54	8	69.2308	55	5	80.7692
56	7	73.0769	57	8	69.2308
58	1	96.1538	59	8	69.2308
60	2	92.3077	62	3	88.4615
63	7	73.0769	64	5	80.7692
65	3	88.4615	66	2	92.3077
67	2	92.3077	68	6	76.9231
69	1	96.1538	70	1	96.1538
71	4	84.6154	72	3	88.4615
73	2	92.3077	74	5	80.7692
75	0	100.0000	76	6	76.9231
77	6	76.9231	78	4	84.6154
79	7	73.0769	80	2	92.3077
81	7	73.0769	83	1	96.1538
84	3	88.4615	85	6	76.9231
86	3	88.4615	87	2	92.3077
88	3	88.4615	90	1	96.1538
91	1	96.1538	92	3	88.4615
93	4	84.6154	94	4	84.6154
95	1	96.1538	96	3	88.4615
97	2	92.3077	98	7	73.0769
99	4	84.6154	100	5	80.7692
102	2	92.3077	103	3	88.4615
104	2	92.3077	105	5	80.7692
107	3	88.4615	108	1	96.1538
109	1	96.1538	110	1	96.1538
111	2	92.3077	112	1	96.1538
113	1	96.1538	114	1	96.1538
115	3	65.3846	116	1	96.1538
117	2	92.3077	118	4	84.6154
119	1	96.1538	120	2	92.3077
121	4	84.6154	122	5	80.7692
123	1	96.1538	124	2	92.3077
125	6	76.9231	126	2	92.3077
127	1	96.1538	128	4	84.6154
129	5	80.7692	130	3	88.4615
131	7	73.0769	132	2	92.3077
133	4	84.6154	134	3	88.4615
135	1	96.1538	136	1	96.1538
137	4	84.6154	138	1	96.1538
139	0	100.0000	140	1	96.1538
141	1	96.1538	142	2	92.3077
143	1	96.1538	144	1	96.1538
145	2	92.3077	146	4	84.6154
147	6	76.9231	148	5	80.7692
149	1	96.1538	150	3	88.4615

TOTAL # MISTAKES= 419 TOTAL % CORRECT= 88.8088

RESULTS OF ISOLATED LETTERS COMPARISON
STANDARD OF COMPARISON IS EUCLIDEAN DISTANCE IN THE PROTOTYPE SPACE
RANGE OF COMPARISON IS ALPHABET 1 TO 150, LETTERS 1 TO 26
ALPHABETS EXCLUDED IN THE RANGE OF INTEREST ARE 24, 61, 82, 89, 101, 106,
PROTOTYPE FILE #17, LETTERS 1 TO 26
PROTOTYPE SPACE IS 49

SUMMARY OF PROTYPE SET PERFORMANCE BY LETTER

LETTER	#MISTAKES	%CORRECT
A	1	99.3056
B	17	88.1944
C	13	90.9722
D	15	89.5833
E	9	93.7500
F	4	97.2222
G	27	81.2500
H	20	86.1111
I	22	84.7222
J	34	76.3889
K	16	88.8889
L	5	96.5278
M	9	93.7500
N	18	87.5000
O	23	80.5556
P	5	96.5278
Q	37	74.3056
R	30	79.1667
S	5	96.5278
T	8	94.4444
U	26	81.9444
V	25	82.6389
W	14	90.2778
X	10	93.0556
Y	16	88.8889
Z	5	96.5278

TOTAL # MISTAKES= 419 TOTAL % CORRECT= 88.8088

CONFUSION MATRIX. ROWS ARE THE CORRECT LETTER AND THE COLUMNS ARE THE NUMBER OF TIMES THE LETTER WAS IDENTIFIED AS THE COLUMNATED LETTER

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
A	141	0	1	0	0	0	0	0	0	2	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B	0	132	0	1	7	0	0	0	0	1	1	0	0	0	0	2	1	3	0	0	0	0	0	2	0	0
C	0	1	141	2	1	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
D	0	2	1	133	0	0	0	0	0	0	0	0	0	0	1	0	5	0	0	0	1	0	0	0	0	0
E	0	0	2	0	139	5	0	0	1	0	1	0	0	0	0	0	0	0	1	0	0	0	0	2	0	0
F	0	0	1	0	0	142	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
G	0	0	0	1	0	0	126	0	0	0	0	0	0	0	0	0	7	1	0	0	0	0	0	0	0	0
H	0	5	0	1	0	1	0	129	0	4	1	0	0	0	0	0	1	3	0	0	0	0	0	0	0	0
I	0	0	0	0	0	0	0	0	127	11	2	0	0	0	0	0	0	0	0	1	0	0	1	2	0	0
J	0	1	1	0	1	1	0	0	2	139	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	1
K	0	0	0	0	0	0	0	0	0	0	139	0	1	0	0	0	0	3	0	0	0	0	1	0	0	0
L	0	2	0	0	0	0	0	0	1	0	1	142	0	0	0	0	0	0	0	0	0	0	0	0	0	0
M	0	0	0	0	0	2	0	0	0	0	2	0	138	1	0	0	0	1	0	0	0	1	0	1	0	0
N	0	0	0	0	0	0	0	0	0	0	0	0	5	135	0	0	1	0	0	0	0	3	0	0	0	0
O	0	0	2	5	0	0	0	2	0	1	0	0	0	0	92	0	42	0	0	0	0	0	0	0	0	0
P	0	0	0	0	0	3	0	0	0	0	0	0	0	0	0	133	3	0	0	0	0	0	0	0	0	0
Q	0	0	5	1	0	0	1	0	0	1	0	0	0	0	2	0	132	0	0	0	2	0	0	0	0	0
R	0	4	0	0	1	1	0	1	0	0	3	0	0	0	0	0	1	137	0	0	0	0	0	0	0	0
S	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	144	0	0	0	0	0	0	0
T	0	0	1	0	0	0	0	0	3	3	0	0	0	0	0	0	0	0	0	137	0	0	0	2	1	0
U	0	0	2	0	0	1	0	0	0	9	0	0	0	0	0	0	0	1	0	0	129	0	0	0	0	0
V	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	136	0	0	8	0
W	2	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	136	0	0	0
X	0	0	2	0	0	0	0	0	0	0	10	0	0	0	0	0	0	0	0	2	1	14	124	0	1	0
Y	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	134	0	0
Z	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	2	0	0	0	142	0
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	143

SUMMARY OF PROTOTYPE SET PERFORMANCE BY ALPHABET

ALPH	#MISTAKES	%CORRECT	ALPH	#MISTAKES	%CORRECT
1	5	80.7692	2	2	92.3077
3	1	95.1538	4	2	92.3077
5	3	88.4615	6	7	73.0769
7	6	75.3231	8	1	95.1538
9	0	100.0000	10	1	100.0000
11	8	69.2308	12	2	92.3077
13	2	92.3077	14	4	84.6154
15	2	92.3077	16	0	100.0000
17	1	95.1538	18	2	92.3077
19	1	95.1538	20	2	92.3077
21	1	95.1538	22	1	95.1538
23	1	95.1538	25	2	92.3077
25	1	95.1538	27	2	92.3077
27	0	100.0000	29	1	95.1538
28	2	92.3077	31	4	84.6154
32	1	95.1538	33	0	100.0000
34	1	95.1538	35	0	100.0000
36	2	92.3077	37	1	95.1538
38	6	75.3231	39	2	92.3077
40	3	88.4615	41	2	92.3077
42	2	92.3077	43	2	92.3077
44	2	92.3077	45	5	80.7692
46	3	88.4615	47	3	88.4615
48	1	95.1538	49	2	92.3077
50	2	92.3077	51	2	92.3077
52	3	88.4615	53	2	92.3077
54	7	73.0769	55	5	80.7692
56	7	73.0769	57	9	65.3846
58	1	95.1538	59	8	69.2308
60	2	92.3077	62	3	88.4615
63	7	73.0769	64	6	76.9231
65	3	88.4615	66	3	88.4615
67	2	92.3077	68	5	80.7692
69	1	95.1538	70	1	95.1538
71	4	84.6154	72	3	88.4615
73	2	92.3077	74	5	80.7692
75	0	100.0000	76	6	76.9231
77	5	75.3231	78	4	84.6154
79	7	73.0769	80	3	88.4615
81	7	73.0769	83	1	95.1538
84	3	88.4615	85	6	76.9231
86	3	88.4615	87	2	92.3077
88	3	88.4615	90	1	96.1538
91	1	95.1538	92	3	88.4615
93	5	80.7692	94	5	80.7692
95	1	95.1538	96	3	88.4615
97	2	92.3077	98	7	73.0769
99	4	84.6154	100	5	80.7692
102	2	92.3077	103	2	92.3077
104	2	92.3077	105	5	80.7692
107	3	88.4615	108	2	92.3077
109	1	95.1538	110	1	96.1538
111	2	92.3077	112	1	96.1538
113	1	95.1538	114	1	96.1538
115	7	73.0769	116	1	96.1538
117	1	95.1538	118	4	84.6154
119	1	95.1538	120	2	92.3077
121	4	84.6154	122	5	80.7692
123	1	95.1538	124	2	92.3077
125	6	75.3231	126	2	92.3077
127	1	95.1538	128	4	84.6154
129	4	84.6154	130	3	88.4615
131	5	80.7692	132	2	92.3077
133	4	84.6154	134	3	88.4615
135	1	95.1538	136	1	96.1538
137	4	84.6154	138	2	92.3077
139	0	100.0000	140	1	96.1538
141	1	95.1538	142	2	92.3077
143	1	95.1538	144	1	96.1538
145	2	92.3077	146	4	84.6154
147	7	73.0769	148	6	76.9231
149	1	95.1538	150	2	92.3077

