7 9 - 1 0 1 6 7
CR 160116

DEPARTMENT OF MATHEMATICS
UNIVERSITY OF HOUSTON HOUSTON, TEXAS

FINAL REPORT
NAS-9-15543

MARCH 14,1978 - NOVEMBER 30,1978

(E79-10167) SMALL GRAINS SEPARATION AND OPTIMAL PASS SELECTION Final Report, 14 Mar. - 30 Nov. 1978 (Houston Univ.) 58 p HC A04/MF A01 CSCL 02C
N79-20438
Unclas
G3/43 00167

PREPARED FOR
EARTH OBSERVATION DIVISION, JSC
UNDER
CONTRACT NAS-9-15543

HOUSTON, TEXAS 77004

STAR Information Form

1. Report No.	2. Government Accession No.	3. Recipient's Catalog No.
FINAL		

4. Title and Subtitle Small Grains Separation and Optimal Pass Selection	5. Report Date 1/14/79 6. Performing Organization Code
7. Author(s) H.P. Decell, Jr., B.C. Peters, W.A. Coberly, W. Tally, M. Malek, R. Basu, J. Sparra	8. Performing Organization Report No. FINAL 10. Work Unit No.
9. Performing Organization Name and Address University of Houston Mathematics Department 3801 Cullen Blvd. Houston, Texas 77004	11. Contract or Grant No. NAS-9-15543 13. Type of Report and Period Covered March 14, 1978-Nov. 30, 1978
12. Sponsoring Agency Name and Address NASA/JSC Houston, TX 77058	14. Sponsoring Agency Code SF3/NASA/JSC

15. Supplementary Notes

16. Abstract

Separation of spring wheat (SW) from competing small grains (SG) has been the subject of several studies. In this report we approach the problem from an empirical point of view, using nonparametric discriminate function methods to investigate the feasibility of SW/SG separation. A limited set consisting of five segments with multiple acquisitions is used to illustrate the software and to make preliminary conclusions with respect to appropriate features.

An experimental design and feature selection program is developed for the purpose of selecting *a priori* statistically optimum subsets of LANDSAT acquisitions for analysis to separate wheat from nonwheat when given an adequate sample of labelled wheat and nonwheat LACIE segment pixel data.

A criterion for linear feature selection is proposed which is based on mean square approximation of class density functions. It is shown that for the widest possible class of approximants, the criterion reduces to Devijer's Bayesian distance. For linear approximants the criterion is equivalent to well known generalized Fisher criteria.

17. Key Words (Suggested by Author(s))	18. Distribution Statement
	Per Contract Requirements

19. Security Classif. (of this report)	20. Security Classif. (of this page)	21. No. of Pages	22. Price*
Unclassified		56	

*For sale by the National Technical Information Service, Springfield, Virginia 22161

Final Report

NAS-9-15543

March 14, 1978 - November 30, 1978

Prepared for:

Earth Observations Division

Johnson Spacecraft Center

Houston, Texas

Henry P. Decell, Jr., Principal Investigator
B. Charles Peters , Research Associate
William A. Coberly , Consultant
William Tally, Programmer
Massoud Malek , Research Assistant
Rehka Basu , Research Assistant
Jim Sparra , Research Assistant

SMALL GRAINS FEATURE SELECTION

INTRODUCTION

Separation of spring wheat (SW) from competing small grains (SG) hrs been the subject of several studies. In this project we wish to approach the problem from an emperical point of view, using nonparametric discriminate function methods to investigate the feasibility of SW/SG separation. A limited data set consisting of five segments with multiple acquisitions was used to illustrate the software and to make preliminary conclusions with respect to appropriate features. The data sets used are given in Table 1.

Segment	Acquisition data in 1976						
	1	2	3	4	5	6	7
1614	130			183	201	219	
1618	127		163		199		235
1624	128	146					236
1642	127	145	163	182	199		236
1645	127	145	164	181			235

Table 1. Acquisitions use in experiment.

LINEAR DISCRIMINATE FUNCTIONS

Several methods including Principal Components, Fischer's linear discriminate function, minimization of the Perceptron criteria function, and Minimum Squared Error (MSE) procedures were considered and tested. The MSE procedure using the Ho-Kashyap algorithm was chosen as the most consistent, based on limited testing. A brief discription of the procedure follows.

Let $X_i = (x_{i1}, x_{i2}, \ldots, x_{ip})^T$ be the measurement vector for the ith prototype. Define

$$Y_i = \begin{cases} (1, X_i^T) & \text{for } X_i \text{ in class 1} \\ (-1, -X_i^T) & \text{for } X_i \text{ in class 2} \end{cases}$$

The problem is to select a $(p+1)$ vector α, such that

$$Y_i^T \alpha > 0 \qquad \text{for all } i = 1, \ldots, n.$$

Or, failing that, minimize the number of errors. A more tractable problem is the MSE criteria which is defined as follows.

Let

$$A = \begin{bmatrix} Y_1^T \\ Y_2^T \\ \vdots \\ Y_n^T \end{bmatrix}_{n \times (p+1)}$$

Now find a solution (if possible) to the equation

$$A\,\alpha = \beta > 0.$$

Or, minimize $||A\alpha - \beta||$ over all α and β. For each fixed β, the minimum norm solution is given by

$$\alpha = A^{\#}\,\beta,$$

where $A^{\#}$ is the pseudo-inverse of A. So the difficulty is one of determining the appropriate β. The Ho-Kashyap algorithm solves this problem in the following way. Define the initial values of two vector sequences α and β by

$$\beta(0) = (1, \ldots, 1)^T$$
$$\alpha(0) = A^{\#}\,\beta(0).$$

For $k = 1,2,\ldots$ define

$$e(k) = A\,\alpha(k) - \beta(k)$$
$$e^{+}(k) = \frac{1}{2}(\, e(k) + |e(k)|\,).$$

and

$$\beta(k+1) = \beta(k) + \rho e^{+}(k)$$
$$\alpha(k+1) = \alpha(k) + \rho A^{\#} e^{+}(k)$$

where $\rho > 0$. If the prototypes are separable, then $||e(k)||$ converges to 0. If not, then $||e(k)||$ converges to $c > 0$.

THE EXPERIMENT

In this test we have used the Ho-Kashyap algorithm to determine three discriminant functions for each of five segments over various pass combinations. The three discriminant functions are defined as follows.

4-CH The original four channel LANDSAT measurements are used for each acquisition.

L(B,G) The Tassel Cap coordinates B and G are computed for each acquisition.

Q(B,G) The Tassel Cap coordinates B and G plus the quadratic terms B^2, G^2, and BG are computed for each acquisition. In Tables 2-6 the error rates are given for each of the above discriminant functions, calculated by the Ho-Kashyap algorithm.

SEGMENT 1614

ERRORS (SW/SG)

ACQ.	4-CH	L(B,G)	Q(B,G)
1	1/5	1/11	1/11
4	3/7	2/7	1/7
5	2/5	1/6	0/6
6	2/3	0/6	0/4
1,4	1/4	1/6	1/6
1,5	0/0	1/5	0/5
1,6	0/1	2/5	0/4
4,5	3/4	1/5	2/5
4,6	2/3	1/4	0/2
5,6	1/3	0/4	0/1
1,4,5	0/0	1/5	1/2
1,4,6	0/0	0/3	0/2
1,5,6	0/0	0/5	0/2
4,5,6	2/3	1/5	0/1
1,4,5,6	0/0	2/4	0/0

Labeled Dot Distribution: SW-31 / SG-13

Acquisition Dates:

1	-	76130
4	-	76183
5	-	76201
6	-	76219

Table 2.

SEGMENT 1618

ERRORS (SW/SG)

ACQ.	4-CH	L(B,G)	Q(B,G)
1	0/22	0/22	2/21
3	3/21	2/21	4/14
5	3/10	3/15	3/4
7	4/9	2/13	2/13
1,3	2/20	2/20	6/13
1,5	3/10	3/15	2/3
1,7	5/9	2/12	4/11
3,5	3/10	3/15	1/4
3,7	4/7	1/13	2/7
5,7	3/8	3/13	1/5
1,3,5	3/9	3/13	1/3
1,3,7	5/7	4/11	5/5
1,5,7	4/5	3/9	0/2
3,5,7	3/4	3/12	1/2
1,3,5,7	3/3	3/9	0/2

Labeled Dot Distribution: SW-60 / SG-22

Acquisition Dates:
1 - 76127
3 - 76163
5 - 76199
7 - 76235

Table 3.

SEGMENT 1624

ERRORS (SW/SG)

ACQ.	4-CH	L(B,G)	Q(B,G)
1	4/18	0/22	1/20
2	0/22	0/21	1/21
7	5/16	5/16	5/13
1,2	3/16	0/21	1/19
1,7	6/11	5/14	4/13
2,7	5/13	5/13	5/13
1,2,7	4/12	4/14	4/11

Labeled Dot Distribution:	SW-62 / SG-22		
Acquisition Dates:	1	-	76128
	2	-	76146
	7	-	76236

Table 4.

SEGMENT 1642

ERRORS (SW/SG)

ACQ.	4-CH	L(B,G)	Q(B,G)
1	0/18	0/19	0/18
2	2/14	2/18	1/13
3	3/16	2/15	3/12
4	0/19	0/19	0/16
5	0/18	0/19	3/16
7	1/19	1/18	0/18
1,2	3/14	1/18	1/12
1,3	2/9	2/12	4/11
1,4	1/17	0/19	0/16
1,5	3/16	2/18	4/9
1,7	1/17	1/17	1/16
2,3	3/11	3/12	1/9
2,4	3/18	2/18	2/11
2,5	2/14	2/18	5/10
2,7	4/15	1/18	1/10
3,4	3/13	3/14	3/10
3,5	5/10	3/14	4/8
3,7	4/13	4/14	2/8
4,5	2/16	0/18	2/19
4,7	1/19	1/18	0/15
5,7	0/17	0/18	3/14
1,2,3,4,5,7	7/7	4/9	0/0

Labeled Dot Distribution: SW-58 / SG-19

Acquisition Dates:

1 - 76127
2 - 76145
3 - 76163
4 - 76182
5 - 76199
7 - 76236

Table 5.

SEGMENT 1645

ERRORS (SW/SG)

ACQ.	4-CH	L(B,G)	Q(B,G)
1	0/20	0/20	0/20
2	0/20	0/20	0/20
3	1/20	0/20	0/19
4	1/19	0/20	0/20
7	0/19	0/20	2/16
1,2	0/19	0/20	0/20
1,3	0/18	0/20	1/19
1,4	0/17	0/20	0/20
1,7	1/12	2/17	2/11
2,3	0/20	0/20	0/19
2,4	1/18	0/20	0/20
2,7	0/18	0/20	3/11
3,4	4/16	2/20	1/18
3,7	2/16	1/17	3/11
4,7	3/10	3/13	3/10
1,2,3,4,7	3/10	6/11	2/6

Labeled Dot Distribution:	SW-75 / SG-20		
Acquisition Dates:	1	-	76127
	2	-	76145
	3	-	76164
	4	-	76181
	7	-	76235

Table 6.

