

AgRISTARS

SR-P0-00469
NAS9-15466

A Joint Program for
Agriculture and
Resources Inventory
Surveys Through
Aerospace
Remote Sensing

Supporting Research

July 1980

Technical Report

A Multispectral Data Simulation Technique

by Marwan J. Muasher and Philip H. Swain

Purdue University
Laboratory for Applications of Remote Sensing
West Lafayette, Indiana 47907



NASA



80-10311
CR-163379

SR-P0-00469
NAS9-15466

AgRISTARS

"Made available under NASA sponsorship
in the interest of early and wide dis-
semination of Earth Resources Survey
Program information and without liability
for any use made thereof."

A Joint Program for
Agriculture and
Resources Inventory
Surveys Through
Aerospace
Remote Sensing

Supporting Research

July 1980

Technical Report

A Multispectral Data Simulation Technique

by Marwan J. Muasher and Philip H. Swain

(E80-10311) A MULTISPECTRAL DATA SIMULATION
TECHNIQUE (Purdue Univ.) 27 p HC A03/MF A01
CSCL 05B

N80-33830

Unclassified
G3/43 00311

Purdue University
Laboratory for Applications of Remote Sensing
West Lafayette, Indiana 47907



NASA



SR-PO-00469

NAS9-15466

TECHNICAL REPORT

A MULTISPECTRAL DATA SIMULATION TECHNIQUE

By

M. J. Muasher

and

P. H. Swain

This report describes activity carried out
in the Supporting Research Project.

PURDUE UNIVERSITY

LABORATORY FOR APPLICATIONS OF REMOTE SENSING

1220 Potter Drive

WEST LAFAYETTE, INDIANA 47906, U.S.A.

JULY 1980

Star Information Form

1. Report No. SR-PO-00469	2. Government Accession No.	3. Recipient's Catalog No.	
4. Title and Subtitle A MULTISPECTRAL DATA SIMULATION TECHNIQUE		5. Report Date July 1980	6. Performing Organization Code
7. Author(s) Marwan J. Muasher and Philip H. Swain		8. Performing Organization Report No. 070980	10. Work Unit No.
9. Performing Organization Name and Address Laboratory for Applications of Remote Sensing Purdue University West Lafayette, IN 47906		11. Contract or Grant No. NAS9-15466	13. Type of Report and Period Covered
12. Sponsoring Agency Name and Address National Aeronautics and Space Administration Johnson Space Center Houston, TX 77058 Tech. Monitor: Richard Heydorn		14. Sponsoring Agency Code	
15. Supplementary Notes			
16. Abstract <p>In remote sensing data analysis, several assumptions are made that are not always precisely met. These assumptions include that the classes in the data are normally distributed, that training data are representative of the area of interest, that the number of classes is known, and that all pixels are pure.</p> <p>In testing new algorithms, deviations from the assumptions may obscure the action of the new process. One way to clarify the situation is to apply the algorithm first to a data set satisfying the assumptions.</p> <p>A method is presented to obtain an artificial data set through simulation. While retaining the natural spatial and spectral information in the scene by basing the simulation on a classification, the data set provides the analyst with an exact number of classes in the scene, true distributions of these classes, independent measurements and "pure" pixels.</p> <p>Program listings in both Fortran and C-Language are provided in the appendices.</p>			
17. Key Words (Suggested by Author(s)) Data Simulation Multispectral Data		18. Distribution Statement	
19. Security Classif. (of this report) Unclassified	20. Security Classif. (of this page) Unclassified	21. No. of Pages	22. Price*

A MULTISPECTRAL DATA SIMULATION TECHNIQUE*

Marwan J. Muasher and Philip H. Swain

For remote sensing data analysis, several assumptions are commonly made. These assumptions are usually that the data are class-conditionally distributed multivariate normal and that the data used to train the classifier are representative of the area of interest. This second assumption actually has several parts. The assumption is made that in the process of training, all classes present in the scene are found, and all spectral subclasses of each class are also represented in the training data. Furthermore, the parameters of the distribution of each subclass are also assumed to be known from the training data. Each pixel is assumed to come from one of the training classes, and also is assumed to be entirely of one cover type.