TOTAL # MISTAKES= 407 TOTAL % CORRECT= 89.1293

RESULTS OF ISOLATED LETTERS COMPARISON
STANDARD OF COMPARISON IS EUCLIDEAN DISTANCE IN THE PROTOTYPE SPACE
RANGE OF COMPARISON IS ALPHABET 1 TO 150, LETTERS 1 TO 26
ALPHABETS EXCLUDED IN THE RANGE OF INTEREST ARE 24, 61, 82, 89,101,106,
PROTOTYPE FILE #17, LETTERS 1 TO 26
PROTOTYPE SPACE IS 49

SUMMARY OF PROTYPE SET PERFORMANCE BY LETTER

LETTER	#MISTAKES	%CORRECT
A	2	98.6111
B	18	87.5000
C	12	91.6667
D	13	90.9722
E	8	94.4444
F	3	97.9167
G	25	81.9444
H	22	84.7222
I	20	86.1111
J	33	77.0833
K	13	90.9722
L	2	98.6111
M	9	93.7500
N	17	88.1944
O	26	81.9444
P	5	96.5278
Q	39	72.9167
R	29	79.8611
S	4	97.2222
T	8	94.4444
U	27	81.2500
V	25	82.6389
W	15	89.5833
X	10	93.0556
Y	13	88.8889
Z	5	96.5278

TOTAL # MISTAKES= 407 TOTAL % CORRECT= 89.1293

CONFUSION MATRIX. ROWS ARE THE CORRECT LETTER AND THE COLUMNS ARE THE NUMBER OF TIMES THE LETTER WAS IDENTIFIED AS THE COLUMNATED LETTER

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
A	143	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B	0	127	0	0	2	0	0	9	0	1	1	0	2	0	0	1	0	1	0	0	0	0	0	0	0	0
C	0	0	131	4	1	0	5	0	0	0	0	0	0	0	0	2	0	1	0	0	0	0	0	0	0	0
D	0	8	0	129	0	0	0	2	0	0	0	0	0	0	0	3	0	0	0	0	0	2	0	0	0	0
E	0	1	1	0	135	2	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	1
F	0	0	0	0	0	140	0	0	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
G	0	4	9	1	0	0	117	0	0	0	0	0	0	0	6	0	3	1	2	0	0	0	1	0	0	0
H	0	7	0	1	0	1	0	124	0	2	0	0	2	0	0	0	0	6	0	0	1	0	0	0	0	0
I	0	2	0	0	1	0	0	0	122	2	1	1	0	0	0	0	0	0	0	13	0	0	5	0	0	0
J	0	0	0	0	0	1	0	1	9	116	0	0	0	0	6	0	1	0	0	11	5	0	0	0	0	0
K	0	1	0	0	2	0	0	2	1	0	128	1	0	1	0	0	0	2	0	0	0	0	1	4	0	1
L	0	0	0	0	0	0	0	0	5	0	0	139	0	0	0	0	0	0	0	0	0	0	0	0	0	0
M	0	2	0	0	0	0	0	2	1	0	0	0	135	1	0	0	0	1	0	0	0	1	0	1	0	0
N	0	0	0	0	0	0	0	3	0	0	0	0	12	126	0	0	0	0	0	0	0	1	1	1	0	0
O	0	4	2	7	0	0	0	3	0	0	0	0	0	0	116	0	12	0	0	0	0	0	0	0	0	0
P	0	0	0	3	0	2	0	0	0	0	0	0	0	0	0	139	0	0	0	0	0	0	0	0	0	0
Q	3	1	3	4	0	0	3	0	0	1	0	0	1	0	20	0	107	1	0	0	2	0	1	0	0	0
R	0	13	0	0	2	1	0	7	2	0	2	0	0	0	0	3	0	114	0	0	0	0	0	0	0	0
S	0	2	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	139	0	0	0	1	0	0	0
T	0	0	0	0	0	0	0	0	7	1	0	0	0	0	0	0	0	0	0	135	0	0	0	0	0	0
U	0	5	0	2	0	1	0	4	2	1	0	1	3	0	4	0	0	1	0	0	113	2	0	0	6	0
V	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	2	1	119	3	0	19	1
W	3	0	0	0	0	0	0	0	2	0	0	0	0	1	0	0	0	0	0	1	0	7	130	0	0	0
X	0	0	0	0	0	0	0	0	0	0	9	0	0	0	0	0	0	0	0	0	0	0	0	134	0	1
Y	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	3	0	7	2	0	126	3
Z	1	0	0	0	0	0	0	0	1	0	3	0	0	0	0	0	0	0	0	1	0	0	0	0	0	139

SUMMARY OF PROTOTYPE SET PERFORMANCE BY ALPHABET

ALPH	#MISTAKES	%CORRECT	ALPH	#MISTAKES	%CORRECT
1	3	88.4615	2	1	96.1538
3	0	100.0000	4	1	96.1538
5	1	96.1538	6	3	88.4615
7	0	84.6154	8	0	100.0000
9	0	100.0000	10	0	100.0000
11	2	92.3077	12	1	96.1538
13	2	92.3077	14	2	84.6154
15	1	96.1538	16	2	92.3077
17	1	96.1538	18	2	92.3077
19	1	96.1538	20	2	92.3077
21	1	96.1538	22	1	96.1538
23	0	100.0000	24	0	100.0000
25	0	100.0000	26	0	100.0000
27	0	100.0000	28	0	100.0000
29	1	96.1538	30	0	100.0000
31	0	100.0000	32	0	100.0000
33	1	96.1538	34	1	96.1538
35	3	88.4615	36	1	96.1538
37	1	96.1538	38	0	100.0000
39	3	88.4615	40	0	100.0000
41	2	92.3077	42	2	92.3077
43	2	92.3077	44	2	92.3077
45	2	92.3077	46	3	88.4615
47	0	100.0000	48	0	100.0000
49	0	100.0000	50	0	100.0000
51	1	96.1538	52	1	96.1538
53	3	88.4615	54	3	88.4615
55	7	73.0769	56	6	76.9231
57	3	88.4615	58	7	73.0769
59	1	96.1538	60	1	96.1538
61	2	92.3077	62	3	88.4615
63	2	92.3077	64	3	88.4615
65	2	92.3077	66	0	100.0000
67	4	84.6154	68	5	80.7692
69	2	92.3077	70	1	96.1538
71	3	88.4615	72	2	92.3077
73	1	96.1538	74	2	92.3077
75	0	100.0000	76	2	92.3077
77	4	84.6154	78	2	92.3077
79	3	88.4615	80	1	96.1538
81	0	100.0000	82	0	100.0000
84	2	92.3077	85	4	84.6154
86	3	88.4615	87	3	88.4615
89	0	100.0000	90	0	100.0000
91	2	92.3077	92	2	92.3077
93	5	80.7692	94	5	80.7692
95	4	84.6154	96	4	84.6154
97	3	88.4615	98	3	88.4615
99	4	84.6154	100	4	84.6154
102	0	100.0000	103	0	100.0000
104	4	84.6154	105	4	84.6154
107	0	100.0000	108	0	100.0000
109	1	96.1538	110	1	96.1538
111	1	96.1538	112	1	96.1538
113	3	88.4615	114	3	88.4615
115	0	100.0000	116	0	100.0000
117	3	88.4615	118	3	88.4615
119	4	84.6154	120	4	84.6154
121	6	76.9231	122	6	76.9231
123	1	96.1538	124	1	96.1538
125	3	88.4615	126	3	88.4615
127	1	96.1538	128	1	96.1538
129	2	92.3077	130	2	92.3077
131	1	96.1538	132	1	96.1538
133	0	100.0000	134	1	96.1538
135	1	96.1538	136	1	96.1538
137	1	96.1538	138	1	96.1538
139	0	100.0000	140	0	100.0000
141	1	96.1538	142	1	96.1538
143	0	100.0000	144	0	100.0000
145	2	92.3077	146	2	92.3077
147	3	88.4615	148	1	96.1538
149	0	100.0000	150	0	100.0000

TOTAL # MISTAKES= 266 TOTAL % CORRECT= 92.8953

RESULTS OF ISOLATED LETTERS COMPARISON
STANDARD OF COMPARISON IS EUCLIDEAN DISTANCE IN THE PROTOTYPE SPACE
RANGE OF COMPARISON IS ALPHABET 1 TO 150, LETTERS 1 TO 25
ALPHABETS EXCLUDED IN THE RANGE OF INTEREST ARE 24, 61, 82, 89,101,106,
PROTOTYPE FILE #1, LETTERS 1 TO 25
PROTOTYPE SPACE IS 49

SUMMARY OF PROTYPE SET PERFORMANCE BY LETTER

LETTER	#MISTAKES	%CORRECT
A	3	97.0107
B	12	91.6667
C	4	97.0202
D	11	92.7511
E	12	91.6667
F	3	98.6111
G	15	87.5000
H	15	88.5897
I	17	88.1166
J	5	96.6273
K	5	95.6278
L	2	98.6111
M	6	95.1389
N	3	97.7800
O	52	67.9889
P	5	95.8333
Q	12	91.6667
R	11	92.7511
S	4	100.0000
T	7	95.1389
U	15	88.0889
V	8	94.4444
W	20	86.1111
X	13	97.0556
Y	2	98.6111
Z	1	99.3056

TOTAL # MISTAKES= 266 TOTAL % CORRECT= 92.8953

CONFUSION MATRIX. ROWS ARE THE CORRECT LETTER AND THE COLUMNS ARE THE NUMBER OF TIMES THE LETTER WAS IDENTIFIED AS THE COLUMNATED LETTER