CONCLUSIONS

The first observation must be, that, in these segments, the separation of SW from SG is not an easy task. A reasonable error rate was achieved with one pass only in segment 1614. Reasonable two pass error rates were achieved in segments 1614 and 1618, and to a lesser degree in 1642. Segment 1624 had inadequate acquisitions and segment 1645 gave poor results even when all acquisitions were used. Segment 1645 and 1642 did not have acquisitions in windows 5 or 6. These two windows provided the best results in the other three segments in single pass or two pass combinations.

The other observation is that generally the Q(B,G) features provided as good or better separation as did the 4-CH features. The L(B,G) features did not compete as well. The advantage of the Q(B,G) features is that the two dimensional quadratic discriminant function can be plotted on a per pass basis, making the prospect of generating graphical AI aids a possibility.

In conclusion, it appears that windows 5 and 6 play an important role in the SW/SG separation problem. (The corresponding crop indices should be determined for this strata.) In addition we recommend further testing of the Q(B,G) features over a broad range of spring wheat blind sites.

APPENDIX

UHLDF Program

```
C     ..............................................................
C                 ****  PROGRAM  UHLDF  ****
C
C        THIS PROGRAM COMPUTES THE 2-CLASS NONPARAMETRIC
C        DISCRIMINATE FUNCTION.  THE HO-KASHYAP ALGORITHM
C        IS IMPLIMENTED.  .
C
C     ..............................................................
C
      DIMENSION D(4000),DINV(4000),A(40),ICLASS(500),B(500),Y(500)
      DIMENSION Z(500),C(500),IQ(500),V(40),E(500)
      DIMENSION U(40,40),AFLAG(40),ATEMP(40),W(2500)
      EQUIVALENCE  (U(1,1),B(1))    ,    (AFLAG(1),Z(1))
     *              ,(ATEMP(1),Y(51)),    (W(1),B(1))
      DATA YES/'Y'/
   66 WRITE(108,6600)
      READ(105,6666) YESS
 6600 FORMAT(' AGAIN?')
 6666 FORMAT(A1)
      IF(YESS.NE.YES) STOP
C
C          GET DATA ARRAY
C          D      -  NSAMP BY NV ARRAY CONTAINING PROTOTYPES AS ROWS.
C          DINV   -  TRANSPOSE OF THE PSEUDO-INVERSE OF D.  (NSAMP BY NV)
C
      REWIND 1
      CALL GETD(DINV,NSAMP,NV,ICLASS)
      IF(NSAMP.EQ.0) GO TO 500
      NSIZE=NSAMP*NV
      OUTPUT,NSIZE
      CALL TRANS(DINV,D,NV,NSAMP)
      CALL MOVE(D,DINV,NSIZE)
C
C
C
C
C
      MAXITR=100
C
C           COMPUTE PSEUDO-INVERSE OF D. (TRANSPOSE)
C           INITIALIZE B-VECTOR
C           INITIALIZE A-VECTOR. (LINEAR DISCRIMINATE FUNCTION)
C           COMPUTE MISSCLASSIFICATIONS FOR INITIAL DF.
C
      CALL GINV2M(DINV,NSAMP,NV,NSAMP,NV,KZ,U,AFLAG,ATEMP,1.E-12)
      CALL FILL(B,NSAMP,1.)
      CALL TPRD(DINV,B,A,NSAMP,NV,0,0,1)
      WRITE(108,8800)  (A(J),J=1,NV)
      CALL MPRD(D,A,Y,NSAMP,NV,0,0,1)
      CALL MISSCL(Y,NSAMP,MISS1,MISS2,ICLASS)
```

```
      MISS=MISS1+MISS2
      OUTPUT MISS1,MISS2,MISS
C
C                   CALL HO-KASHYAP ALGORITHM.
C
      CALL HOKASH(D,DINV,A,B,Y,E,NV,NSAMP,MAXITR,ICLASS)
 8800 FORMAT(2X,5F10.4)
      GO TO 66
      END
C
C                COPY ARRAY X INTO ARRAY Y
C
C
      SUBROUTINE MOVE(X,Y,N)
      DIMENSION X(1),Y(1)
      DO 1 I=1,N
    1 Y(I)=X(I)
      RETURN
      END
C
C                INITIALIZE N LOCATIONS IN ARRAY X WITH VALUE C
C
C
      SUBROUTINE FILL(X,N,C)
      DIMENSION X(1)
      DO 1 I=1,N
    1 X(I)=C
      RETURN
      END
C
C                MOVE TRANSPOSE OF MATRIX A INTO B.
C
C
C
      SUBROUTINE TRANS(A,B,N,M)
      DIMENSION A(N,M),B(M,N)
      DO 1 I=1,N
      DO 1 J=1,M
    1 B(J,I)=A(I,J)
      RETURN
      END
```

```
C     ..................................................          HOKA0000
C                                                                  HOKA0010
C          HO-KASHYAP ALGORITHM.                                   HOKA0020
C                                                                  HOKA0030
C     ..................................................          HOKA0040
C                                                                  HOKA0050
      SUBROUTINE HOKASH(D,DINV,A,B,Y,E,NV,NSAMP,MAXITR,ICLASS)     HOKA0060
      DIMENSION D(1),DINV(1),A(1),B(1),Y(1),E(1),ICLASS(1)         HOKA0070
      ITR=0                                                        HOKA0080
   10 ITR=ITR+1                                                    HOKA0090
      CALL MPRD(D,A,Y,NSAMP,NV,0,0,1)                              HOKA0100
      CALL MSUB(Y,B,E,NSAMP,1,0,0)                                 HOKA0110
      CALL MISSCL(Y,NSAMP,MISS1,MISS2,ICLASS)                      HOKA0120
      IF(ITR.GT.MAXITR) GO TO 200                                  HOKA0130
      CALL TEST(E,NSAMP,KEY)                                       HOKA0140
      IF(KEY) 200,100,100                                          HOKA0150
C                                                                  HOKA0160
C          ONE MORE TIME                                           HOKA0170
C                                                                  HOKA0180
  100 CALL POS(E,NSAMP)                                            HOKA0190
      CALL MADD(B,E,B,NSAMP,1,0,0)                                 HOKA0200
      CALL   TPRD(DINV,E,Y,NSAMP,NV,0,0,1)                         HOKA0210
      CALL MADD(A,Y,A,NV,1,0,0)                                    HOKA0220
      GO TO 10                                                     HOKA0230
C                                                                  HOKA0240
C          TERMINATE                                               HOKA0250
C                                                                  HOKA0260
  200 CONTINUE                                                     HOKA0270
      OUTPUT, ITR,MISS1,MISS2                                      HOKA0280
      WRITE(108,8800) (A(J),J=1,NV)                                HOKA0290
 8800 FORMAT(2X,5F10.4)                                            HOKA0300
      RETURN                                                       HOKA0310
      END                                                          HOKA0320
C                                                                  HOKA0330
C                                                                  HOKA0340
C                                                                  HOKA0350
C                                                                  HOKA0360
C                                                                  HOKA0370
      SUBROUTINE TEST(Y,N,KEY)                                     HOKA0380
      DIMENSION Y(N)                                               HOKA0390
      KEY=0                                                        HOKA0400
      LN=0                                                         HOKA0410
      LP=0                                                         HOKA0420
      CALL NORM(Y,N,YNORM)                                         HOKA0430
      DO 10 I=1,N                                                  HOKA0440
      IF(Y(I)/YNORM-1.E-50) 1,1,2                                  HOKA0450
    1 LN=1                                                         HOKA0460
      GO TO 3                                                      HOKA0470
    2 LP=1                                                         HOKA0480
    3 IF(LN.EQ.1.AND.LP.EQ.1) RETURN                               HOKA0490
```

```
   10 CONTINUE
      KEY=1
      IF(LP.EQ.0) KEY=-1
      RETURN
      END
C
C
C
C
C
      SUBROUTINE POS(Y,N)
      DIMENSION Y(N)
      DO 10 I=1,N
   10 Y(I)=(Y(I)+ABS(Y(I)))/2.
      RETURN
      END
C
C
C
C
C
      SUBROUTINE MISSCL(Y,NSAMP,MISS1,MISS2,ICLASS)
      DIMENSION Y(1),ICLASS(1)
      MISS1=0
      MISS2=0
      DO 30 I=1,NSAMP
      IF(Y(I).GE.0.) GO TO 30
      IF(ICLASS(I).EQ.1) MISS1=MISS1+1
      IF(ICLASS(I).EQ.2) MISS2=MISS2+1
   30 CONTINUE
      RETURN
      END
C
C
C
C
C
      SUBROUTINE NORM(Y,N,YNORM)
      DIMENSION Y(1)
      YNORM=0.
      DO 10 I=1,N
   10 YNORM=YNORM+Y(I)*Y(I)
      YNORM=SQRT(YNORM)
      RETURN
      END
```

```
C     .....................................................
C
C                   ***   4 CHANNEL LANDSAT   ***
C
C      THIS VERSION OF GETD USES THE 4 LANDSAT CHANNELS AS
C      FEATURES. (UP TO 7 PASSES)
C
C      D   -   THIS IS THE DATA ARRAY WHICH IS RETURNED TO THE
C              CALLING ROUTINE.  THE DIMENSION IS  NV  BY NSAMP.
C
C              THE COLUMN VECTORS OF  D  CONSIST OF
C
C                   ( 1. , X1, X2, ... , XN)
C
C               IF THE PROTOTYPE IS IN CLASS 1 AND THE NEGATIVE
C               OF THE ABOVE VECTOR IF THE PROTOTYPE IS IN
C               CLASS 2.
C
C       NSAMP-  THE NUMBER OF PROTOTYPES.  THIS VALUE IS RETURNED TO
C               THE CALLING PROGRAM.
C
C       NV   -  THE NUMBER OF VARIABLES PLUS ONE FOR THE
C               CONSTANT TERM. THIS VALUE IS RETURNED TO THE CALLING
C               PROGRAM.
C
C       ICLASS- THIS IS A VECTOR, RETURNED TO THE CALLING PROGRAM WHICH
C              IDENTIFIES (1 OR 2) THE CLASS ASSIGNMENT OF EACH
C              PROTOTYPE.
C
C      ...........................................................