In actual practice, these assumptions are not met. The number of spectral classes in the area is not known and clustering or some other method is used to determine the number of subclasses, in addition to estimating the statistics of those subclasses. Some of these methods also lead to non-normal subclasses. In particular, the clustering algorithm available through LARSYS truncates the tails of the subclass distributions and so leads to non-normal distributions.

There are also questions relating to a single picture element. A single pixel in Landsat data covers an area approximately 80 meters by 50 meters. More than one cover type may be present in this area and result in a "mixture pixel" observation. It is not clear how the distribution of the spectral response of mixture pixels can be related to the distribution of the spectral response of "pure pixels."

*This work was sponsored by NASA Contract NAS9-15466.

There has been much speculation in the remote sensing community as to the effect of the non-satisfaction of the basic assumptions. Whenever new algorithms are brought forth, the old questions are raised again, indicating that there is insufficient understanding of the interaction of the real attributes of the data and the theory of the algorithms. At times it is not clear whether a particular result is due to aspects of the algorithm or to the extent the data set deviates from the assumptions.

In testing new algorithms, deviations from the assumptions may obscure the action of the new process. One way to clarify the situation is to apply the algorithm first to a data set satisfying the assumptions.

Such a data set could be obtained artificially, through simulation. The analyst could then know: how many classes exist in the data; the true distributions of the classes, including normality if desired; the observations could really be independent; and no pixel would be a "mixture pixel." New algorithms could be studied on such a data set with the knowledge that any "strange" effects are indeed algorithm rather than data problems.

In many cases where simulated data have been used in the past, the data were too artificial, in the sense that all aspects of the image were controlled, removing the natural variation in object size, position, and relationship which occur in real data. This limited the use of the simulated data sets in testing new algorithms.

The natural spatial information occurring in multispectral data could be retained in a simulated image by spatially basing the simulation on a classification. It would be even better to base the simulated data on a digitized "ground truth" map if the spectral characteristics of the cover types were known. By basing the simulation on a classification, the number of classes, their exact distributions, and the class of each pixel in the

area are known. If the classification was sufficiently accurate, then the spatial information held in the classification map will be close to the actual cover type map and actual spatial content of the original data. For each pixel in the area, a random vector distributed according to the pixel's class statistics could be generated. This becomes the simulated data vector.

Statistical Background

From the classification chosen as a basis for the simulation, the following are known: the number of classes K, the set of classes $\{\omega_i, i=1, \dots, K\}$, the class distributions $\{f(\omega_i), i=1, \dots, K\}$, their means and covariances $\{\mu_i\}$ and $\{\Sigma_i, i=1, \dots, K\}$, the number of channels p, and the class of every pixel in the scene.

From classifical statistics:

- (1) Let $X:p_{xl}$, $A:p_{xp}$ and $b:p_{xl}$.

If $X \sim N(0, I_p)$, then $Y = AX + b \sim N(b, A I_p A^T = A A^T)$
(where I_p is the identity matrix having dimensionality p).

- (2) Let Σ be a symmetric, positive definite matrix. Then there exists A, such that

$$A A^T = \Sigma \quad (A \text{ is denoted } \Sigma^{\frac{1}{2}})$$

To simulate a pixel which was a member of class i in the base classification, $N(0, I_p)$ (the random vector for each pixel is indepedent of other vectors) is generated. (See Appendix I.) Next $Y = \Sigma_i^{\frac{1}{2}} X + \mu_i$ is calculated; it is then a random vector from the population $N(\mu_i, \Sigma_i)$. This process is repeated for each pixel of the base classification and the random vectors thus generated are stored appropriately, i.e., so as to correspond to their simulated spatial location.

The program requires as an input a classification map stored on a results tape. The results tape has the class statistics

for p-dimensions also stored on it. The program, then, uses the results map and the stored statistics to generate a p-dimensional data set, which is stored on a user specified output tape in LARSYS format.