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
A	142	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B	2	125	0	0	0	0	0	8	0	1	2	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0
C	2	0	132	4	1	0	4	0	0	0	0	0	0	0	2	0	1	0	0	0	0	0	0	0	0	0
D	0	5	0	131	0	0	0	2	0	0	0	0	0	0	3	0	0	0	0	0	2	0	0	0	0	0
E	0	1	1	0	136	3	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
F	0	0	0	0	0	141	0	0	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
G	0	4	0	1	0	0	118	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
H	0	0	0	1	0	1	0	122	0	2	0	0	0	0	5	0	3	1	2	0	0	0	1	0	0	0
I	0	3	0	0	0	1	0	0	124	2	1	1	0	0	0	0	0	0	7	0	0	1	0	0	0	0
J	0	0	0	0	0	1	0	1	9	111	0	0	0	0	5	0	0	0	0	9	0	0	3	0	0	0
K	0	2	0	0	2	2	0	2	1	0	131	1	0	1	0	0	1	0	0	0	11	5	0	0	0	0
L	2	0	3	0	0	0	0	0	2	0	0	142	0	0	0	0	0	1	0	0	0	0	0	0	0	0
M	0	1	0	0	0	0	0	2	1	0	0	0	135	1	0	0	0	0	0	0	0	0	0	0	0	0
N	2	0	0	0	0	0	0	2	0	0	0	0	12	127	0	0	0	0	2	0	0	1	0	1	0	0
O	0	3	2	11	0	0	0	2	0	0	0	0	0	0	118	0	4	0	0	0	0	0	1	1	1	0
P	0	0	0	7	0	2	0	0	0	0	0	0	0	0	0	133	0	0	0	0	0	0	0	0	0	0
Q	0	1	1	5	0	0	3	0	0	1	0	0	1	0	21	0	105	1	0	0	0	0	0	0	0	0
R	0	13	0	0	2	1	0	7	2	0	4	0	0	0	0	3	0	115	0	0	0	2	0	1	0	0
S	0	1	0	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	140	0	0	0	0	0	0	0
T	2	0	0	0	0	0	0	0	7	1	0	0	0	0	0	0	0	0	0	140	0	0	1	0	0	0
U	2	5	0	2	0	2	0	4	1	1	0	1	3	0	4	0	0	1	0	0	135	0	0	0	0	0
V	3	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	117	2	0	0	0
W	3	0	0	0	0	0	0	0	2	0	0	0	0	1	0	0	0	0	0	1	0	1	119	3	0	19	1
X	0	0	0	0	0	0	0	0	0	0	3	0	0	0	0	0	0	0	0	0	0	0	8	120	0	0	0
Y	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	134	0	1
Z	1	0	0	0	1	0	0	0	1	0	2	0	0	0	0	0	0	0	0	0	0	0	2	0	128	3
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	139