C
      SUBROUTINE GETD(D,NSAMP,NV,ICLASS)
      DIMENSION D(1),ICLASS(1),X(28),IP(7)
      DATA IBLK,IO,IW/' ','O','W'/
   88 WRITE(108,4000)
 4000 FORMAT(' INPUT NUMBER OF PASSES. 1-7')
      READ(105,3000,ERR=88) NPASS
      OUTPUT NPASS
      WRITE(108,2000)
 2000 FORMAT(' INPUT PASS NO.  1-7')
      READ(105,3000,ERR=88) (IP(KK),KK=1,NPASS)
      WRITE(108,2500) (IP(KK),KK=1,NPASS)
 2500 FORMAT(7X,7I3)
 3000 FORMAT(7I1)
      NSAMP=0
      N1=0
      N2=0
      IND=0
      NV=NPASS*4+1
      DO 100 K=1,209
```

```
C                                                                       GD010500
C              READ LABELED DOTS (UNDER FORMAT 1000)                    GD010510
C                                                                       GD010520
      READ(1,1000,END=999) IG1,IG2,X                                    GD010530
 1000 FORMAT(2A1,8F5.1,/,2X,8F5.1,/,2X,8F5.1,/,2X,4F5.1)                GD010540
      IF(IG1.NE.IBLK) GO TO 100                                         GD010550
      IF(IG2.EQ.IBLK.OR.IG2.EQ.IO) GO TO 100                            GD010560
      DO 10 KK=1,NPASS                                                  GD010570
      I1=(IP(KK)-1)*4+1                                                 GD010580
      IF(X(I1).GT.98.5) GO TO 100                                       GD010590
   10 CONTINUE                                                          GD010600
      NSAMP=NSAMP+1                                                     GD010610
      ICLASS(NSAMP)=1                                                   GD010620
      IF(IG2.NE.IW) ICLASS(NSAMP)=2                                     GD010630
      IND=IND+1                                                         GD010640
      D(IND)=1.                                                         GD010650
      IF(ICLASS(NSAMP).EQ.2) D(IND)=-1.                                 GD010660
      DO 30 NPP=1,NPASS                                                 GD010670
      I1=(IP(NPP)-1)*4+1                                                GD010680
      I2=I1+3                                                           GD010690
      DO 20 J=I1,I2                                                     GD010700
      IND=IND+1                                                         GD010710
      D(IND)=X(J)                                                       GD010720
      IF(ICLASS(NSAMP).EQ.2) D(IND)=-D(IND)                             GD010730
   20 CONTINUE                                                          GD010740
   30 CONTINUE                                                          GD010750
      IF(ICLASS(NSAMP).EQ.1) N1=N1+1                                    GD010760
      IF(ICLASS(NSAMP).EQ.2) N2=N2+1                                    GD010770
  100 CONTINUE                                                          GD010780
  999 CONTINUE                                                          GD010790
      OUTPUT N1,N2,NSAMP                                                GD010800
      RETURN                                                            GD010810
      END                                                               GD010820
```

```
C   ..........................................................
C
C                   ***   B,G   ***
C
C      THIS VERSION OF GETD USES THE FIRST 2 TASSEL CAP VARIABLES (B,G)
C      FEATURES. (UP TO 7 PASSES)
C
C      D   -   THIS IS THE DATA ARRAY WHICH IS RETURNED TO THE
C              CALLING ROUTINE.  THE DIMENSION IS  NV  BY NSAMP.
C
C              THE COLUMN VECTORS OF  D  CONSIST OF
C
C                   ( 1. , X1, X2, ... , XN)
C
C               IF THE PROTOTYPE IS IN CLASS 1 AND THE NEGATIVE
C               OF THE ABOVE VECTOR IF THE PROTOTYPE IS IN
C               CLASS 2.
C
C       NSAMP-  THE NUMBER OF PROTOTYPES.  THIS VALUE IS RETURNED TO
C               THE CALLING PROGRAM.
C
C       NV   -  THE NUMBER OF VARIABLES PLUS ONE FOR THE
C               CONSTANT TERM. THIS VALUE IS RETURNED TO THE CALLING
C               PROGRAM.
C
C       ICLASS- THIS IS A VECTOR, RETURNED TO THE CALLING PROGRAM WHICH
C              IDENTIFIES (1 OR 2) THE CLASS ASSIGNMENT OF EACH
C              PROTOTYPE.
C
C    ..............................................................
C
      SUBROUTINE GETD(D,NSAMP,NV,ICLASS)
      DIMENSION D(1),ICLASS(1),X(28),IP(7)
      DATA IBLK,IO,IW/' ','O','W'/
   88  WRITE(108,4000)
 4000 FORMAT(' INPUT NUMBER OF PASSES. 1-7')
      READ(105,3000,ERR=88) NPASS
      OUTPUT NPASS
      WRITE(108,2000)
 2000 FORMAT(' INPUT PASS NO.  1-7')
      READ(105,3000,ERR=88)  (IP(KK),KK=1,NPASS)
      WRITE(108,2500)  (IP(KK),KK=1,NPASS)
 2500 FORMAT(7X,7I3)
 3000 FORMAT(7I1)
      NSAMP=0
      N1=0
      N2=0
      IND=0
      NV=NPASS*2+1
      DO 100 K=1,209
C
```

```
C                 READ LABELED DOTS  (UNDER FORMAT 1000)
C
      READ(1,1000,END=999) IG1,IG2,X
 1000 FORMAT(2A1,8F5.1,/,2X,8F5.1,/,2X,8F5.1,/,2X,4F5.1)
      IF(IG1.NE.IBLK) GO TO 100
      IF(IG2.EQ.IBLK.OR.IG2.EQ.IO) GO TO 100
      DO 10 KK=1,NPASS
      I1=(IP(KK)-1)*4+1
      IF(X(I1).GT.98.5) GO TO 100
   10 CONTINUE
      NSAMP=NSAMP+1
      ICLASS(NSAMP)=1
      IF(IG2.NE.IW) ICLASS(NSAMP)=2
      IND=IND+1
      D(IND)=1.
      IF(ICLASS(NSAMP).EQ.2) D(IND)=-1.
      DO 30 NPP=1,NPASS
      I1=(IP(NPP)-1)*4+1
      CALL KAUTH(X(I1),B,G)
      D(IND+1)=B
      D(IND+2)=G
      DO 20 J=1,2
      IF(ICLASS(NSAMP).EQ.2) D(IND+J)=-D(IND+J)
   20 CONTINUE
      IND=IND+2
   30 CONTINUE
      IF(ICLASS(NSAMP).EQ.1) N1=N1+1
      IF(ICLASS(NSAMP).EQ.2) N2=N2+1
  100 CONTINUE
  999 CONTINUE
      OUTPUT N1,N2,NSAMP
      RETURN
      END
```

```
C    .......................................................
C
C                   ****   B,G  QUAD   ***
C
C     THIS VERSION OF GETD USES THE FIRST 2 TASSEL CAP VARIABLES PLUS
C     QUADRATIC TERMS AS FEATURES. (UP TO 7 PASSES)
C
C     D   -   THIS IS THE DATA ARRAY WHICH IS RETURNED TO THE
C             CALLING ROUTINE.  THE DIMENSION IS  NV  BY NSAMP.
C
C             THE COLUMN VECTORS OF  D  CONSIST OF
C
C                  ( 1. , X1, X2, ... , XN)
C
C              IF THE PROTOTYPE IS IN CLASS 1 AND THE NEGATIVE
C              OF THE ABOVE VECTOR IF THE PROTOTYPE IS IN
C              CLASS 2.
C
C      NSAMP-  THE NUMBER OF PROTOTYPES.  THIS VALUE IS RETURNED TO
C              THE CALLING PROGRAM.
C
C      NV   -  THE NUMBER OF VARIABLES PLUS ONE FOR THE
C              CONSTANT TERM. THIS VALUE IS RETURNED TO THE CALLING
C              PROGRAM.
C
C      ICLASS- THIS IS A VECTOR, RETURNED TO THE CALLING PROGRAM WHICH
C             IDENTIFIES (1 OR 2) THE CLASS ASSIGNMENT OF EACH
C             PROTOTYPE.
C
C     ............................................................
C
      SUBROUTINE GETD(D,NSAMP,NV,ICLASS)
      DIMENSION D(1),ICLASS(1),X(28),IP(7)
      DATA IBLK,IO,IW/' ','O','W'/
   88  WRITE(108,4000)
 4000 FORMAT(' INPUT NUMBER OF PASSES. 1-7')
      READ(105,3000,ERR=88) NPASS
      OUTPUT NPASS
      WRITE(108,2000)
 2000 FORMAT(' INPUT PASS NO.  1-7')
      READ(105,3000,ERR=88)  (IP(KK),KK=1,NPASS)
      WRITE(108,2500)  (IP(KK),KK=1,NPASS)
 2500 FORMAT(7X,7I3)
 3000 FORMAT(7I1)
      NSAMP=0
      N1=0
      N2=0
      IND=0
      NV=NPASS*5+1
```

```
      DO 100 K=1,209
C
C              READ LABELED DOTS  (UNDER FORMAT 1000)
C
      READ(1,1000,END=999) IG1,IG2,X
 1000 FORMAT(2A1,8F5.1,/,2X,8F5.1,/,2X,8F5.1,/,2X,4F5.1)
      IF(IG1.NE.IBLK) GO TO 100
      IF(IG2.EQ.IBLK.OR.IG2.EQ.IO) GO TO 100
      DO 10 KK=1,NPASS
      I1=(IP(KK)-1)*4+1
      IF(X(I1).GT.98.5) GO TO 100
   10 CONTINUE
      NSAMP=NSAMP+1
      ICLASS(NSAMP)=1
      IF(IG2.NE.IW) ICLASS(NSAMP)=2
      IND=IND+1
      D(IND)=1.
      IF(ICLASS(NSAMP).EQ.2) D(IND)=-1.
      DO 30 NPP=1,NPASS
      I1=(IP(NPP)-1)*4+1
      CALL KAUTH(X(I1),B,G)
      D(IND+1)=B
      D(IND+2)=G
      D(IND+3)=B*B
      D(IND+4)=G*G
      D(IND+5)=B*G
      IF(ICLASS(NSAMP).EQ.1) GO TO 25
      DO 20 J=IND+1,IND+5
   20 D(J)=-D(J)
   25 IND=IND+5
   30 CONTINUE
      IF(ICLASS(NSAMP).EQ.1) N1=N1+1
      IF(ICLASS(NSAMP).EQ.2) N2=N2+1
  100 CONTINUE
  999 CONTINUE
      OUTPUT N1,N2,NSAMP
      RETURN
      END
```

```
C     ..................................................
C
C       THIS ROUTINE COMPUTES THE TASSEL CAP COORDINATES
C       B AND G FROM THE FOUR LANDSAT CHANNELS IN THE VECTOR X.
C
C     ..................................................
C
        SUBROUTINE KAUTH(X,B,G)
        DIMENSION X(1),FB(4),FG(4)
        DATA FB,FG/.33231,.60316,.67581,.26278,
       *           -.28317,-.66006,.57735,.38833/
        B=0.
        G=0.
        DO 1 I=1,4
        B=B+X(I)*FB(I)
      1 G=G+X(I)*FG(I)
        RETURN
        END
```

```
      SUBROUTINE GINV2M(A,MR,MC,NR,NC,KZ,U,AFLAG,ATEMP,TOK)             GINV0000
C-----------------------------------------------------------------------GINV0010
C                                                                       GINV0020
C     THIS ROUTINE COMPUTES THE TRANSPOSE OF THE GENERALIZED INVERSE    GINV0030
C         OF A AND STORES IT IN A                                       GINV0040