Appendix I provides a mathematical derivation related to the generation of normally distributed samples. Appendix II provides the Fortran program listing for the simulation program. Appendix III provides the C program listing for the same program.

APPENDIX I

APPENDIX I

Let U_1 and U_2 be two random variables independent and identically distributed Uniform $(0,1)$.

Then let $Z_1 = (-2 \ln U_1)^{\frac{1}{2}} \cos 2\pi U_2$

and $Z_2 = (-2 \ln U_1)^{\frac{1}{2}} \sin 2\pi U_2$

then Z_1 and Z_2 are independent and identically distributed normal $(0,1)$.

Proof:

$$f(U_1, U_2) = \begin{cases} 1 & 0 < U_1 < 1, 0 < U_2 < 1 \\ 0 & \text{otherwise} \end{cases}$$

is the probability density function of two independent uniforms.

$$U_1 = \exp[-\frac{1}{2}(Z_1^2 + Z_2^2)]$$

$$U_2 = \frac{1}{2\pi} \arctan\left(\frac{Z_2}{Z_1}\right)$$

The Jacobian of the transformation is:

$$J = -\frac{1}{2\pi} \exp[-\frac{1}{2}(Z_1^2 + Z_2^2)]$$

$$f(Z_1, Z_2) = f(U_1, U_2) \cdot |J|$$

$$= \frac{1}{2\pi} \exp[-\frac{1}{2}(Z_1^2 + Z_2^2)] \quad 0 < [\exp -\frac{1}{2}(Z_1^2 + Z_2^2)] < 1$$

$$0 < \frac{1}{2\pi} \arctan\left(\frac{Z_2}{Z_1}\right) < 1$$

$$= 0 \quad \text{otherwise}$$

$$\therefore f(Z_1) \sim N(0,1) \quad f(Z_2) \sim N(0,1)$$

The side conditions give $-\infty < z_1 < \infty$, $-\infty < z_2 < \infty$. Strictly speaking, z_1 cannot equal zero; however, $\text{prob}(z_1 = 0) = 0$ as we are working with continuous densities.

To test the effectiveness of the pseudo random vectors in the multivariate case, random vectors distributed $N(0, I_p)$ were generated and then tested with a Kolmogorov-Smirnov test. Since the multivariate normal cdf is difficult to evaluate, the sum of squares was calculated and compared to the χ_p^2 distribution.

For sample sizes greater than 100, the pseudo random vectors were distributed properly. For sample sizes less than 100, the K-S test is not valid. Since we would generally (over an entire area) be working with more than 100 points per class, this was not pursued further.

In addition, the sample covariance matrices were tested for homogeneity against the true class statistics. For sample runs of up to 2000 points, there were not significant differences at the $\alpha = 0.10$ level.