APPENDIX M

Support and Utility Subroutines

```
66800=      SUBROUTINE INITC(NENT,LIST,IADD)
66900=C
67000=C**     THIS SUBMODULE INITIALIZES AN EMPTY INDEX
67100=C**     FOR THE CLAS VECTORS FILE. THIS FILE USES
67200=C**     A LINKED LIST OF USED AND UNUSED VECTORS
67300=C**     AS INDEX TO THE POSITION OF THE PROTOTYPE
67400=C**     DEFINITIONS  FOR EACH CLASS.
67500=C**
67600=C**     ENTRY POSITION WITHIN THIS LIST --> VECTOR
67700=C**     POSITION WITHIN THE FILE PLUS TWO
67800=C**
67900=C**     'NAME' ENTRY IN THE LIST IS A STRUCTURED KEY
68000=C**     WHERE NAME --> IC*1000+KTYP,  AND
68100=C**     IC=CLASS ID NUMBER, KTYP=VECTOR TYPE
68200=C**        KTYP=100 --> CAV, CLASS MEAN VECTOR
68300=C**             =201 --> CSDL, DEVIATION VECTOR (LEFT)
68400=C**             =202 --> CSDR,    "       "  (RIGHT)
68500=C**             =30N --> KER,  HUSK  VECTOR (N=1,MAXKE/JDX)
68600=C**
68700=C**     POINTER TO TOP OF UNUSED CHAIN IS LIST(1,1)
68800=C**     POINTER TO TOP OF USED VECTOR CHAIN IS LIST(1,2)
68900=C**     NAME INDEX ENTRY IS LIST(I,1)
69000=C**     LINK INDEX POINTER IS LIST(I,2)
69100=C**     VECTOR POSITION IS AT LIST(I,2) FOR THE VECTOR
69200=C**     WHOSE NAME APPEARS AT LIST(I,1)
69300=C**
69400=C**     NOTE THAT CHAINS ARE NOT CIRCULAR
69500=C**
69600=C**

69700=      DIMENSION LIST(NENT,2)
69800=C
69900=      NUM=IADD
70000=      IF(IADD.GT.0) GOTO 10
70100=      NUM=3
70200=      LIST(1,1)=3
70300=      LIST(1,2)=2
70400=      LIST(2,1)=99999999
70500=      LIST(2,2)=1
70600= 10   CONTINUE
70700=      DO 5 I=NUM,NENT
70800=      LIST(I,1)=0
70900=      LIST(I,2)=I+1
71000= 5    CONTINUE
71100=C
71200=C**      (1,1)=  3      (1,2)=    2
71300=C**      (2,1)=  999999 (2,2)=    1
71400=C**      (3,1)=  0      (3,2)=    4
71500=C**      (4,1)=  0      (4,2)=    5
71600=C**      (5,1)=  0      (5,2)=    6
71700=C
71800=C**                CLASS FILE HAS NENT + 1 VECTORS
71900=      RETURN
72000=      END
```

```
72100=
72200=
72300=      SUBROUTINE ADD(NAME,LIST,NENT,NPOS,IEOJ)
72400=      COMMON/TRACE/ILOG,ITRACE
72500=      DIMENSION LIST(NENT,2)
72600=      IEOJ=0
72700=C**           ADDS VECTORS TO THE CLAS  FILE
72800=      IU=LIST(1,1)
72900=      IF(IU.GT.NENT) GOTO 4
73000=      NAI=LIST(1,2)
73100=      LAI=1
73200=C**           IF NOT IN-BETWEEN
73300=  1   IF(LIST(NAI,1).LE.NAME)GOTO 2
73400=C**           ELSE INSERT NAME
73500=      LIST(IU,1)=NAME
73600=C**           UPDATE NEXT UNUSED POINTER
73700=      LIST(1,1)=LIST(IU,2)
73800=C**           SET NEW ENTRY PTR TO NEXT ENTRY(NAI)
73900=      LIST(IU,2)=LIST(LAI,2)
74000=C**           SET PREVIOUS ENTRY PTR TO NEW ENTRY
74100=      LIST(LAI,2)=IU
74200=      KAI=IU-2
74300=      GOTO 3
74400=  2   CONTINUE
74500=C**           FOLLOW POINTER CHAIN
74600=      LAI=NAI
74700=      NAI=LIST(NAI,2)
74800=      GOTO 1
74900=C**           SETUP NEW VECTOR NUMBER
75000=  3   CONTINUE
75100=      NPOS=KAI
75200=      GOTO 5
75300=C**           ERROR EXIT--> NO MORE ROOM
75400=  4   CONTINUE
75500=      IEOJ=2
75600=C**           NORMAL EXIT
75700=  5   CONTINUE
75800=      IF(ILOG.NE.0.AND.ILOG.LE.2)
75900=     1  WRITE(ITRACE,957)NPOS,IEOJ,LAI
76000= 957  FORMAT(7H  ADD  ,5I4)
76100=      RETURN
76200=      END
76300=
```

```
76400=
76500=      SUBROUTINE DEL(NAME,LIST,NENT,NPOS,IEOJ)
76600=C
76700=      COMMON/TRACE/ILOG,ITRACE
76800=      DIMENSION LIST(NENT,2)
76900=      IEOJ=0
77000=C**       DELETES VECTORS FROM CLAS FILE
77100=      IU=LIST(1,1)
77200=      NAI=LIST(1,2)
77300=      LAI=1
77400=C**         IF NOT YET IN BETWEEN
77500=  1   CONTINUE
77600=      IF(LIST(NAI,1).LT.NAME) GOTO 2
77700=      IF(LIST(NAI,1).GT.NAME) GOTO 4
77800=C**         LINK AROUND DELETED ENTRY
77900=      LIST(LAI,2)=LIST(NAI,2)
78000=C**         RESET NEXT UNUSED POINTER
78100=      LIST(1,1)=NAI
78200=C**         DELETE NAME ENTRY
78300=      LIST(NAI,1)=0
78400=C**       HOOK INTO UNUSED CHAIN
78500=      LIST(NAI,2)=IU
78600=      KAI=NAI-2
78700=      GOTO 3
78800=C**       FOLLOW THE NAME KEY CHAIN
78900=  2   CONTINUE
79000=      LAI=NAI
79100=      NAI=LIST(NAI,2)
79200=      GOTO 1
79300=C**       CLEAR THE RECORD
79400=C**       ASSUMED DONE BY CALLER
79500=  3   CONTINUE
79600=      NPOS=KAI
79700=      GOTO 5
79800=C**       RETURN NO SUCH NAME ERROR
79900=  4   CONTINUE
80000=      IEOJ=3
80100=C**           NORMAL EXIT
80200=  5   CONTINUE
80300=      IF(ILOG.NE.0.AND.ILOG.LE.2)
80400=     1  WRITE(ITRACE,957)NPOS,IEOJ,LAI,IU
80500= 957  FORMAT(7H  DEL  ,5I4)
80600=      RETURN
80700=      END
```

```
93800=
93900=      FUNCTION KERGET(CLAS,JDX,NEXC,IEOJ)
94000=      COMMON/KFILE/ MAXKV,NCIX,NENT,MSCAL,MZAPD
94100=      COMMON/TRACE/ILOG,ITRACE
94200=      INTEGER CLAS(JDX,1)
94300=      DATA LCLAS,NEXKE/0,0/
94400=      IEOJ=0
94500=      JD=JDX-3
94600=      IF(NEXC.EQ.LCLAS) GOTO 20
94700=C**      INITIALIZE ON FIRST CALL FOR CLASS
94800=      NAME=NEXC*1000+300
94900=      KV=0
95000=C**      OBTAIN NEW KERVEC POSITION
95100= 10   CONTINUE
95200=      KV=KV+1
95300=      IF(KV.GT.MAXKV) GOTO 30
95400=      NAME=NAME+1
95500=      NEXKE=0
95600=      CALL RIX(NAME,CLAS(1,NCIX),NENT,NPOS,IEOJ)
95700=      IF(IEOJ.GT.0) GOTO 40
95800=C**      BUMP ENTRY NUMBER AND GET KVEC
95900= 20   CONTINUE
96000=      NEXKE=NEXKE+1
96100=      IF(NEXKE.GT.JD) GOTO 10
96200=      KVEC=CLAS(NEXKE,NPOS)
96300=      KERGET=KVEC
96400=      IF(KVEC.GT.0) GOTO 50
96500=C**       GOT LAST ENTRY
96600= 30   CONTINUE
96700=      KERGET=-1
96800=      LCLAS=0
96900=      NEXKE=0
97000=      GOTO 50
97100=C**       NO SUCH KERVEC
97200= 40   CONTINUE
97300=      KERGET=0
97400=      IEOJ=5
97500=C**          EXIT ROUTINE
97600= 50   CONTINUE
97700=      IF(ILOG.NE.0.AND.ILOG.LE.2)
97800=     1   WRITE(ITRACE,957) NPOS,IEOJ
97900= 957  FORMAT(7H  KGET ,315)
98000=      RETURN
98100=      END
```

```
87800=
87900=      SUBROUTINE RIX(NAME,LIST,NENT,NPOS,IEOJ)
88000=      DIMENSION LIST(NENT,2)
88100=C**      RETURNS POSITION OF NAMED VECTOR
88200=      IEOJ=0
88300=      NAI=LIST(1,2)
88400=  1   IF(LIST(NAI,1).GT.NAME) GOTO 4
88500=      IF(LIST(NAI,1).LT.NAME) GOTO 2
88600=      NPOS=NAI-2
88700=      GOTO 5
88800=  2   NAI=LIST(NAI,2)
88900=      GOTO 1
89000=C**       NO SUCH VECTOR ERROR
89100=  4   CONTINUE
89200=      IEOJ=3
89300=C**       EXIT ROUTINE
89400=  5   CONTINUE
89500=      RETURN
89600=      END
89700=
```

```
13900=      SUBROUTINE INDEX(CLAS,JDX,ICX,IEOJ)
14000=      COMMON/CNUM/NEXC,NTC
14100=      COMMON/KFILE/MAXKV,NCIX,NENT,MSCAL,MZAPD
14200=      COMMON/NAMES/INPF,IOUTF,IFEAT,IOLDC,INEWC,ILOGF,IHISC
14300=      COMMON/TRACE/ILOG,ITRACE
14400=      COMMON/CONX/LCID(51,3),SCALE(2),ICONF,ANS(3),IMAC
14500=      DIMENSION CLAS(JDX,1)
14600=
14700=C**           SET UP POINTERS TABLE TO DRIVE PROGEAM
14800=C**           SAVES TIME ON MULTIPLE PASSES THRU CLAS
14900=      JD=JDX-3
15000=C**           FINDS MIN,MAX VECTORS
15100=      CALL RIX(98888,CLAS(1,NCIX),NENT,LCID(51,1),IEOJ)
15200=      IF(IEOJ.GT.0) GOTO 200
15300=      CALL RIX(99888,CLAS(1,NCIX),NENT,LCID(51,2),IEOJ)
15400=      IF(IEOJ.GT.0) GOTO 200
15500=      NCMIN=LCID(51,1)
15600=      NCMAX=LCID(51,2)
15700=C**           CLASS VECTORS
15800=      DO 100 N=1,NTC
15900=      NAMC=N*1000+100
16000=      NAML=N*1000+201
16100=      NAMR=N*1000+202
16200=      CALL RIX(NAMC,CLAS(1,NCIX),NENT,LCID(N,1),IEOJ)
16300=      IF(IEOJ.GT.0) GOTO 200
16400=      NCAV=LCID(N,1)
16500=      CALL RIX(NAML,CLAS(1,NCIX),NENT,LCID(N,2),IEOJ)
16600=      IF(IEOJ.GT.0) GOTO 200
16700=      NCSDL=LCID(N,2)
16800=      CALL RIX(NAMR,CLAS(1,NCIX),NENT,LCID(N,3),IEOJ)
16900=      IF(IEOJ.GT.0) LCID(N,3)=-1
17000=      NCSDR=LCID(N,3)
17100=      IF(SCALE(1).LE.1.0)GOTO 100
17200=C**           SCALE PROTOS ONCE AT START OF PROCESS
17300=      CALL XSCAL(CLAS(1,NCMIN),CLAS(1,NCMAX),CLAS(1,NCAV),JD,SCALE,1)
17400=      CALL XSCAL(CLAS(1,NCMIN),CLAS(1,NCMAX),CLAS(1,NCSDL),JD,SCALE,2)
17500=      IF(NCSDR.LT.0) GOTO 100
17600=      CALL XSCAL(CLAS(1,NCMIN),CLAS(1,NCMAX),CLAS(1,NCSDR),JD,SCALE,2)
17700= 100  CONTINUE
17800=C**           NORMAL EXIT
17900= 300  CONTINUE
18000=      IEOJ=0
18100=C**           ERROR EXIT
18200= 200  CONTINUE
18300=      IF(ILOG.NE.0)WRITE(ILOGF,900)
18400=     1 (N,(LCID(N,K),K=1,3),N=1,NTC)
18500= 900  FORMAT(17H  CLAS FILE INDEX/
18600=     1 ((( 6(I2,1H=,3I3,1H,) ))) )