C  MR  IS THE MAXIMUM ROW DIMENSION OF A                                GINV0050
C  MC  IS THE MAXIMUM COLUMN DIMENSION OF A                             GINV0060
C  NR   IS THE NUMBER OF ROWS IN A                                      GINV0070
C  NC   IS THE NUMBER OF COLUMNS IN A                                   GINV0080
C  KZ   OUTPUT, KZ= 0 IMPLIES THAT A IS SINGULAR                        GINV0090
C               KZ = 1 IMPLIES THAT A IS NONSINGULAR                    GINV0100
C  U      TEMPORARY WORKING STORAGE, DIMENSIONED MC BY MC               GINV0110
C  AFLAG   TEMPORARY WORKING STORAGE, DIMENSIONED MC                    GINV0120
C  ATEMP   TEMPORARY WORKING STORAGE, DIMENSIONED MC                    GINV0130
C  TOK   TOLERANCE FOR INNER PRODUCT ZERO                               GINV0140
C  TOL   TOLREANCE FOR THE RATIO OF TWO INNER PRODUCTS                  GINV0150
C                                                                       GINV0160
C-----------------------------------------------------------------------GINV0170
      DOUBLE PRECISION SUM , FAC                                        GINV0180
      DIMENSION A(MR,MC),U(MC,MC),AFLAG(MC),ATEMP(MC)                   GINV0190
      KZ = 1                                                            GINV0200
      DO 10 I=1,NC                                                      GINV0210
      DO 5 J=1,NC                                                       GINV0220
    5 U(I,J) = 0.0                                                      GINV0230
   10 U(I,I) = 1.0                                                      GINV0240
      DO 12 L=1,NC                                                      GINV0250
      FAC = DOT(MR,NR,A,L,L)                                            GINV0260
      K=L                                                               GINV0270
      IF(FAC.GT.TOK) GO TO 11                                           GINV0280
      DO 13 J=1,NR                                                      GINV0290
   13 A(J,L)=0.0                                                        GINV0300
      KZ=0                                                              GINV0310
   12 AFLAG(L)=0.0                                                      GINV0320
      GO TO 1000                                                        GINV0330
   11 FAC= 1.0/SQRT(FAC)                                                GINV0340
      L=K                                                               GINV0350
      AFLAG(L)=1.0                                                      GINV0360
      DO 15 I=1,NR                                                      GINV0370
   15 A(I,L) = A(I,L)*FAC                                               GINV0380
      DO 20 I=1,NC                                                      GINV0390
   20 U(I,L) = U(I,L)*FAC                                               GINV0400
C                                                                       GINV0410
C     DEPENDENT COL TOLERANCE FOR N BIT FLOATING POINT FRACTION         GINV0420
C                                                                       GINV0430
      N=27                                                              GINV0440
      TOL=(10.*0.5**N)**2                                               GINV0450
      L1=L+1                                                            GINV0460
      IF(L1.LE.NC) GO TO 21                                             GINV0470
      DO 22 I=1,NR                                                      GINV0480
   22 A(I,L) = A(I,L)*FAC                                               GINV0490
```

```
      GO TO 1000
   21 DO 100 J=2,NC
      DOT1 = DOT(MR,NR,A,J,J)
      JM1 = J-1
      DO 50 L=1,2
      DO 30 K=1,JM1
   30 ATEMP(K) = DOT(MR,NR,A,J,K)
      DO 45 K=1,JM1
      DO 35 I=1,NR
   35 A(I,J) = A(I,J)-ATEMP(K)*A(I,K)*AFLAG(K)
      DO 40 I=1,NC
   40 U(I,J) = U(I,J)-ATEMP(K)*U(I,K)
   45 CONTINUE
   50 CONTINUE
      DOT2 = DOT(MR,NR,A,J,J)
      IF((DOT2/DOT1)-TOL) 55,55,70
   55 DO 61 I=1,JM1
      SUM=0.0
      DO 60 K=1,I
   60 SUM = SUM + U(K,I)*U(K,J)
   61 ATEMP(I) = SUM
      DO 63 I=1,NR
      SUM = 0.0
      DO 65 K=1,JM1
   65 SUM = SUM - A(I,K)  *ATEMP(K)*AFLAG(K)
   63 A(I,J) = SUM
      AFLAG(J) = 0.0
      KZ = 0
      FAC = DOT(MC,NC,U,J,J)
      IF(FAC.GT.TOK) GO TO 66
      DO 67 I=1,NC
   67 U(I,J) = 0.0
      GO TO 100
   66 FAC = 1.0/SQRT(FAC)
      GO TO 75
   70 IF(DOT2.GT.TOK) GO TO 71
      KZ = 0
      AFLAG(J) = 0.0
      DO 72 I=1,NR
   72 A(I,J) = 0.0
      GO TO 100
   71 AFLAG(J) = 1.0
      FAC = 1.0/SQRT(DOT2)
   75 DO 80 I=1,NR
   80 A(I,J)=A(I,J)*FAC
      DO 85 I=1,NC
   85 U(I,J) = U(I,J)*FAC
  100 CONTINUE
      DO 130 J=1,NC
      DO 130 I=1,NR
```

```
      FAC = 0.0                                                         GINV1000
      DO 120 K=J,NC                                                     GINV1010
  120 FAC = FAC+A(I,K)*U(J,K)                                           GINV1020
  130 A(I,J) = FAC                                                      GINV1030
 1000 RETURN                                                            GINV1040
      END                                                               GINV1050
      FUNCTION DOT(MR,NR,A,JC,KC)                                       GINV1060
C                                                                       GINV1070
C     COMPUTES THE INNER PRODUCT OF COLUMNS JC AND KC                   GINV1080
C     OF MATRIX A                                                       GINV1090
      DIMENSIÓN A(MR,1)                                                 GINV1100
      DOT = 0.0                                                         GINV1110
      DO 5 I=1,NR                                                       GINV1120
    5 DOT = DOT+ A(I,JC)*A(I,KC)                                        GINV1130
       RETURN                                                           GINV1140
        END                                                             GINV1150
```

```
C
C     ..............................................................
C
C        SUBROUTINE MADD
C
C        PURPOSE
C           ADD TWO MATRICES ELEMENT BY ELEMENT TO FORM RESULTANT
C           MATRIX
C
C        USAGE
C           CALL MADD(A,B,R,N,M,MSA,MSB)
C
C        DESCRIPTION OF PARAMETERS
C           A - NAME OF INPUT MATRIX
C           B - NAME OF INPUT MATRIX
C           R - NAME OF OUTPUT MATRIX
C           N - NUMBER OF ROWS IN A,B,R
C           M - NUMBER OF COLUMNS IN A,B,R
C           MSA - ONE DIGIT NUMBER FOR STORAGE MODE OF MATRIX A
C                  0 - GENERAL
C                  1 - SYMMETRIC
C                  2 - DIAGONAL
C           MSB - SAME AS MSA EXCEPT FOR MATRIX B
C
C        REMARKS
C           NONE
C
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C           LOC
C
C        METHOD
C           STORAGE MODE OF OUTPUT MATRIX IS FIRST DETERMINED. ADDITION
C           OF CORRESPONDING ELEMENTS IS THEN PERFORMED.
C           THE FOLLOWING TABLE SHOWS THE STORAGE MODE OF THE OUTPUT
C           MATRIX FOR ALL COMBINATIONS OF INPUT MATRICES
C                       A                B                R
C                  GENERAL          GENERAL          GENERAL
C                  GENERAL          SYMMETRIC        GENERAL
C                  GENERAL          DIAGONAL         GENERAL
C                  SYMMETRIC        GENERAL          GENERAL
C                  SYMMETRIC        SYMMETRIC        SYMMETRIC
C                  SYMMETRIC        DIAGONAL         SYMMETRIC
C                  DIAGONAL         GENERAL          GENERAL
C                  DIAGONAL         SYMMETRIC        SYMMETRIC
C                  DIAGONAL         DIAGONAL         DIAGONAL
C
C     ..............................................................
C
      SUBROUTINE MADD(A,B,R,N,M,MSA,MSB)
      DIMENSION A(1),B(1),R(1)
```

```
C
C        DETERMINE STORAGE MODE OF OUTPUT MATRIX
C
      IF(MSA-MSB) 7,5,7
    5 CALL LOC(N,M,NM,N,M,MSA)
      GO TO 100
    7 MTEST=MSA*MSB
      MSR=0
      IF(MTEST) 20,20,10
   10 MSR=1
   20 IF(MTEST-2) 35,35,30
   30 MSR=2
C
C        LOCATE ELEMENTS AND PERFORM ADDITION
C
   35 DO 90 J=1,M
      DO 90 I=1,N
      CALL LOC(I,J,IJR,N,M,MSR)
      IF(IJR) 40,90,40
   40 CALL LOC(I,J,IJA,N,M,MSA)
      AEL=0.0
      IF(IJA) 50,60,50
   50 AEL=A(IJA)
   60 CALL LOC(I,J,IJB,N,M,MSB)
      BEL=0.0
      IF(IJB) 70,80,70
   70 BEL=B(IJB)
   80 R(IJR)=AEL+BEL
   90 CONTINUE
      RETURN
C
C        ADD MATRICES FOR OTHER CASES
C
  100 DO 110 I=1,NM
  110 R(I)=A(I)+B(I)
      RETURN
      END
```

```
C
C     ..................................................................
C
C        SUBROUTINE MPRD
C
C        PURPOSE
C           MULTIPLY TWO MATRICES TO FORM A RESULTANT MATRIX
C
C        USAGE
C           CALL MPRD(A,B,R,N,M,MSA,MSB,L)
C
C        DESCRIPTION OF PARAMETERS
C           A - NAME OF FIRST INPUT MATRIX
C           B - NAME OF SECOND INPUT MATRIX
C           R - NAME OF OUTPUT MATRIX
C           N - NUMBER OF ROWS IN A AND R
C           M - NUMBER OF COLUMNS IN A AND ROWS IN B
C           MSA - ONE DIGIT NUMBER FOR STORAGE MODE OF MATRIX A
C                  0 - GENERAL