APPENDIX II

FILE: SWRITE FORTRAN A PURDUE / LARS 3031

```
DO 100 IP=1,NOCLAS          SWRC03170
DO 100 IO=1,NOCHAN          SWRC3180
IF(CLAPNT(IP).LE.0) GO TO 98  SWRC03190
RMEAN(IP,IO)=FLOAT(IMEAN(IP,IO))/FLOAT(CLAPNT(IP)) SWRC3200
98 DO 100 IT=1,NOCHAN        SWRC03210
IF(CLAPNT(IP).LE.1) GO TO 100  SWRC3220
REPNT=FLOAT(CLAPNT(IP))      SWRC3230
REVAR=FLOAT(IVAR(IP,IO,IT))   SWRC3240
REMEAN=FLOAT(IMEAN(IP,IO))    SWRC03250
SEMEAN=FLOAT(IMEAN(IP,IT))   SWRC03260
RVAR(IP,IO,IT)=1./REPNT-1.0*(REVAR-REMEAN+SEMEAN/REPNT) SWRC03270
RVAR(IP,IT,IO)=RVAR(IP,IO,IT) SWRC03280
100 CONTINUE                  SWRC3290
DO 645 IP=1,NOCLAS          SWRC3300
WRITE(6,605)IP,CLAPNT(IP)    SWRC3310
605 FORMAT(IH1/5X,'CLASS NUMBER',I3,5X,I8,' POINTS'///) SWRC3320
WRITE(6,610)                 SWRC3330
610 FORMAT(37X,'ACTUAL',4X,'SIMULATED') SWRC03340
WRITE(6,615)                 SWRC3350
615 FORMAT(38X,'MEAN',7X,'MEAN') SWRC3360
DO 622 IX=1,NOCHAN          SWRC3370
NINC=NOCOMP+NOCLAS+(IP-1)*NOCHAN SWRC03380
WRITE(6,620)FETVC3(IX),FRQCAL(1,FETVC3(IX)),FETVC3(IX),Z$ SWRC03390
$2(NINC+IX),RMEAN(IP,IX)      SWRC03400
620 FORMAT(5X,'CHANNEL',I3,2X,'(',F5.2,'-',F5.2,')',5X,F8.3,3X,F8.3) SWRC3410
622 CONTINUE                  SWRC3420
WRITE(6,625)                 SWRC3430
625 FORMAT(//5X,'ACTUAL COVARIANCE MATRIX') SWRC3440
DO 630 NO=1,NOCOMP          SWRC3450
NINC=(IP-1)*NOCOMP          SWRC3460
630 A(:NO)=Z2(NINC+NO)       SWRC3470
CALL WRTMTX(A,NOCHAN,FRQCAL,THREE,FETVC3) SWRC3480
WRITE(6,635)                 SWRC3490
635 FORMAT(//5X,'SIMULATED COVARIANCE MATRIX') SWRC03500
NO=C                         SWRC3510
DO 640 IO=1,NOCHAN          SWRC3520
DO 640 IN=1,IO               SWRC3530
NO=NO+1                      SWRC3540
640 A(:NO)=RVAR(IP,IO,IN)    SWRC3550
CALL WRTMTX(A,NOCHAN,FRQCAL,THREE,FETVC3) SWRC3560
645 CONTINUE                  SWRC3570
CALL TOPEF(12,IER)           SWRC3580
650 IDREC(IX)=0              SWRC03590
CALL TOPWR(12,80C,IER,IDREC) SWRC03600
IF(IER.NE.0) WRITE(16,234)IER SWRC3610
IF(IER.GT.0) GO TO 310       SWRC3620
GO TO 320                   SWRC3640
234 FORMAT(5X,'ERROR IS',I5) SWRC3650
C***** ERROR MESSAGES ***** SWRC3660
C 300 WRITE(6,305)            SWRC3670
305 FORMAT(5X,'ERROR -1')     SWRC3680
310 WRITE(6,315)              SWRC3690
315 FORMAT(5X,'ERROR GT 1')   SWRC3700
320 STOP                      SWRC3740
END                         SWRC3750
SWRC3760
```