18700=C**           TRACE-DUMP SCALED CLAS FILE
18800=      IF((SCALE(1).GT.1.0.AND.ILOG.EQ.3).OR.
18810=     1  (ILOG.EQ.1.OR.ILOG.EQ.2))
18900=     2 CALL PRCLAS(CLAS,CLAS,CLAS,JDX,ICX)
19000=      RETURN
19100=      END
```

```
119800=
119900=      SUBROUTINE PRCLAS(CLAK,LIST,CLAS,JDX)
120000=      COMMON/CNUM/NEXC,NTC
120100=      COMMON/NAMES/INPF,IOUTF,IFEAT,IOLDC,INEWC,ILOGF,IHISC
120200=      COMMON/KFILE/MAXKV,NCIX,NENT,MSCAL,MZAPD
120300=      INTEGER CLAK(JDX,1),LIST(1),CLAS(JDX,1),KTYP(4)
120400=      DATA KTYP(1),KTYP(2),KTYP(3),KTYP(4)/
120500=     1   4HCAV ,4HCSDL,4HCSDR,4HHUSK/
120600=C**               PRINT VECTOR TAGS
120700=      IX=NCIX-1
120800=      JD=JDX-3
120900=      J1=JDX-1
121000=      WRITE(ILOGF,900)(CLAK(JDX,I),CLAK(J1,I),I=1,IX)
121100= 900  FORMAT(17H CLAS VECTOR TAGS/(((  20(I4,1H/,I1) ))) )
121200=C**               PRINT CLAS FILE
121300=      DO 15 I=1,IX
121400=      IF(CLAK(JD+2,I).LE.2) GOTO 10
121500=C**               PRINT HUSKS
121600=C**      KTYP(--JD+2,  NEXC(--JDX,  NEXKE(--JD+1
121700=      WRITE(ILOGF,901)CLAK(JD+2,I),I,CLAK(JDX,I),CLAK(JD+1,I),
121800=     1  (CLAK(J,I),J=1,JD)
121900= 901  FORMAT(15H  HUSK VECTOR= ,I3,3H AT,I3,12H FOR CLASS  ,I3,
122000=     1    6H WITH ,I3,10H ENTRIES   /(((30I4)))  )
122100=      GOTO 15
122200=C**               PRINT VECTORS
122300= 10   CONTINUE
122400=      IP=CLAK(JD+2,I)+1
122500=      IF(CLAK(JDX,I).LE.0) GOTO 15
122600=      WRITE(ILOGF,903)I,KTYP(IP),CLAK(JD+3,I),(CLAS(J,I),J=1,JD)
122700= 903  FORMAT(2X,9H AT VEC= ,I4,1X,A4,7H CLAS= ,I3/
122800=     1   ((((10(1X,E9.3))))))
122900= 15   CONTINUE
123000=C**               PRINT LIST.IX
123100=      NLIS1=1+IX+JDX
123200=      NLIS2=NLIS1+NENT+20
123300=      DO 20 N=NLIS1,NLIS2,20
123400=      I1=N
123500=      I2=I1+20-1
123600=      N1=I1+NENT
123700=      N2=I2+NENT
123800=      NGO=N-NLIS1+1
123900=      WRITE(ILOGF,902)NGO,NENT,(LIST(I),I=I1,I2),(LIST(I),I=N1,N2)
124000=      IF(LIST(N).EQ.0) GOTO 21
124100= 902  FORMAT( 15H  LIST.IX FROM ,I6,4H OF ,I6/(((20I6/20I6/)))  )
124200= 20   CONTINUE
124300= 21   CONTINUE
124400=      RETURN
124500=      END
```

```
86900=
87000=      SUBROUTINE LOADC(CLAS,JDX,ICX,IEOJ)
87100=      COMMON/NAMES/INPF,IOUTF,IFEAT,IOLDC,INEWC,ILOGF,IHISC
87200=      DIMENSION CLAS(JDX,1)
87300=C**      LOAD CLAS FILES
87400=      READ(IOLDC)((CLAS(J,I),J=1,JDX),I=1,ICX)
87500=      RETURN
87600=      END
87700=
```

```
118500=
118600=
118700=      SUBROUTINE OPENH(LABLH,IEOJ)
118800=      COMMON/PARAM/MEMC,MEM,NHX,JDX,ICX,NC1,NL1,NB1,IC
118900=      COMMON/NAMES/INPF,IOUTF,IFEAT,IOLDC,INEWC,ILOGF,IHISC
119000=      COMMON/TRACE/ILOG,ITRACE
119100=      DIMENSION IDUM(6)
119200=      DATA NAMK/4HHISC/
119300=C***               OPENS HISTOGRAM FILES
119400=      REWIND IHISC
119500=      READ(IHISC)NAME,LABLH,NHX,JD,IDUM
119600=      NBUC=NHX-8
119700=      IF(ILOG.GT.0)WRITE(ILOGF,900)NAME,LABLH,NBUC,JD
119800=      WRITE(IOUTF,900)NAME,LABLH,NBUC,JD
119900= 900  FORMAT(21H  HISTOGRAM FILE OPEN,2X,A4,1X,I8,
120000=     1  1X,11H  BUCKETS= ,I3,6H DIMS=,I3)
120100=      JDX=JD+3
120200=      IF(NAME.NE.NAMK)IEOJ=2
120300=      RETURN
120400=      END
```

```
90900=
91000=      SUBROUTINE OPENX(IMOD,IE,LABLC,LABLF,IEOJ)
91100=      COMMON/CNUM/NEXC,NTC
91200=      COMMON/NAMES/INPF,IOUTF,IFEAT,IOLDC,INEWC,ILOGF,IHISC
91300=      COMMON/KFILE/MAXKV,NCIX,NENT,MSCAL,MZAPD
91400=      COMMON/PARAM/MEMC,MEM,NHX,JDX,ICX,NC1,NL1,NH1,IC
91500=      COMMON/CREC/LASV,IEOFR,NBPR,LB,MAXV
91600=      COMMON/DIST/FIRS,FLAS,FIRC,FLAC,NBUC,KTR,NI
91700=      COMMON/TRACE/ILOG,ITRACE
91800=      DATA IHI,IHC,IHF/1HI,1HC,1HF/
91900=C***     BYPASS FEATURE FILE IF NOT REQUIRED
92000=      IF(IE.GT.2) GOTO 35
92100=C**      OPEN FEAT FILE AND LOG ITS ID RECORD
92200=      REWIND IFEAT
92300=      READ(IFEAT)NAME,LABLF,JDX,LB,IC,MAXV,IOPT,
92400=     1  IHIS,FIRS,FLAS
92500=      WRITE(IOUTF,902) LABLF
92600= 902  FORMAT(12H  FEAT FILE ,I7)
92700=      IF(ILOG.NE.0)WRITE(ILOGF,903) NAME,LABLF,JDX,LB,IC,MAXV,IOPT,
92800=     1  IHIS,FIRS,FLAS
92900= 903  FORMAT(34H  OPENED FEATURE FILE WITH HEADER /
93000=     1  52H  NAME,LABL,JD, LB,IC, MV,IOPT,IHIS,   FIRS,   FLAS /
93100=     2  2X44,1XI4,1XI2,1XI3,1XI2,1XI3,4XI1,4XI1,2E8.2)
93200=      IF(IMOD.EQ.IHI)  GOTO 70
93300=C***     INITIALIZE OLD CLAS FILE IF REQUIRED
93400= 35   CONTINUE
93500=      REWIND IOLDC
93600=      READ(IOLDC)NAME,LABLC,MJDX,MICX,MTC,MBUC,MKV,MENT,
93700=     1  MCIX,MSCAL,MZAPD
93800=      WRITE(IOUTF,903) LABLC
93900= 903  FORMAT(12H  CLAS FILE ,I7)
94000=      IF(ILOG.NE.0)WRITE(ILOGF,901)LABLC,MJDX,MICX,MTC,MBUC,MKV,MENT,
94100=     1 MCIX,MSCAL,MZAPD
94200= 901  FORMAT(18H  CLAS FILE HEADER/
94300=     1  34H  CLAS    JDX    ICX    NTC     NB,
94400=     2  35H  MAXKV   NENT   NCIX   MSCAL MZAPD/
94500=     3   2X,10I7)
94600=C**      FILE LABLES: FEAT-->AANN,  CLAS-->AANNXX
94700=C***       IE IS SET TO 2,3,4,5 WHEN OLDC USED IN FORMAT
94800=      IF(IE.EQ.0.AND.LABLF.NE.LABLC/100000) GOTO 55
94900=      IF(IE.EQ.1.OR.IMOD.EQ.IHI) GOTO 70
95000=      JDX=MJDX
95100=      MAXKV=MKV
95200=      NBUC=MBUC
95300=      NCIX=MCIX
95400=      NENT=MENT
95500=      ICX=MICX
95600=      NTC=MTC
95700=      MEMC=JDX+ICX
95800=      GOTO 70
95900=C**      ERROR EXIT
96000= 55   CONTINUE
96100=      IEOJ=4
96200=C**      NORMAL EXIT
96300= 70   CONTINUE
96400=      RETURN
96500=      END
```

```
43100=
43200=
43300=      SUBROUTINE RFEAT(FEAT,JDX,INIT,IEOJ)
43400=      COMMON/NAMES/INPF,IOUTF,IFEAT,IOLDC,INEWC,ILOGF,IHISC
43500=      COMMON/TRACE/ILOG,ITRACE
43600=      COMMON/CNUM/NEXC,NTC
43700=      COMMON/CREC/  LASV,IEOFR,NBPR,LB,MAXV
43800=      INTEGER FEAT(JDX,1),IDUM(10)
43850=      IF(INIT.EQ.2) GOTO 6
43900=      IF(IGO.EQ.1) GOTO 1
44000=      NBPR=0
44100=      LASC=0
44200=      IGO=1
44300= 1    CONTINUE
44400=      IF(INIT.NE.1) GOTO 2
44500=C**         RESET FEAT FILE BEFORE FIRST BLOCK
44600=      REWIND IFEAT
44700=      NBPR=0
44800=      LASC=0
44900=      INIT=0
45000=      READ(IFEAT)IDUM
45100=      GOTO 30
45200=C**         READS NEXT FEAT BLOCK FOR OPTED CLASS=NEXC
45300=C**         FINDS FIRST FEAT BLOCK OF NEXT CLASS=NEXC
45400= 2    CONTINUE
45500=      IF(NEXC.NE.LASC.OR.LASC.EQ.0) GOTO 4
45600=C**    RESET TO REPEAT THIS CLASS
45700=      DO 3 N=1,NBPR
45800=      BACKSPACE IFEAT
45900= 3    CONTINUE
46000=      NBPR=0
46100=      LASC=0
46200=C**    PROCESS THIS CLASS
46300= 4    CONTINUE
46400=      IF(IEOFR.GT.0)NBPR=0
46500=      JD=JDX-3
46600=      IEOFR=0
46700=      LASV=LB
46800= 5    CONTINUE
46900=      READ(IFEAT)((FEAT(J,L),J=1,JDX),L=1,LB)
47000=C**         NEXC MUST MATCH BLOCK DATA
47100=      IF(IABS(FEAT(JD+2,1)).LT.NEXC) GOTO 5
47200=C**         COUNT BLOCKS FOR THIS CLAS
47300=      NBPR=NBPR+1
47400=      IF(IABS(FEAT(JD+2,1)).EQ.NEXC) GOTO 10
47500=C**         NO SUCH FEATURE CLASS
47600=      IEOJ=2
47700=      GOTO 30
47710=C**          READ TRAILER RECORD
47720= 6    CONTINUE
47730=      READ(IFEAT)((FEAT(J,L),J=1,JDX),L=1,2)
47740=      GOTO 30
47900=C**         EACH FEAT RECORD HAS JD+2 ENTRY NEC AT LAST VEC
47910= 10   CONTINUE
48000=      DO 20 L=1,LB
48100=      IF(FEAT(JD+2,L).GT.0) GOTO 20
48200=      LASV=L
48300=      LASC=NEXC
48400=      IEOFR=1
48500=      GOTO 30
48600= 20   CONTINUE
48700=C**                EXIT
48800= 30   CONTINUE
48900=      IF(ILOG.NE.0.AND.ILOG.LE.2)
49000=     1  WRITE(ITRACE,957)IEOFR,LASV,IEOJ,NBPR,LASC,NEXC
49100= 957  FORMAT(8H  RFEAT ,8I5)
49200=      RETURN
49300=      END
```

```
115300=
115400=
115500=      SUBROUTINE RHIST(HISC,NHX,IEOJ)
115600=      COMMON/NAMES/INPF,IOUTF,IFEAT,IOLDC,INEWC,ILOGF,IHISC
115650=      COMMON/CNUM/NEXC,NTC
115700=      DIMENSION HISC(NHX,1)
115800=      COMMON/TRACE/ILOG,ITRACE
115900=      COMMON/HREC/ISYM,KTR,LABLH,NI,NBUC,JD
116000=C***              READS AND LOGS HISTOGRAM RECORDS
116100= 5    CONTINUE
116200=      IF(NEXC.GT.NTC) GOTO 75
116300=      READ(IHISC)INEXC,ISYM,KTR,NI,NBUC,((HISC(N,J),N=1,NHX),J=1,JD)
116400=      IF(INEXC.LT.NEXC) GOTO 5
116500=      IF(INEXC.GT.NEXC) GOTO 80
116600=      NB=NBUC
116700=      IF(ISYM.EQ.2)NB=NI
116800=      IF(ILOG.GT.0)WRITE(ILOGF,900)NEXC,ISYM,NB
116900=      WRITE(IOUTF,900) NEXC,ISYM,NB
117000= 900  FORMAT(26H  LOADED HIST RECORD FOR  ,I3/
117100=     1   12H  WITH SYM= ,I1,13H  INTERVALS= ,I3)
117200=      GOTO 85