C                  1 - SYMMETRIC
C                  2 - DIAGONAL
C           MSB - SAME AS MSA EXCEPT FOR MATRIX B
C           L - NUMBER OF COLUMNS IN B AND R
C
C        REMARKS
C           MATRIX R CANNOT BE IN THE SAME LOCATION AS MATRICES A OR B
C           NUMBER OF COLUMNS OF MATRIX A MUST BE EQUAL TO NUMBER OF ROW
C           OF MATRIX B
C
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C           LOC
C
C        METHOD
C           THE M BY L MATRIX B IS PREMULTIPLIED BY THE N BY M MATRIX A
C           AND THE RESULT IS STORED IN THE N BY L MATRIX R. THIS IS A
C           ROW INTO COLUMN PRODUCT.
C           THE FOLLOWING TABLE SHOWS THE STORAGE MODE OF THE OUTPUT
C           MATRIX FOR ALL COMBINATIONS OF INPUT MATRICES
C                        A                B                R
C                     GENERAL          GENERAL          GENERAL
C                     GENERAL          SYMMETRIC        GENERAL
C                     GENERAL          DIAGONAL         GENERAL
C                     SYMMETRIC        GENERAL          GENERAL
C                     SYMMETRIC        SYMMETRIC        GENERAL
C                     SYMMETRIC        DIAGONAL         GENERAL
C                     DIAGONAL         GENERAL          GENERAL
C                     DIAGONAL         SYMMETRIC        GENERAL
C                     DIAGONAL         DIAGONAL         DIAGONAL
C
C     ..................................................................
```

```
C
      SUBROUTINE MPRD(A,B,R,N,M,MSA,MSB,L)
      DIMENSION A(1),B(1),R(1)
C
C        SPECIAL CASE FOR DIAGONAL BY DIAGONAL
C
      MS=MSA*10+MSB
      IF(MS-22) 30,10,30
   10 DO 20 I=1,N
   20 R(I)=A(I)*B(I)
      RETURN
C
C        ALL OTHER CASES
C
   30 IR=1
      DO 90 K=1,L
      DO 90 J=1,N
      R(IR)=0
      DO 80 I=1,M
      IF(MS) 40,60,40
   40 CALL LOC(J,I,IA,N,M,MSA)
      CALL LOC(I,K,IB,M,L,MSB)
      IF(IA) 50,80,50
   50 IF(IB) 70,80,70
   60 IA=N*(I-1)+J
      IB=M*(K-1)+I
   70 R(IR)=R(IR)+A(IA)*B(IB)
   80 CONTINUE
   90 IR=IR+1
      RETURN
      END
```

```
C
C     ...............................................................
C
C        SUBROUTINE TPRD
C
C        PURPOSE
C           TRANSPOSE A MATRIX AND POSTMULTIPLY BY ANOTHER TO FORM
C           A RESULTANT MATRIX
C
C        USAGE
C           CALL TPRD(A,B,R,N,M,MSA,MSB,L)
C
C        DESCRIPTION OF PARAMETERS
C           A - NAME OF FIRST INPUT MATRIX
C           B - NAME OF SECOND INPUT MATRIX
C           R - NAME OF OUTPUT MATRIX
C           N - NUMBER OF ROWS IN A AND B
C           M - NUMBER OF COLUMNS IN A AND ROWS IN R
C           MSA - ONE DIGIT NUMBER FOR STORAGE MODE OF MATRIX A
C                  0 - GENERAL
C                  1 - SYMMETRIC
C                  2 - DIAGONAL
C           MSB - SAME AS MSA EXCEPT FOR MATRIX B
C           L - NUMBER OF COLUMNS IN B AND R
C
C        REMARKS
C           MATRIX R CANNOT BE IN THE SAME LOCATION AS MATRICES A OR B
C
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C           LOC
C
C        METHOD
C           MATRIX TRANSPOSE OF A IS NOT ACTUALLY CALCULATED. INSTEAD,
C           ELEMENTS IN MATRIX A ARE TAKEN COLUMNWISE RATHER THAN
C           ROWWISE FOR MULTIPLICATION BY MATRIX B.
C           THE FOLLOWING TABLE SHOWS THE STORAGE MODE OF THE OUTPUT
C           MATRIX FOR ALL COMBINATIONS OF INPUT MATRICES
C                         A                B                R
C                      GENERAL          GENERAL          GENERAL
C                      GENERAL          SYMMETRIC        GENERAL
C                      GENERAL          DIAGONAL         GENERAL
C                      SYMMETRIC        GENERAL          GENERAL
C                      SYMMETRIC        SYMMETRIC        GENERAL
C                      SYMMETRIC        DIAGONAL         GENERAL
C                      DIAGONAL         GENERAL          GENERAL
C                      DIAGONAL         SYMMETRIC        GENERAL
C                      DIAGONAL         DIAGONAL         DIAGONAL
C
C     ...............................................................
C
```

```
      SUBROUTINE TPRD(A,B,R,N,M,MSA,MSB,L)
      DIMENSION A(1),B(1),R(1)
C
C        SPECIAL CASE FOR DIAGONAL BY DIAGONAL
C
      MS=MSA*10+MSB
      IF(MS-22) 30,10,30
   10 DO 20 I=1,N
   20 R(I)=A(I)*B(I)
      RETURN
C
C        MULTIPLY TRANSPOSE OF A BY B
C
   30 IR=1
      DO 90 K=1,L
      DO 90 J=1,M
      R(IR)=0.0
      DO 80 I=1,N
      IF(MS) 40,60,40
   40 CALL LOC(I,J,IA,N,M,MSA)
      CALL LOC(I,K,IB,N,L,MSB)
      IF(IA) 50,80,50
   50 IF(IB) 70,80,70
   60 IA=N*(J-1)+I
      IB=N*(K-1)+I
   70 R(IR)=R(IR)+A(IA)*B(IB)
   80 CONTINUE
   90 IR=IR+1
      RETURN
      END
```

```
C
C     ..................................................................
C
C        SUBROUTINE LOC
C
C        PURPOSE
C           COMPUTE A VECTOR SUBSCRIPT FOR AN ELEMENT IN A MATRIX OF
C           SPECIFIED STORAGE MODE
C
C        USAGE
C           CALL LOC (I,J,IR,N,M,MS)
C
C        DESCRIPTION OF PARAMETERS
C           I    - ROW NUMBER OF ELEMENT
C           J    - COLUMN NUMBER  OF ELEMENT
C           IR   - RESULTANT VECTOR SUBSCRIPT
C           N    - NUMBER OF ROWS IN MATRIX
C           M    - NUMBER OF COLUMNS IN MATRIX
C           MS   - ONE DIGIT NUMBER FOR STORAGE MODE OF MATRIX
C                  0 - GENERAL
C                  1 - SYMMETRIC
C                  2 - DIAGONAL
C
C        REMARKS
C           NONE
C
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C           NONE
C
C        METHOD
C           MS=0   SUBSCRIPT IS COMPUTED FOR A MATRIX WITH N*M ELEMENTS
C                  IN STORAGE (GENERAL MATRIX)
C           MS=1   SUBSCRIPT IS COMPUTED FOR A MATRIX WITH N*(N+1)/2 IN
C                  STORAGE (UPPER TRIANGLE OF SYMMETRIC MATRIX). IF
C                  ELEMENT IS IN LOWER TRIANGULAR PORTION, SUBSCRIPT IS
C                  CORRESPONDING ELEMENT IN UPPER TRIANGLE.
C           MS=2   SUBSCRIPT IS COMPUTED FOR A MATRIX WITH N ELEMENTS
C                  IN STORAGE (DIAGONAL ELEMENTS OF DIAGONAL MATRIX).
C                  IF ELEMENT IS NOT ON DIAGONAL (AND THEREFORE NOT IN
C                  STORAGE), IR IS SET TO ZERO.
C
C     ..................................................................
C
      SUBROUTINE LOC(I,J,IR,N,M,MS)
C
      IX=I
      JX=J
      IF(MS-1) 10,20,30
   10 IRX=N*(JX-1)+IX
      GO TO 36
   20 IF(IX-JX) 22,24,24
   22 IRX=IX+(JX*JX-JX)/2
      GO TO 36
   24 IRX=JX+(IX*IX-IX)/2
      GO TO 36
   30 IRX=0
      IF(IX-JX) 36,32,36
   32 IRX=IX
   36 IR=IRX
      RETURN
      END
```

```
C
C     ..................................................................
C
C        SUBROUTINE MSUB
C
C        PURPOSE
C           SUBTRACT TWO MATRICES ELEMENT BY ELEMENT TO FORM RESULTANT
C           MATRIX
C
C        USAGE
C           CALL MSUB(A,B,R,N,M,MSA,MSB)
C
C        DESCRIPTION OF PARAMETERS
C           A - NAME OF INPUT MATRIX
C           B - NAME OF INPUT MATRIX
C           R - NAME OF OUTPUT MATRIX
C           N - NUMBER OF ROWS IN A,B,R
C           M - NUMBER OF COLUMNS IN A,B,R
C           MSA - ONE DIGIT NUMBER FOR STORAGE MODE OF MATRIX A
C                  0 - GENERAL
C                  1 - SYMMETRIC
C                  2 - DIAGONAL
C           MSB - SAME AS MSA EXCEPT FOR MATRIX B
C
C        REMARKS
C           NONE
C
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C           LOC
C
C        METHOD
C           STRUCTURE OF OUTPUT MATRIX IS FIRST DETERMINED. SUBTRACTION
C           OF MATRIX B ELEMENTS FROM CORRESPONDING MATRIX A ELEMENTS
C           IS THEN PERFORMED.
C           THE FOLLOWING TABLE SHOWS THE STORAGE MODE OF THE OUTPUT
C           MATRIX FOR ALL COMBINATIONS OF INPUT MATRICES
C                       A                B                R
C                    GENERAL          GENERAL          GENERAL
C                    GENERAL          SYMMETRIC        GENERAL
C                    GENERAL          DIAGONAL         GENERAL
C                    SYMMETRIC        GENERAL          GENERAL
C                    SYMMETRIC        SYMMETRIC        SYMMETRIC
C                    SYMMETRIC        DIAGONAL         SYMMETRIC
C                    DIAGONAL         GENERAL          GENERAL
C                    DIAGONAL         SYMMETRIC        SYMMETRIC
C                    DIAGONAL         DIAGONAL         DIAGONAL
C
C     ..................................................................
C
      SUBROUTINE MSUB(A,B,R,N,M,MSA,MSB)
```

```
      DIMENSION A(1),B(1),R(1)
C
C        DETERMINE STORAGE MODE OF OUTPUT MATRIX
C
      IF(MSA-MSB) 7,5,7
    5 CALL LOC(N,M,NM,N,M,MSA)
      GO TO 100
    7 MTEST=MSA*MSB
      MSR=0
      IF(MTEST) 20,20,10
   10 MSR=1
   20 IF(MTEST-2) 35,35,30
   30 MSR=2
C
C        LOCATE ELEMENTS AND PERFORM SUBTRACTION
C
   35 DO 90 J=1,M
      DO 90 I=1,N
      CALL LOC(I,J,IJR,N,M,MSR)
      IF(IJR) 40,90,40
   40 CALL LOC(I,J,IJA,N,M,MSA)
      AEL=0.0
      IF(IJA) 50,60,50
   50 AEL=A(IJA)
   60 CALL LOC(I,J,IJB,N,M,MSB)
      BEL=0.0
      IF(IJB) 70,80,70
   70 BEL=B(IJB)
   80 R(IJR)=AEL-BEL
   90 CONTINUE
      RETURN
C
C        SUBTRACT MATRICES FOR OTHER CASES
C
  100 DO 110 I=1,NM
  110 R(I)=A(I)-B(I)
      RETURN
      END
```