117300=C***              END OF FILE
117400=C***              DEFINE-OUTPUT ENDS AT NTC RECORDS
117500= 75   CONTINUE
117600=      IEOJ=-9
117700=      GOTO 85
117800=C***              ERROR EXIT
117900= 80   CONTINUE
118000=      IEOJ=1
118100=C***              NORMAL EXIT
118200= 85   CONTINUE
118250=      IF(ILOG.EQ.1)WRITE(ITRACE,957)NEXC,INEXC,NTC,ISYM,KTR,NI,NB,
118255=     1  NBUC,IEOJ
118260= 957  FORMAT(8H RHIST  ,9I4)
118300=      RETURN
118400=      END
```

```
37800=
37900=
38000=      SUBROUTINE XSCAL(FMIN,FMAX,VEC,NN,SCALE,IOP)
38100=      DIMENSION VEC(NN),SCALE(2),FMIN(NN),FMAX(NN)
38250=      E=1E-9
38300=C**            SCALES NEG TO POS RANGE
38400=      DO 10 N=1,NN
38500=      B=FMAX(N)
38600=      A=FMIN(N)
38700=      D=B-A+E
38800=      IF(IOP.EQ.1) VEC(N)=((VEC(N)-A)/D)*SCALE(1)
38900=      IF(IOP.EQ.2)VEC(N)=((VEC(N)/D))*SCALE(1)
39000=      VEC(N)=IFIX(VEC(N))
39100= 10   CONTINUE
39200=      RETURN
39300=      END
```

```
102600=      SUBROUTINE WRCLAS(CLAS,JDX,ICX,IS,LABLC)
102700=      COMMON/NAMES/INPF,IOUTF,IFEAT,IOLDC,INEWC,ILOGF,IHISC
102800=      COMMON/CNUM/NEXC,NTC
102900=      COMMON/KFILE/MAXKV,NCIX,NENT,MSCAL,MZAPD
103000=      COMMON/DIST/FIRS,FLAS,FIRC,FLAC,NBUC,KTR,NI
103100=      COMMON/TRACE/ILOG,ITRACE
103200=      INTEGER CLAS(JDX,1)
103300=      DATA NAME/4HCLAS/
103400=      IFIL=INEWC
103500=      IF(IS.EQ.1) IFIL=IOLDC
103600=C**       FILE  HEADER RECORD
103700=      WRITE(IFIL)NAME,LABLC,JDX,ICX,NTC,NBUC,MAXKV,NENT,NCIX
103800=     1  ,MSCAL,MZAPD
103810= 903  FORMAT(17H WROTE CLAS FILE /8H LABEL= ,I9,7H ,S/Z= ,2I3)
103820=      WRITE(IOUTF,903) LABLC,MSCAL,MZAPD
103900=      IF(ILOG.GE.2)WRITE(ILOGF,900)LABLC,JDX,ICX,NTC,
104000=     1  NBUC,MAXKV,NENT,NCIX,MSCAL,MZAPD
104100= 900  FORMAT(18H  NEWC FILE HEADER/
104200=     1 34H  CLAS    JDX    ICX    NTC     NB,
104300=     2 35H  MAXKV   NENT   NCIX   MSCAL MZAPD/
104400=     3  2X,10I7)
104500=C**       WHAT PACKING IS USEFUL?
104600=      WRITE(IFIL) ((CLAS(J,I),J=1,JDX),I=1,ICX)
104700=      JJ=JDX-1
104800=      II=NCIX-1
104900=      IF(ILOG.GE.2)WRITE(ILOGF,901)
105000=     1 ((CLAS(J,I),J=JJ,JDX),I=1,II)
105100= 901  FORMAT(17H NEWC VECTOR TAGS/((((12(1X,I4,1H/,I4)))) )
105200=      RETURN
105300=      END
```



```
114100=
114200=      SUBROUTINE WRHIS(HISC,NEXC,ISYM,LABLC,NBUC,KTR,NI,NHX,JDX)
114300=      COMMON/NAMES/INPF,IOUTF,IFEAT,IOLDC,INEWC,ILOGF,IHISC
114400=      COMMON/TRACE/ILOG,ITRACE
114500=      DIMENSION HISC(NHX,1),IDUM(6)
114600=      DATA NAME/4HHISC/
114700=      JD=JDX-3
114800=C**       WRITES HISC FILE WITH HISC REC FOR THIS CLASS
114900=      IF(ISW.EQ.1) GOTO 10
115000=      ISW=1
115100=      IF(ILOG.NE.0)WRITE(ILOGF,902) NAME,LABLC,NHX,JD
115200= 902  FORMAT(21H  HISTOGRAM FILE OPEN,2X,A4,I8,2I5)
115300=      WRITE(IHISC)NAME,LABLC,NHX,JD,IDUM
115400=C**      WRITE HISTOGRAM RECORD
115500= 10   CONTINUE
115600=C**     PRINT STATISTICS AND HISTOGRAM
115700=      IF(ILOG.NE.0)WRITE(ILOGF,900)NEXC,ISYM,KTR,NI,NBUC
115800= 900  FORMAT(12H HISC RECORD,5I6)
115900=      N1=1
116000=      N2=8
116100=      K1=9
116200=      K2=NHX
116300=      DO 100 I=1,ISYM
116400=      IF(ISYM.EQ.1) GOTO 20
116500=      IF(I.EQ.1) K2=KTR-1
116600=      IF(I.NE.2) GOTO 20
116700=      N1=KTR
116800=      N2=KTR+7
116900=      K1=KTR+8
117000=      K2=NHX
117100= 20   CONTINUE
117110=      IF(ILOG.NE.2) GOTO 100
117200=      DO 30 J=1,JD
117300=      WRITE(ILOGF,901)
117400=     1  J,(HISC(N,J),N=N1,N2),
117500=     2      (HISC(K,J),K=K1,K2)
117600= 901  FORMAT(38H COMPONENT  NUM        AV      CSDL   ,
117700=     * 2X4HCSDR,7X3HMIN,7X3HMAX,6X4HMODE,5X5HVMODE/
117800=     1  1XI2,2H= ,8(1XE9.3)/
117900=     2   ((( 20(1XF5.0)))) )
118000= 30   CONTINUE
118100= 100  CONTINUE
118200= 200  CONTINUE
118300=      WRITE(IHISC) NEXC,ISYM,KTR,NI,NBUC,((HISC(N,J),N=1,NHX),J=1,JD)
118400=      RETURN
118500=      END
```

```
58900=
59000=      SUBROUTINE STATX(LAS,FIN,FT,CAV,CSD,ISW)
59100=      DIMENSION FT(8)
59200=C
59300=C      COMPUTES STATISTICS (S-SET,U-USE,*-SET/USED,#-SCR)
59400=C      FT HOLDS SCRATCH VALUES AND EXTRA RESULTS
59500=C      FT(1),FT(2),FT(3),FT(4),FT(5),FT(6),FT(7),FT(8)
59600=C       NUM   AV    CSD   MBUC  MIN   MAX   MODE  VMOD
59700=C         S    #     #           *     *
59800=C      RESETS CAV ONLY WHEN ISW --> ZERO
59900=C
60000=C                INITIALIZE TEMPORARIES ON FIRST INPUT
60010=      IF(LAS.EQ.1) GOTO 25
60100=      IF(FT(1).NE.0.) GOTO 10
60200=      FT(5)=1E+10
60300=      FT(6)=-1E+10
60400=C**      CLEARS MBUC COUNTER FOR STATH
60500=      FT(4)=0.
60600= 10   CONTINUE
60700=C                COUNT NUMBER OF INPUT VALUES
60800=      FT(1)=FT(1)+1.
60900=C**              COMPUTE MIN,MAX
61000=      IF(FIN.LT.FT(5)) FT(5)=FIN
61100=      IF(FIN.GT.FT(6))FT(6)=FIN
61200=      IF(ISW.NE.0) GOTO 20
61300=C                COMPUTE MEAN
61400=      FT(3)=FT(3)+FIN
61600=C                COMPUTE STANDARD DEVIATION
61700=      FT(2)=FT(2)+FIN**2
62000=      GOTO 30
62100=C                 COMPUTE ASSYMETRIC DEVIATION
62200= 20   CONTINUE
62300=      FT(2)=FT(2)+(FIN-CAV)**2
62400=      GOTO 30
62410=C**               SET CAV, CSD ON LAST CALL
62420= 25   CONTINUE
62425=      IF(ISW.EQ.0) CAV=FT(3)/FT(1)
62430=      IF(ISW.EQ.0) T=ABS(FT(2)/FT(1) - CAV**2)
62450=      IF(ISW.EQ.1) T=FT(2)/FT(1)
62460=      IF(T.LT.1E-12) T=1E-12
62470=      CSD=SQRT(T)
62500=C**              NORMAL EXIT
62600= 30   CONTINUE
62700=      RETURN
62800=      END
```

```
63000=
63100=      SUBROUTINE STATH(LAS,FIN,FT,FHIS,NI,FIRS,FLAS)
63200=C        COMPUTES HISTOGRAM AND MODE VALUES
63300=C**     FIN-->INPUT VALUE   (VECTOR COMPONENT;SEE CDEFI)
63400=C**     NI-->NO. BUCKETS IN HISTOGRAM  (SEE CDEFI)
63500=C**     FHIS-->HISTOGRAM ARRAY
63600=C**     FIRS-->LOWER LIMIT OF VALUE RANGE
63700=C**     FLAS-->UPPER LIMIT OF VALUE RANGE
63800=C**     FT(1)-->COUNT OF INPUT VALUES  (SEE STATX)
63900=C**     FT(2)-->AVERAGE INPUT VALUE   (SEE STATX)
64000=C**     FT(3)-->VARIANCE     (SEE STATX)
64100=C**     FT(4) + 1  -->POINTS TO BUCKET CONTAINING MODE (MBUC)
64200=C**     FT(5)-->MINIMUM VALUE   (SEE STATX)
64300=C**     FT(6)-->MAXIMUM VALUE   (SEE STATX)
64400=C**     FT(7)-->MODE VALUE
64500=C**     FT(8)-->MODE COUNT AS PERCENT OF FT(1)
64600=      DIMENSION FT(8),FHIS(NI)
64610=      IF(LAS.EQ.1) GOTO 150
64700=      BEGIN=FIRS
64800=      STEP=(FLAS-FIRS)/FLOAT(NI)
64900=      DO 130 N=1,NI
65000=      K=N
65100=      BEGIN=BEGIN+STEP
65200=      IF(FIN.LT.BEGIN)GOTO 140
65300= 130  CONTINUE
65400= 140  CONTINUE
65500=      FHIS(K)=FHIS(K)+1.
65600=      LAST=FT(4)+1.
65700=C              FT(4) MUST BE INIT AT 0 FOR EACH STREAM
65800=C              FHIS(FT(4)) MUST BE LT FHIS(K) ON FIRST ENTRY
65900=      IF(FHIS(K).GT.FHIS(LAST)) FT(4)=K-1
66050=      GOTO 160
66070=C**             COMPUTE MODE AND PERCENTAGE ON LAST CALL
66075= 150   I=FT(4)+1.0
66100=      FT(7)=STEP*FLOAT(I-1)+STEP/2.+FIRS
66300=      FT(8)=FHIS(I)/FT(1)
66310=C**              NORMAL EXIT
66320= 160  CONTINUE
66400=      RETURN
66500=      END
```

```
63200=
63300=      SUBROUTINE PLOT3D(FCN2D,HID,IMAX,JMAX)
63400=      COMMON/PL3D/BSCALE,ZSCALE,ELEV,ICPS,IGRID
63500=      REAL HID,HLASTM,HLAST
63600=      DIMENSION FCN2D(IMAX,JMAX),HID(2)
63700=C
63800=C     ***COURTESY OF LT R ROACH, AFIT/ENY . ORIGINAL SOURCE
63900=C     ***FROM GEORGIA TECH OFFICE OF COMPUTING SERVICES, BY
64000=C     ***   MR RAY SPALDING....
64100=C     ***          MODIFIED FOR TEKTRONIX 4014 5/78 - JRL***
64200=C     MODULE PLOT3D PLOTS A THREE-DIMENSIONAL SURFACE AS
64300=C     A TWO-DIMENSIONAL PROJECTION ON A CALCOMP PLOTTER.
64400=C     THE SURFACE IS ROTATED 45 DEGREES IN I-J PLANE BEFORE
64500=C     PLOTTING.  ALL HIDDEN LINES ARE REMOVED.
64600=C      DEFINITIONS OF ARGUMENTS:
64700=C     BSCALE:  LINES PER INCH IN I AND J DIRECTIONS
64800=C     ZSCALE:  UNITS PER INCH IN Z DIRECTION
64900=C     ELEV:    ANGLE OF OBSERVER FROM HORIZONTAL, IN DEGREES
65000=C     FCN2D:   TWO-DIMENSIONALARRAY CONTAINING Z-VALUES
65100=C     IMAX:    FIRST DIMENSION OF FCN2D
65200=C     JMAX:    SECOND DIMENSION OF FCN2D
65300=C     IGRID:   3--> GRID,  2-->STRIPS
65400=C     HID MUST HAVE  (IMAX+JMAX)*2 FREE ENTRIES
65500=C     PLX REPLACES CALL TO CALCOMP 'PLOT'
65600=C     ..USE IT WITH CARE ELSEWHERE
65700=C     REVISE PLX, AND COMMENTED CALL TO 'WHERE' FOR CALCOMP
65800=C     TEKTRONIX VERSION ASSUMES CALL INITT IS EXTERNAL
65900=C
66000=      DATA ZCON/6HZ++++++/
66100=      IF(IGRID.NE.3) IGRID=2
66200=      IANDJ=IMAX+JMAX
66300=      RIANDJ=IANDJ-2
66400=      XSIZE=RIANDJ*.70710678119/BSCALE
66500=      YSIZE=XSIZE*SIN(ELEV*.0174532925l9)
66600=      ZSIZE=COS(ELEV*.0174532925l9)/ZSCALE
66700=      ZMIN=ZCON
66800=      DO 926 J=1,JMAX
66900=      DO 927 I=1,IMAX
67000=   10 ZMIN=AMIN1(ZMIN,FCN2D(I,J))
67100=  927 CONTINUE
67200=  926 CONTINUE
67300=C     CALL WHERE(XPAGE,YPAGE,HID)
67400=      CALL PLX(XPAGE,YPAGE,-3)
67500=      HID(1)=-30.*HID(1)
67600=      DO 928 KK=1,IGRID,2
67700=      K=KK-2
67800=      DO 929 I=2,IANDJ
67900=   20 HID(I)=HID(1)
68000=  929 CONTINUE
68100=      IF(K.EQ.1) GO TO 30
68200=      K1=1
68300=      K1F=IMAX
68400=      K2=1
68500=      K2F=JMAX
68600=      GO TO 40
68700=   30 K1=JMAX
68800=      K1F=1
68900=      K2=IMAX
69000=      K2F=1
69100=   40 K1I=K1
69200=      INC=-K
69300=   50 IPEN=3
69400=      IF(K.EQ.1) GO TO 60
69500=      J=K2
69600=      GO TO 70
69700=   60 I=K2
69800=   70 IF(K.EQ.1) GO TO 80
69900=      I=K1
70000=      GO TO 90
70100=   80 J=K1
70200=   90 XLAST=XPAGE
70300=      YLAST=YPAGE
70400=      HLAST=HLASTM
70500=      IPJ=I+J
70600=      HLASTM=HID(IPJ)
70700=      XPAGE=XSIZE*FLOAT(I+J-2)/RIANDJ
70800=      YPAGE=YSIZE*FLOAT(IMAX-I+J-1)/RIANDJ+ZSIZE*(FCN2D(I,J)-ZMIN)
70900=      IF(YPAGE-HLASTM) 100,120,120
71000=  100 IF(IPEN.EQ.3) GO TO 210
71100=      IF(INC.NE.K) GO TO 110
71200=      IGK=IMAX+JMAX+I+J
71300=      IF(HID(IGK)-YPAGE) 110,110,170
71400=  110 XINC=(XPAGE*(YLAST-HLAST)+XLAST*(HLASTM-YPAGE))/(HLASTM-HLAST-
71500=     1     YPAGE+YLAST)
71600=      YINC=(HLASTM*YLAST-HLAST*YPAGE)/(HLASTM-HLAST-YPAGE+YLAST)
71700=      GO TO 160
```

```
71800=  120  IPJ=I+J
71900=       HID(IPJ)=YPAGE
72000=       IF(IPEN.EQ.2) GO TO 190
72100=       IF(INC.EQ.K) GO TO 140
72200=       IF(K1.EQ.K1F) GO TO 210
72300=       IF(K1.EQ.K1I) GO TO 130
72400=       IGK=IMAX+JMAX+1
72500=       IF(YLAST-HID(IGK)) 130,150,150
72600=  130  CALL PLX(IPAGE,YPAGE,3)
72700=       IPEN=2
72800=       GO TO 210
72900=  140  IF(K1.EQ.K1I) GO TO 210
73000=       IF(K1.NE.K1I+INC) GO TO 150
73100=       XINC=XLAST
73200=       YINC=YLAST
73300=       GO TO 160
73400=  150  XINC=(XPAGE*(YLAST-HLAST)+XLAST*(HLASTM-YPAGE))/
73500=      1    (HLASTM-HLAST-YPAGE+YLAST)
73600=       YINC=(HLASTM*YLAST-HLAST*YPAGE)/(HLASTM-HLAST-YPAGE+YLAST)
73700=  160  CALL PLX(XINC,YINC,IPEN)
73800=  170  IPEN=5-IPEN
73900=       IF(IPEN.EQ.3) GO TO 210
74000=  190  CALL PLX(IPAGE,YPAGE,IPEN)
74100=  210  IGK=IMAX+JMAX+1
74200=       IGK1=IMAX+JMAX+I+J
74300=       HID(IGK)=HID(IGK1)
74400=       HID(IGK1)=YPAGE
74500=       IF(K1.EQ.K1F) GO TO 220
74600=       K1=K1+INC
74700=       GO TO 70
74800=  220  IF(K2.EQ.K2F) GO TO 230
74900=       K2=K2-K
75000=       K1F=K1I
75100=       K1I=K1
75200=       INC=-INC
75300=       GO TO 50
75400=  230  CALL PLX(0.0,0.0,-3)
75500=  928  CONTINUE
75600=       RETURN
75700=       END
75800=
```

```
75900=
76000=       SUBROUTINE PLX(X,Y,II)
76100=C***          EMULATES CALCOMP 'PLOT' ROUTINE
76200=C***          X,Y ARE INPUT AND OUTPUT VARIABLES.  NB!
76300=       IX=100*X
76400=       IY=100*Y
76500=       IF(II.EQ.2) CALL DRWABS(IX,IY)
76600=       IF(II.EQ.3) CALL MOVABS(IX,IY)
76700=       IF(II.NE.-3) RETURN
76800=       X=0.
76900=       Y=0.
77000=       RETURN
77100=       END
```

```
77600=      SUBROUTINE ILINE
77700=      COMMON/TRANS/LBYTES(16),IVAL,NVAL,LTYP,LADR(4),LIINE(43),IBYTE
77800=      DIMENSION KHEX(16)
77900=      DATA KHEX(1),KHEX(2),KHEX(3),KHEX(4),KHEX(5),
78000=     1  KHEX(6),KHEX(7),KHEX(8),KHEX(9),KHEX(10),KHEX(11),
78100=     2  KHEX(12),KHEX(13),KHEX(14),KHEX(15),KHEX(16)/
78200=     3  1H0,1H1,1H2,1H3,1H4,1H5,1H6,1H7,
78300=     4  1H8,1H9,1HA,1HB,1HC,1HD,1HE,1HF/
78400=      DATA KOLON/1H:/
78500=      DATA IHB/1H /
78600=      DATA JCRTN,JLINF/1H0,1H0/
78700=C**           @-->0D-->CR-->15(OCT),  0-->0A-->LF-->12(OCT)
78800=C**             SETUP START OF LINE (SOL)
78900=C***
79000=C***          TRANSFORMS LBYTES INTO A PAPERTAPE HEX-DUMP FORMAT
79100=C***          COMMON TO MANY 8080 MICROPROCESSOR SYSTEMS. EACH
79200=C***          DATA-BYTE IS REPRESENTED AS A 2-CHARACTER HEX VALUE
79300=C***          LINE FORMAT FOLLOWS:
79400=C***          :-->S.O.L.
79500=C***          NN-->A COUNT (IN HEX) OF DATA BYTES TO COME
79600=C***          AAAA-->ADDRESS OF FIRST BYTE  (HEX)
79700=C***          TT-->LINE TYPE(00=DATA LINE,01=EOR LINE)
79800=C***          HH-->HEX DATA BYTE VALUE
79900=C***          CC-->HEX CHECKSUM VALUE = FF-SUM(HH-VALUES)
80000=C***
80100=      LINE(1)=KOLON
80200=      LINE(2)=KHEX(2)
80300=      LINE(3)=KHEX(1)
80400=      DO 4 I=4,43
80500= 4    LINE(I)=IHB
80600=C**             SETUP PROCESS PER LTYP
80700=      IF(LTYP.GT.1) GOTO 7
80800=C**             INITIAL PROCESS
80900=      KADR=0
81000=      DO 6 I=1,4
81100=      DO 5 J=1,16
81200=      IF(LADR(I).NE.KHEX(J)) GOTO 5
81300=      LL=4-I
81400=      KADR=(J-1)*16**(LL)+KADR
81500=      GOTO 6
81600= 5    CONTINUE
81700= 6    CONTINUE
81800=C**             FOLLOW ON ADDRESS
81900= 7    CONTINUE
82000=      IF(LTYP.EQ.3) GOTO 700
82100=      IF(LTYP.EQ.2) KADR=KADR+16
82200=C**             DECODE CHAR ADDRESS TO HEX
82300=      K1=KADR/4096
82400=      KR=KADR-K1*4096
82500=      K2=KR/256
82600=      KR=KR-K2*256
82700=      K3=KR/16
82800=      K4=KR-K3*16
82900=      LINE(4)=KHEX(K1+1)
83000=      LINE(5)=KHEX(K2+1)
83100=      LINE(6)=KHEX(K3+1)
83200=      LINE(7)=KHEX(K4+1)
83300=      ICSUM=ICSUM+K1*16+K2   +K3*16+K4
83400=C**             LOOP ENCODES 16 BYTES
83500= 9    CONTINUE
83600=      LINE(8)=KHEX(1)
83700=      LINE(9)=KHEX(1)
83800=      DO 10 J=1,16
83900=      ICL=LBYTES(J)/16
84000=      ICR=LBYTES(J)-ICL*16
84100=      IF(ICL.GT.15.OR.ICR.GT.15) GOTO 750
84200=      L=2*J-1
84300=      LINE(L+9)=KHEX(ICL+1)
84400=      LINE(L+10)=KHEX(ICR+1)
84500= 10   ICSUM=ICSUM+ ICL*16+ICR
84600=C**             INSERT CHECK SUM INTO LINE
84700=      K1=ICSUM/256
84800=      KR=ICSUM-256*K1
84900=      KCL=15-KR/16
85000=      KCR=15-(KR-(KR/16)*16) + 1
85100=      LINE(42)=KHEX(KCL+1)
85200=      LINE(43)=KHEX(KCR+1)
85300=      GOTO 800
85400=C**             LAST LINE PROCESS
85500= 700  CONTINUE
85600=      DO 20 J=4,8
85700= 20   LINE(J)=KHEX(1)
85800=      LINE(9)=KHEX(2)
85900=      LINE(10)=KHEX(16)
86000=      LINE(11)=KHEX(16)
86100=      GOTO 800
86200=C**             ERROR EXIT
86300= 750  CONTINUE
86400=      IEOJ=1
86500=C**             NORMAL EXIT
86600= 800  CONTINUE
86700=      RETURN
86800=      END
```

```
95700=      SUBROUTINE FDSORT(FIN,ITAG,FOUT,NN)
95800=      DIMENSION FIN(NN),ITAG(NN),FOUT(NN)
95900=C**           SORT ROUTINE
96000=C**           SET UP ITAGS
96100=      DO 10 N=1,NN
96200=      ITAG(N)=N
96300= 10   FOUT(N)=FIN(N)
96400=C**           SORT FOUT AND ITAG
96500=      DO 20 N=1,NN
96600=      KN=N+1
96700=      DO 15 K=KN,NN
96800=      IF(FOUT(N).GT.FOUT(K)) GOTO 15
96900=      IT=ITAG(N)
97000=      ITAG(N)=ITAG(K)
97100=      ITAG(K)=IT
97200=      T=FOUT(N)
97300=      FOUT(N)=FOUT(K)
97400=      FOUT(K)=T
97500= 15   CONTINUE
97600= 20   CONTINUE
97700=      RETURN
97800=      END
```

```
94400=      SUBROUTINE IASORT(IN,NN)
94500=      DIMENSION IN(NN)
94600=      DO 20 N=1,NN
94700=      KN=N+1
94800=      DO 15 K=KN,NN
94900=      IF(IN(N).LT.IN(K))GOTO 15
95000=      IT=IN(N)
95100=      IN(N)=IN(K)
95200=      IN(K)=IT
95300= 15   CONTINUE
95400= 20   CONTINUE
95500=      RETURN
95600=      END
```

```
118700=
118800=     SUBROUTINE MARK(IX,IY,IDX,IDY)
118900=     IIX=IX+IDX
119000=     IIY=IY+IDY
119100=     CALL MOVABS(IIX,IIY)
119200=     IIX=IX-IDX
119300=     IIY=IY-IDY
119400=     CALL DRWABS(IIX,IIY)
119450=     CALL MOVABS(IX,IY)
119500=     RETURN
119600=     END
```

```
3820=
3830=       FUNCTION ENER(VEC,N)
3840=       DIMENSION VEC(N)
3850=       T=0.
3860=       DO 10 J=1,N
3870= 10    T=T+VEC(J)**2
3880=       ENER=T
3890=       RETURN
3900=       END
```

```
1870=       SUBROUTINE ERR(IOUTF,NN,IEOJ)
1880=       KK=KK+1
1890=       IF(KK.GT.NN) GOTO 10
1900=       WRITE(IOUTF,900)
1910= 900   FORMAT(2H ?)
1920=       RETURN
1930= 10    IEOJ=-IEOJ
1940=       RETURN
1950=       END
1960=
```