EXPERIMENTAL DESIGN FOR OPTIMAL PASS SELECTIONS

The objective of this task is to develop and demonstrate feature selection programs for the purpose of selecting a priori statistically optimum subsets of LANDSAT acquisitions for analysis to separate wheat from nonwheat when given an adequate sample of labelled wheat and nonwheat LACIE segment pixel data. Throughout the remainder of this report we will identify a LANDSAT acquisition, not by its Julian date, but by a suitable (e.g. Robinson) wheat growth stage index. Denote the distinct values of this index corresponding to growth stages, by

$$I_W = \{ t_1, \ldots, t_k \} \qquad I_S = \{ u_1, \ldots, u_\ell \}$$

for winter wheat and spring wheat respectively. The problem is as follows. Select the subsets of size 1, 2, 3 and 4 of I_W and I_S which maximize the separations of spectral measurement of wheat and nonwheat. In the spring wheat areas we wish to separate small grains from other crops.

The Data Set

The data set will consist of a number of blind sites, preferably from more than one growing season. Multiple acquisitions will be required, however, we will not require every possible acquisition from every segment. Each acquisition will be assigned the appropriate wheat growth stage index. If a segment has multiple acquisitions with the same index, only one (selected at random) will be retained.

The data can now be arranged as follows:

Growth Stage Indices

Winter Wheat Segments

	t_1	t_2	t_3	t_4	. . .	t_n
s_1	x		x	x	. . .	x
s_2		x	x	x	. . .	x
s_3	x	x			. . .	x
⋮					. . .	
s_n	x	x	x	x	. . .	

Growth Stage Indices

Spring Wheat Segments

	u_1	u_2	u_3	u_4	. . .	u_ℓ
s_1		x	x		. . .	x
s_2	x	x	x	x	. . .	x
s_3	x	x			. . .	x
⋮					. . .	
s_n	x			x	. . .	x

The "x"'s denote an acquisition. A "blank" will denote no acquisition available. The data will consist of labelled data as statistical signatures for wheat/nonwheat categories. We recommend that the "Dots" be used in order to avoid dependence on signatures generated from clustering in CAMS. Furthermore, we know that all signatures will not be available for all acquisitions.

Measure of Separation and Procedure

The criteria (figure of merit) used will depend on they type of data available. The two cases are:

1) Dots: Set $s_\ell(j) = \{w_1, \ldots, w_a, o_1, \ldots, o_b\}$ be the set of labelled dots for segment ℓ at crop stage t_j. Let $D_\ell(j)$ be a discriminant rule based on the prototypes $s_\ell(t_j)$. The functional form of $D_\ell(j)$ will be fixed over all segments ℓ and crop stages t_j, however, the coefficients are estimated for each segment and crop stage separately.

Let $C_\ell(j)$ be the proportion of prototypes misclassified by $D_\ell(j)$. If $n_\ell(j)$ is the number of prototypes of $s_\ell(j)$, then the figure of merit for crop stage t_j is defined to be:

$$\phi(j) = \sum_{\ell \in A(j)} \frac{n_\ell(j)\, C_\ell(j)}{N(j)}$$

where A(j) is the index set of all segments having acquisitions for crop stage t_j, and $N(j) = \sum_{\ell \in A(j)} n_\ell(j)$. That is to say, $\phi(j)$ is the misclassification rate over all segments for stage t_j. We define the best crop stage for discriminating between wheat and nonwheat (small grains and other in spring wheat areas) to be that crop stage t_j for which $\phi(j)$ is a minimum. In order to choose the best pair of crop stages, define $s_\ell(t_i, t_j)$ to be the set of two pass prototypes (8 X 1 vectors) and $D_\ell(i,j)$ the chosen discriminant rules. Now A(i,j) is the index set of segments having acquisition at crop stages t_i and t_j (note that A(i,j) A(i). Now

$$\phi(i,j) = \sum_{\ell \in A(i,j)} \frac{N_\ell(i,j)\, C_\ell(i,j)}{N(i,j)}$$

where the definitions of $n_\ell(i,j)$, $C_\ell(i,j)$, and $N(i,j)$ are extended in the obvious way. The best pair of crop stages is determined by the maximum value of ϕ. In general, we minimize

$$\phi(i_1, \ldots, i_k) = \sum_{\ell \in A(i_1, \ldots, i_k)} \frac{n_\ell(u, \ldots, i_k)\, C_\ell(k_1, \ldots, i_k)}{N(i_1, \ldots, i_k)}$$

to choose the best k crop stages. We are obviously constrained by the availability of a sufficient number of segments with acquisition at crop stages $t_{i1}, \ldots, t_{ik}$ for all such subsets of size k of I_W or I_S. The size of $A(i_1, \ldots, i_k)$ must be greater than two for all $i_1, \ldots, i_k$. (Interesting values of k are 1, 2, 3, 4.) This constraint will most likely require use of all blind sites in the study.

The next issue is the definition of the discriminant rules $D_\ell(i_1, \ldots, i_k$ We propose two such rules. Since we have only dots, a nonparametric discriminant rule is recommended (although statistical attributes of the categories or subcategories might be constructed).

Rule 1. For each segment find the hyperplane (which may or may not separate the two categories of prototypes) determined by some non-parametric linear discriminate function (LDF) technique.

Rule 2. Use the same LDF algorithm to determine a quadratic discriminant function. Specifically, transform each acquisition to TACAP brightness (b) - greenness (G) space, B_1 , G_1 , $\cdots$, B_k , G_k. . Construct dummy variables B_i^2 , G_i^2 , B_iG_i for each acquisition so that (together with B_i and G_i) five variables are defined per acqusition for application of the LDF procedure. This will yield a quadratic discriminant function based on the B-G coordinate system.

DATA REQUIREMENTS FOR LIMITED TEST OF EXPERIMENTAL DESIGN FOR OPTIMAL PASS SELECTION.

The University of Houston will perform a limited test of the optimal pass selection experimental design, described earlier in this report, in 1979 with the following data requirements.

> At least 10 blind sites in winter wheat strata, which have 4 - 7 acquisitions, will be designated. Labeled dots along with NASA specified crop indices for each segment and acquisition will be supplied to UH in 9 track 800 BPI EDCDIC tapes.

We will perform the limited test and document test procedures within the scope of the 1979 contract.

Feature Selection for Best Mean Square Approximation of Class Densities

1. Introduction

The purpose of this note is to describe a general mean square approach to linear feature selection which connects certain generalized Fisher criteria in discriminant analysis with a measure of pattern class separation introduced by Devijver[(3)]. The former are typical of those criteria which utilize only low order information about the pattern class distributions, while the latter requires that the class distributions be known, or at least accurately estimated.

Let X denote a random vector in real n-space R^n which arises from one of m pattern classes $\Pi_1, \ldots, \Pi_m$ having known prior probabilities $\alpha_1, \ldots, \alpha_m$, where $\alpha_i > 0$ and $\sum_{i=1}^{m} \alpha_i = 1$. Let $F_j(x)$ denote the j^{th} class conditional distribution function of X and let $F(x) = \sum_{i=1}^{m} \alpha_i F_i(x)$ denote the mixture distribution. For a given measureable transformation $T:R^n \to R^k$ let $G_j(y, T)$ and $G(y, T)$ denote, respectively, the j^{th} class conditional distribution and mixture distribution of the random variable $Y = TX$. We let $f_j(x)$ (resp. $g_j(y, T)$) denote the class conditional densities of X (resp. Y) with respect to their corresponding mixture distributions; i.e.,

$$f_j = \frac{dF_j}{dF} \quad \text{and} \quad g_j(., T) = \frac{dG_j(., T)}{dG(., T)}.$$

We will restrict our attention to the set of linear transformations T of rank k, and assume that each pattern class Π_i has a mean μ_i and positive definite covariance matrix Ω_i. Let $\mu = \sum_{i=1}^{m} \alpha_i \mu_i$ and let

$$S_B = \sum_{i=1}^{m} \alpha_i (\mu_i - \mu)(\mu_i - \mu)^T$$

$$S_W = \sum_{i=1}^{m} \alpha_i \, \Omega_i$$

and $$S = S_W + S_B$$

denote the between class scatter matrix, the average within class scatter matrix, and the total scatter matrix respectively.

A number of interesting feature selection criteria can be formulated using only the parameters μ, μ_i, Ω_i, S, S_W, S_B; e.g., the criteria proposed by Kittler and Young [8], Foley and Sammon[4], Fukanaga and Koontz[6], and the discrete analogue of the modified Karhunen-Loeve expansion of Chien and Fu[1]. The modified K-L expansion minimizes an entropy function, and also best represents the pattern vector X in an overall least squares sense; however, its value for discrimination has been questioned by several authors (see Kittler[7]). Fukanaga[5] considers several criteria of the generalized Fisher type, including

$$J_k(T) = \mathrm{tr}(T^T S_W T)^{-1}(T^T S_B T).$$

Thus, according to this criterion, the best $k \times n$ matrix T of rank k is one which maximizes $J_k(T)$. The solution is any T which is row equivalent to a $k \times n$ matrix whose rows are linearly independent principal eigenvectors (i.e., corresponding to the largest eigenvalues) of $S_W^{-1} S_B$. We also consider a modification

$$J_k'(T) = \mathrm{tr}\,(T^T S T)^{-1}(T^T S_B T)$$

which admits the same maximizing T.

The Bayesian distance corresponding to the pattern classes $\Pi_1, \ldots, \Pi_m$, as defined by Devijver[3], is

$$B_n = \sum_{i=1}^{m} \alpha_i^2 \, E\,[f_i(X)^2]$$

$$= \sum_{i=1}^{m} \alpha_i^2 \int_{R^n} f_i(x)^2 \, dF(x).$$

Its transformed value is

$$B_k(T) = \sum_{i=1}^{m} \alpha_i^2 \int_{R^k} g_i(y, T)^2 \, dG(y, T).$$

Devijver proves a number of interesting inequalities relating B_n to the Bayes probability of misclassification, the Bhattacharrya coefficient, and other measures of class separation. In addition, he notes that Cover and Hart[2] have shown that $1-B_n$ is the asymptotic error rate of the nearest neighbor classifier.

2. Mean Square Optimality of Bayesian Distance

For a given $k \times n$ matrix T of rank k, let $L_2(T)$ denote the set of all measureable functions $\varphi : R^k \to R^1$ such that $\int_{R^k} \varphi(y)^2 \, dG(y, T) < \infty$ and let C_T be a given closed linear subspace of $L_2(T)$. Our general approach to linear feature selection is to choose that $\hat{T}$, if possible, which minimizes

$$R(T) = \sum_{i=1}^{m} \beta_i \min_{\varphi_i \in C_T} \int_{R^n} [\varphi_i(Tx) - f_i(x)]^2 \, dF(x),$$

where the β_i are positive weights. That is, we attempt to find a T which produces a set of approximations $\varphi_i(Tx)$ to the class densities $f_i(x)$ which is best in an overall mean square sense. Given such approximations we may classify observations of X according to the pseudo-Bayes rule: decide that X is from class Π_i if $\alpha_i \varphi_i(TX) > \alpha_j \varphi_j(TX)$ for each $j \neq i$. Since we are interested in classification accuracy, it seems appropriate to choose weights β_i which reflect the relative importance of the classes in the mixture distribution; e.g., $\beta_i = \alpha_i$ for all i or $\beta_i = \alpha_i^2$ for all i. For the remainder of this section we choose $\beta_i = \alpha_i^2$ and $C_T = L_2(T)$.

<u>Proposition 1</u>: For $\beta_i = \alpha_i^2$, $i = 1, \ldots, m$ and $C_T = L_2(T)$ for each T,

$$R(T) = B_n - B_k(T).$$

<u>Proof</u>: Observe that $g_i(y, T) \in L_2(T)$, since it is bounded by α_i^{-1}. Moreover, for each $\varphi \in L_2(T)$,

$$\int_{R^n} \varphi(Tx)[g_i(Tx, T) - f_i(x)] \, dF(x)$$

$$= \int_{R^n} \varphi(Tx) \, g_i(Tx, T) \, dF(x) - \int_{R^n} \varphi(Tx) \, dF_i(x)$$

$$= \int_{R^k} \varphi(y) \, g_i(y, T) \, dG(y, T) - \int_{R^k} \varphi(y) \, d\,G_i(y, T)$$

$$= 0.$$

Therefore,

$$\min_{\varphi \in L_2(T)} \int_{R^n} [\varphi(Tx) - f_i(x)]^2 \, dF(x)$$

$$= \int_{R^n} [g_i(Tx, T) - f_i(x)]^2 \, dF(x)$$

$$= \int_{R^n} f_i(x)^2 \, dF(x) - \int_{R^n} g_i(y, T)^2 \, dG(y, T).$$

The assertion of the proposition follows on multiplying by α_i^2 and summing over i.