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER AFIT/GCS/EE/78-12	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) A DEVELOPMENT SYSTEM FOR MICROPROCESSOR BASED PATTERN RECOGNIZERS		5. TYPE OF REPORT & PERIOD COVERED MS Thesis 6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) John R. Leary Captain		8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS Air Force Institute of Technology (AFIT-EN) Wright Patterson AFB, Ohio 45433		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS		12. REPORT DATE December 1978 13. NUMBER OF PAGES 310
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)		15. SECURITY CLASS. (of this report) Unclassified 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

Approved for public release;distribution unlimited
IAW AFR 190-17

FEB 8 1979 JOSEPH P. HIPPS, Major, USAF
Director of Information

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Microprocessor programs
Pattern Recognition
Feature Selection

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

A tool for developing microprocessor based pattern recognizers is presented. A two segment system of programs is implemented. One segment is a subsystem consisting of a generalized pattern classifier program and utility routines for an INTEL SBC 80/20 microprocessor system. The other segment is a subsystem of four interactive programs. These four programs support feature selection, pattern class definition and performance evaluation using procedures fitted to the classifier algorithm. This subsystem operates on a

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

user supplied file of feature vectors. It produces a class defining structure for use by the classifier. It can use a TEKTRONIX 4014 for graphics support and will operate interactively within the CDC 6600 Intercom partition. Structured design, modular code, buffer allocation algorithms, and ANSI standard FORTRAN code make this segment transportable. The classifier segment requires an 8080 system. Less than 256 bytes of ROM are used. Data buffer locations and sizes, the number of classes and the number of features are specified by the user. Experiments produced estimates of classifier performance for this system. An error rate of less than ten percent is reported for one 26 class character recognition experiment.