We may summarize by saying that if there exists a $k \times n$ matrix T_0 of rank k which maximizes $B_k(T)$, then the functions $g_1(T_0x, T_0), \ldots, g_m(T_0x, T_0)$ constitute the best mean square approximation to the class densities $f_1(x), \ldots, f_m(x)$ attainable through a linear compression of the data into k dimensions. Since $B_k(QT) = B_k(T)$ for each nonsingular $k \times k$ matrix Q and each $k \times n$ matrix T of rank k, $B_k(T)$ has a maximum if and only if it has a maximum on the compact set $\{T \mid TT^T = I_{k \times k}\}$. In particular, if $B_k(T)$ is continuous, it has a maximum.

3. Best Linear Approximation of Class Densities

In this section we let C_T be the set of functions $\varphi(y) = w + b^Ty$, where w is a real number and $b \in R^k$. For simplicity, we use the notation $\varphi(y) = a^Tv(y)$, where $a = \binom{w}{b} \in R^{k+1}$ and $v(y) = \binom{1}{y} \in R^{k+1}$. For given T, $a_i = \binom{w_i}{b_i}$ minimizes

$$\int_{R^n} [a^T v(Tx) - f_i(x)]^2 \, dF(x)$$

$$= a^T [\int_{R^n} v(Tx) v(Tx)^T \, dF(x)] a$$

$$-2a^T \int_{R^n} v(Tx) \, dF_i(x) + \int_{R^n} f_i(x)^2 \, dF(x)$$

$$= a^T \left(\begin{array}{c|c} 1 & \mu^T T^T \\ \hline T\mu & TWT^T \end{array}\right) a - 2(1 \mid \mu_i^T T^T) a$$

$$+ \int_{R^n} f_i(x)^2 \, dF(x)$$

if and only if

$$\left(\begin{array}{c|c} 1 & \mu^T T^T \\ \hline T\mu & TWT^T \end{array}\right) \left(\begin{array}{c} w_i \\ \hline b_i \end{array}\right) = \left(\begin{array}{c} 1 \\ \hline T\mu_i \end{array}\right)$$

where $W = E[XX^T] = S + \mu\mu^T$. Solving this system gives

$$w_i = 1 - \mu^T T^T (TST^T)^{-1} T(\mu_i - \mu)$$

and

$$b_i = (TST^T)^{-1} T(\mu_i - \mu).$$

The corresponding squared error of approximation is

$$-1- (\mu_i - \mu)^T T^T (TST^T)^{-1} T(\mu_i - \mu) + \int_{R^n} f_i(x)^2 \, dF(x).$$

Therefore, the criterion to be minimized is

$$R(T) = -\sum_{i=1}^{m} \beta_i (\mu i - \mu)^T T^T (TST^T)^{-1} T(\mu_i - \mu)$$

$$+ \text{ terms independent of } T.$$

That is, we want to maximize

$$\hat{R}(T) = \text{ trace } (TST^T)^{-1} T\hat{S}_B T^T,$$

where

$$\hat{S}_B = \sum_{i=1}^{m} \beta_i (\mu_i - \mu)(\mu_i - \mu)^T.$$

The solution is $T = QT_0$, where T_0 is a $k \times n$ matrix whose rows are linearly independent principal eigenvalues of, $S^{-1}\hat{S}_B$ and Q is an arbitrary nonsingular $k \times k$ matrix. In particular, for $\beta_i = \alpha_i$ we obtain the same solution given by Fukanaga's criterion,

$$\text{trace } (TS_W T^T)^{-1} (TS_B T^T).$$

4. Concluding Remarks

An equivalent set of criteria for feature selection are expressions such as

$$\overline{R}(T) = \sum_{i=1}^{m} \overline{\beta} i \min_{\varphi \epsilon C_T} \int_{R^n} [\varphi(Tx) - \alpha_i f_i(x)]^2 \, dF(x)$$

42

in which the posterior probabilities $\alpha_i f_i(x)$ of the classes are approximated. If each $\overline{\beta}_i$ is chosen to be 1 $\overline{R}(T)$ is the same as $R(T)$ with $\beta_i = \alpha_i^2$. This, together with Proposition 1 and the relationship between Bayesian distance and the probability of error, seems to indicate that the choice $\beta_i = \alpha_i^2$ is a good one.

In some cases it may be numerically feasible to use $B_k(T)$ as a feature selection criterion when assumptions about the parametric form of the class distributions are made. For example, if each class distribution F_i is multivariate normal, then $B_k(T)$ reduces to an expression which is continuously differentiable in T and which, moreover, can be approximated by sample averages over an unlabeled sample from the mixture distribution. Thus descent algorithms might be successfully employed in maximizing $B_k(T)$.

Finally, we remark that there is no reason in principle why $B_k(T)$ cannot be regarded as a criterion for nonlinear feature extraction. Indeed, Proposition 1 remains true when T is any measureable transformation from R^n onto R^k.

5. Acknowledgements

The author was partially supported by NASA Johnson Space Center under contract NAS-9-15000.

REFERENCES

1. Y. T. Chien and K. S. Fu. On the generalized Karhunen-Loeve expansion. IEEE Trans. IT-15, 518-520, (1967).

2. T. M. Cover and P. E. Hart. Nearest neighbor pattern classification. IEEE Trans. IT-13, 21-27, (1967).

3. P. A. Devijver. On a new class of bounds on Bayes risk in multihypothesis pattern recognition. IEEE Trans. C-23, 70-80, (1974).

4. D. H. Foley and J. W. Sammon, Jr. An optimal set of discriminant vectors. IEEE Trans. C-24, 281-289, (1975).

5. K. Fukanaga. Introduction to Statistical Pattern Recognition, Academic Press, New York, (1972).

6. K. Fukanaga and W. L. G. Koontz. Application of the Karhunen-Loeve expansion to feature selection and ordering. IEEE Trans. C-19, 311-318 (1970).

7. J. Kittler. Mathematical methods of feature selection in pattern recognition. Int. J. Man-Machine Studies, 7, 609-637, (1975).

8. J. Kittler and P. C. Young. A new approach to feature selection based on the Karhunen-Loeve expansion, Pattern Recognition, Vol. 5, 335-352, (1973).

A CONVERGENCE CRITERION FOR OPTIMAL LINEAR COMBINATION PROCEDURES

I. Introduction

In all that follows $\mathcal{U}$ will denote the set of n×n unitary matrices. We will equip $\mathcal{U}$ with the Euclidean topology and recall that $\mathcal{U}$ is a compact set. The subset $\mathcal{H}$ of $\mathcal{U}$ consisting of those matrices of the form $H = I-2xx^T$ with $\|x\|=1$, $x \in R^n$ (the Householder transformations) is a compact subset of $\mathcal{U}$.

If $\psi:\mathcal{U} \to \text{Reals}(=R^1)$ is continuous we will define $\psi_{max} = \text{l.u.b}_{\mathcal{U}}\,\psi(U)$ and examine, for certain sequences $\{U_i\}_{i=1}^{\infty}$ in $\mathcal{U}$, the convergence of the sequences $\{\psi(U_i)\}_{i=1}^{\infty}$ to ψ_{max}. Sequences of this nature have been constructed and studied by Decell and Quirein[1], Decell and Marani[6], Decell and Mayekar in connection with maximizing certain class separability functions in pattern classification problems. Throughout this paper $\mathcal{H}$, $\mathcal{U}$ and ψ will be as described above.

Definition 1. ψ will be called $\mathcal{H}$-sloped provided $U \in \mathcal{U}$ and $\psi(U) < \psi_{max}$ imply there exists some H (dependent on U) such that $\psi(U) < \psi(HU) \le \psi_{max}$.

Definition 2. A sequence $\{U_i\}_{i=1}^{\infty}$ in $\mathcal{U}$ will be called ψ-convergent provided $\{\psi(U_i)\}_{i=1}^{\infty}$ converges.

Definition 3. A sequence $\{U_i\}_{i=1}^{\infty}$ in $\mathcal{U}$ will be called a ψ-Householder sequence provided $H \in \mathcal{H}$ and i an integer imply

(1) $\psi(U_i) \le \psi(U_{i+1})$

(2) $\psi(HU_i) \le \psi(U_{i+1})$

ORIGINAL PAGE IS OF POOR QUALITY

II. Convergence

Proposition 1. Each ψ-Householder sequence $\{U_i\}_{i=1}$ is ψ-convergent and $\lim_i \psi(U_i) = \psi(U) = \text{l.u.b}_i\,\psi(U_i)$ for some $U \in \mathcal{U}$ which is the limit of a subsequence of $\{U_i\}_{i=1}^{\infty}$.

Proof: Let $\{U_i\}_{i=1}^{\infty}$ be a ψ-Householder sequence. Since, by definition, $\{\psi(U_i)\}_{i=1}^{\infty}$ is a monotone increasing sequence bounded by ψ_{max} it must converge to $\text{l.u.b.}_i\,\psi(U_i)$. Since $\mathcal{U}$ is compact, some subsequence $\{U_{i_k}\}_{k=1}^{\infty}$ must converge to $U \in \mathcal{U}$. Moreover, the continuity of ψ insures that $\lim_k \psi(U_{i_k}) = \psi(U)$, that is, $\{\psi(U_{i_k})\}_{k=1}^{\infty}$ is a convergent subsequence of $\{\psi(U_i)\}_{i=1}$ and the convergence of the latter insures the concl usion of the proposition. This completes the proof.

Proposition 2. Each Ψ-Householder sequence Ψ-converges to Ψ_{max} if and only if Ψ is $\mathcal{H}$-sloped.

Proof: Suppose each Ψ-Householder sequence Ψ-converges to Ψ_{max}. If Ψ is not $\mathcal{H}$-sloped then there is some $U \in \mathcal{U}$ for which $\Psi(U) < \Psi_{max}$ and $\Psi(HU) \leq \Psi(U)$ for each H in $\mathcal{H}$. The sequence $\{U_i\}_{i=1}^{\infty}$ with $U_i = U$ for each i is clearly a Ψ-Householder sequence for which $\lim_i \Psi(U_i) < \Psi_{max}$, a contradicti on.

If Ψ is $\mathcal{H}$-sloped and $\{U_i\}_{i=1}^{\infty}$ is a Ψ-Householder sequence then, according to Proposition 1, there is some $U \in \mathcal{U}$ and some subsequence $\{U_{i_j}\}_{j=1}^{\infty}$ for which $\lim_i \Psi(U_i) = \Psi(U)$ and $U = \lim_j U_{i_j}$. Now if $\Psi(U) < \Psi_{max}$ there exists H in $\mathcal{H}$ such that $\Psi(U) < \Psi(HU) \leq \Psi_{max}$. Since both Ψ and the mapping $H \to HU$ are continuous it follows that $\Psi(U) = \lim_i \Psi(U_i) < \Psi(HU) = \lim_j \Psi(HU_{i_j}) \leq \lim_j \Psi(U_{i_j+1}) = \Psi(U)$. The latter inequality is absurd and hence $\lim_i \Psi(U_i) = \Psi_{max}$. This completes the proof.

The next proposition sheds some light on the nature of the convergence of Ψ-Householder sequences for $\mathcal{H}$-sloped Ψ. This result is of particular importance in applying algorithms based upon construction of Ψ-Householder sequences such as those described in [1], [6] and [7].

Proposition 3. If $\{U_i\}_{i=1}^{\infty}$ is a Ψ-Householder sequence and Ψ is $\mathcal{H}$-sloped then exactly one of the following hold:

(1) $\{\Psi(U_i)\}_{i=1}^{\infty}$ is strictly monotonic (and convergent to Ψ_{max});

(2) for some integer k, l.u.b. $\Psi(HU_k) \leq \Psi(U_k)$ (in which case $\Psi(U_k) = \Psi_{max}$).

Proof: We will first show that the hypotheses and $\sim$(1) imply (2).

If $\{\Psi(U_i)\}_{i=1}$ is not strictly monotonic then there is some integer k for which $\Psi(HU_k) \leq \Psi(U_{k+1}) = \Psi(U_k)$ for each H in $\mathcal{H}$ and hence $\underset{\mathcal{H}}{\text{l.u.b.}}\ \Psi(HU_k) \leq \Psi(U_k)$. Now if $\Psi(U_k) < \Psi_{max}$ then for some $\hat{H}$ in $\mathcal{H}$

$\Psi(U_k) < \Psi(\hat{H}U_k) \leq \Psi_{max}$. This, of course, contradicts l.u.b. $\Psi(HU_k) \leq \Psi(U_k)$ and hence $\Psi(U_k) = \Psi_{max}$.

The hypothesis and $\sim$(2) imply, for each integer k, that there exists an H in $\mathcal{H}$ for which $\Psi(U_k) < \Psi(HU_k)$ and, for this H, that $\Psi(U_k) < \Psi(HU_k) \leq \Psi(U_{k+1})$. The convergence of $\{\Psi(U_i)\}_{i=1}^{\infty}$ to Ψ_{max} is a consequence of Proposition 2.

III. Application

We will first prove two propositions before doing the application.

Proposition 4. Let $T:R^n \to R^m$ be a linear transformation. Then there exists an orthonormal basis $\{v_1,\dots,v_m\}$ of R^m satisfying $\langle T(v_i),T(v_j)\rangle = 0$ whenever $1 \le i < j \le m$.

Proof: For $v \in R^m$ define $f(v) = \|T(v)\|^2$. Since f is continuous there exists v^* with $\|v^*\|=1$ and $f(v) \le f(v^*)$ whenever $\|v\|=1$. For v in R^m the partial of f in the direction v at v^* is $2\langle T(v^*),T(v)\rangle$. If $\langle v,v^*\rangle = 0$ and $\langle T(v^*),T(v)\rangle \ne 0$ then W.L.O.G. $\langle T(v^*),T(v)\rangle > 0$ from which we can deduce that there exists a,b with $a^2+b^2=1$ and for which $\|T(av+bv^*)\|^2 > \|T(v^*)\|^2$. This would contradict the choice of v^* and it follows that $\langle T(v),T(v^*)\rangle = 0$ whenever $\langle v,v^*\rangle=0$. Since the collection $[v^*]^\perp$ of all vectors v for which $\langle v,v^*\rangle=0$ is an m-1 dimensional subspace of R^m then by an inductive argument we can assume the existence of an orthonormal basis $\{v_1,\dots,v_{m-1}\}$ of $[v^*]^\perp$ satisfying $\langle T(v_i),T(v_j)\rangle = 0$ whenever $1 \le i < j \le m-1$. By letting $v_m = v^*$ then $\{v_1,\dots,v_m\}$ is the desired orthonormal basis of R^m. This completes the proof.

The collection of k-dimensional subspaces of R^n will be denoted by $\mathcal{S}_n^k$. Let $T:R^n \to R^n$ be a linear transformation. For K in $\mathcal{S}_n^k$ define $\Psi_T(K) = \sum_{i=1}^{k} \|T(v_i)\|^2$ where $\{v_1,\dots,v_k\}$ is an orthonormal basis for K. Ψ_T is well defined since

$$\Psi_T(K) = \sum_{i=1}^{k} \operatorname{trace}\{(Tv_i)(Tv_i)^T\} = \sum_{i=1}^{k} \operatorname{trace}\{(Tv_i)(v_i^T T^T)\}$$

$$= \sum_{i=1}^{k} \operatorname{trace}\{(T^T T)(v_i v_i^T)\} = \operatorname{trace}\{(T^T T)(\sum_{i=1}^{k} v_i v_i^T)\}$$ and since $\sum_{i=1}^{k} v_i v_i^T$ represents the projection P_K of R^n onto K. For K in $\mathcal{S}_n^k$ let $K^\perp$ denote the orthogonal complement of K in R^n. We will say that K is C1 w.r.t. T if $\langle T(x),T(y)\rangle = 0$ whenever $x \in K$ and $y \in K^\perp$. If $\|T(y)\| \le \|T(x)\|$ whenever $x \in K$, $y \in K^\perp$, $\|x\|=\|y\|=1$ then we will say that K is C2 w.r.t. T.

Proposition 5. Let $T: R^n \to R^n$ be a linear transformation and let K_o be in $\mathcal{S}_n^k$.

i) If K_o is C1 and C2 w.r.t. T then $\Psi_T(K) \le \Psi_T(K_o)$ for all K.

ii) If $\Psi_T(K_o) < \Psi_T(K)$ for some K in $\mathcal{S}_n^k$ then there exists H in $\mathcal{H}$ for which $\Psi_T(K_o) < \Psi_T(H(K_o))$.

Proof: For any K, K_o in $\mathcal{S}_n^k$ there exists by Proposition 4 an orthonormal basis $\{w_1, \ldots, w_k\}$ of K for which $\langle P_{K_o}(w_i), P_{K_o}(w_j)\rangle = 0$ whenever $1 \le i < j \le k$. W.L.O.G. we may assume that $P_{K_o}(w_i) \neq 0$ for $1 \le i \le p$ and $P_{K_o}(w_j) = 0$ for $p+1 \le j \le k$.

Let $v_i = \dfrac{P_{K_o}(w_i)}{\|P_{K_o}(w_i)\|}$ for $1 \le i \le p$. $\{v_1, \ldots, v_p\}$ is an orthonormal system in K_o which extends to an orthonormal basis $\{v_1, \ldots, v_k\}$ of K_o. For $1 \le i \le p$ there exists real numbers a_i, b_i satisfying $a_i^2 + b_i^2 = 1$ and a vector v_i^* in $K_o^\perp$ of norm 1 for which $w_i = a_i v_i + b_i v_i^*$. Suppose now that K_o is C1 and C2 w.r.t. T. Then

$$\|T(w_i)\|^2 = a_i^2\|T(v_i)\|^2 + b_i^2\|T(v_i^*)\|^2 + 2a_i b_i\langle T(v_i), T(v_i^*)\rangle$$
$$= a_i^2\|T(v_i)\|^2 + b_i^2\|T(v_i^*)\|^2 \le a_i^2\|T(v_i)\|^2 + b_i^2\|T(v_i)\|^2 = \|T(v_i)\|^2$$

for $1 \le i \le p$, and $\|T(w_j)\|^2 = \|T(v_j)\|^2$ for $p+1 \le j \le k$. It follows from this that $\Psi_T(K) \le \Psi_T(K_o)$ for any K in $\mathcal{S}_n^k$ whenever K_o is C1 and C2 w.r.t. T.

Suppose now that $\Psi_T(K_o) < \Psi_T(K)$. Then for some index i we must have $\|T(w_i)\| > \|T(v_i)\|$. Select H in $\mathcal{H}$ satisfying $H(v_i) = w_i$ and $H(v_j) = v_j$ whenever $1 \le j \le k$ and $j \ne i$. Such an H must exist since $\langle w_i, v_j\rangle = 0$ whenever $j \ne i$. $\{v_1, \ldots, v_{i-1}, w_i, v_{i+1}, \ldots, v_k\}$ is an orthonormal system spanning $H(K_o)$ and since $\|T(w_i)\| > \|T(v_i)\|$ then $\Psi_T(K_o) < \Psi_T(H(K_o))$. This completes the proof.

For a finite family of multivariate normal densities each of whose covariance matrix is the identity matrix, let $\{v_1, \ldots, v_m\}$ be the collection of pairwise differences of mean vectors for distinct pairs. For a k×n rank k matrix B let $\Psi(B)$ be the B-induced interclass divergence [1]. Then

$$\Psi(B) = \left[\sum_{i=1}^{m} \text{trace} \left\{ [BB^T]^{-1} B(I+v_i v_i^T) B^T \right\} \right] - mk.$$

It can be shown that there exists a unitary matrix U in $\mathcal{U}$ for which $\Psi(B) = \Psi([I_k|Z]U)$ where $[I_k|Z]$ is the kxn matrix whose i^{th} row equals e_i^T. For that U we can determine that

$\Psi(B) = \sum_{i=1}^{m} \|P_k(U(v_i))\|^2$ where P_k is the projection of R^n onto the span of the vectors $\{e_1, \ldots, e_k\}$. Let $\mathcal{V}$ be the mxn matrix whose i^{th} row is v_i^T for $1 \le i \le m$. Since

$\|P_k(U(v_i))\|^2 = \langle U^T(e_1), v_i \rangle^2 + \cdots + \langle U^T(e_k), v_i \rangle^2$ then

$\Psi(B) = \sum_{i=1}^{k} \|\mathcal{V}(U^T(e_i))\|^2$. If we let K_0 be the span of $\{e_1, \ldots, e_k\}$ then $\Psi(B) = \Psi_{\mathcal{V}}(U^T(K_0))$. There fore maximizing interclass divergence in this case can be done by maximizing the function $U \to \Psi_{\mathcal{V}}(U(K_0))$ on $\mathcal{U}$. Proposition 5 implies that this function is $\mathcal{H}$-sloped.

REFERENCES

1. Decell, H. P. and Smiley, W. G. III, Optimal Linear Combinations Using Householder Transformations, Comm. in Statistics A, Vol 5, #12, 1976, pp. 1153-1162.
2. Royden, H. L., Real Analysis, page 148, 1970, Macmillan Company, London.
3. Anderson, T. W., An Introduction to Multivariate Statistical Ahalysis, 1958 John Wiley and Sons, Inc., New York.
4. Kullback, Solomon, Information Theory and Statistics, 1968 Dover Publications, New York.
5. Decell, H. P. and Quirein, J. A., An Iterative Approach to the Feature Selection Problem, Proceedings of IEEE Conference on Machine Processing of Remotelly Sensed Data, 1973, IEEE Cat. #CH0834-2-GE, pp. 3B-1--3B-12.
6. Decell, H. P. and Marini, S., Feature Combinations and the Bhattacharya Criterion, Comm. in Stat A., Vol. 5, 1976, #12, pp. 1143-1152.
7. Decell, H. P. and Mayekar, S., Feature Combinations and the Divergence Criterion, Computers and Mathematics with Applications Vol. 3, 1976, pp. 71-76.
8. Householder, Alston S., The Theory of Matrices in Numerical Analysis, 1964 Dover Publications, New York.