FILE: SWRITE FORTRAN 4 PURDUE / LARS 3031

```

C 150 CONTINUE
C
      CALL TOPWR(12,800,IER,IREC)
      IF(IER.NE.0) WRITE(16,234) IER
      IF(IER.GT.0) GO TO 31C
      DO 50 MA=1,NOCLAS
      CLAPNT:MAI=0
      DO 50 MB=1,NOCHAN
      IMEAN:MA,MB)=0
      RMEAN:MA,MB)=0.0
      DO 50 MC=1,NOCHAN
      IVAR:MA,MB,MC)=0
      50 RVAR:MA,MB,MC)=C.0
      LNWRIT=0
      55 READ(11) J,K,LINENO,(PNTCLS:IX),IX=1,NCPNTS
      IF(J.GT.6) GO TO 95
      LNWRIT=LNWRIT+1
      IF(MOD(LNWRIT,25).EQ.0) WRITE(16,57) LNWRIT,NOLINE
      57 FORMAT(5X,I4,' LINES OUT OF ',I4,' ARE COMPLETED')
C *****
C GENERATE AND WRITE DATA POINTS
C *****
C
      60 IZ=IL IN(2)
      DATOUT:1)=L1:1
      DATOUT:2)=L1:2
      IZ=32767
      DATOUT:3)=L1:1
      DATOUT:4)=L1:2
      IZ=0
      ICOUNT=4
      DO 90 IX=1,NOPNTS
      ICOUNT=ICOUNT+1
      IZ=PNTCLS:IX
      L1:1)=.FALSE.
      IPOL=;IZ-1)*NOCHAN
      IBEG=;IZ-1)*NOCOMP
      K=IBEG
      DO 65 IY=1,NOCHAN
      DO 65 IZ=1,IY
      K=K+1
      B:IY,IZ)=Z(K)
      IF(IY.EQ.IZ) GO TO 65
      B:IZ,IY)=0.0
      65 CONTINUE
      DO 70 IY=1,NOCH
      CALL RANDU;ISTART:IY),NXINP,A2:IY)
      ISTART:IY)=NXINP
      CALL RANDU;ISTART:IY),NXINP,A:IY)
      ISTART:IY)=NXINP
      A:IY)=SQRT(-2.* ALOG(A2:IY)*COS(6.28318*A:IY))
      70 CONTINUE
      CLAPNT:IZ)=CLAPNT:IZ+1
      DO 80 IY=1,NOCHAN
      DATA:IY)=0.0
      IQ=NOPOL*NOCOMP+IPOL+IY
      DO 75 IZ=1,NOCHAN
      DATA:IY)=DATA:IY)+B:IY,IZ)*A:IZ)
      DATA:IY)=DATA:IY)+Z:IQ)
      INTDAT=DATA:IY)+.5
      IF(INTDAT.LT.0) INTDAT=0
      IF(INTDAT.GT.255) INTDAT=255
      ISTAT:IY)=INTDAT
      DATOUT:(IY-1)*NOSAM+ICOUNT)=LOGDAT:2)
      DO 92 IZ=1,6
      DATOUT:(IY-1)*NOSAM+ICOUNT+IZ)=.FALSE.
      92 CONTINUE
      DO 90 II=1,NOCHAN
      IMEAN:I2,II)=IMEAN:I2,II)+ISTAT:II)
      DO 90 JJ=1,NOCHAN
      IVAR:I2,II,JJ)=IVAR:I2,II,JJ)+ISTAT:II)+ISTAT:JJ)
      90 CONTINUE
      NOBYTE=4*NOCHAN*NOSAM
      CALL TOPWR(12,NOBYTE,IER,DATOUT)
      IF(IER.NE.0) WRITE(16,234) IER
      IF(IER.GT.0) GO TO 31C
      GO TO 55
      95 CONTINUE

```

SWR C2380
SWR C2390
SWR C2400
SWR C2410
SWR C2420
SWR C2430
SWR C2440
SWR C2450
SWR C2460
SWR C2470
SWR C2480
SWR C2490
SWR C2500
SWR C2510
SWR C2520
SWR C2530
SWR C2540
SWR C2550
SWR C2560
SWR C2570
SWR C2580
SWR C2590
SWR C2600
SWR C2610
SWR C2620
SWR C2630
SWR C2640
SWR C2650
SWR C2660
SWR C2670
SWR C2680
SWR C2690
SWR C2700
SWR C2710
SWR C2720
SWR C2730
SWR C2740
SWR C2750
SWR C2760
SWR C2770
SWR C2780
SWR C2790
SWR C2800
SWR C2810
SWR C2820
SWR C2830
SWR C2840
SWR C2850
SWR C2860
SWR C2870
SWR C2880
SWR C2890
SWR C2900
SWR C2910
SWR C2920
SWR C2930
SWR C2940
SWR C2950
SWR C2960
SWR C2970
SWR C2980
SWR C2990
SWR C3000
SWR C3010
SWR C3020
SWR C3030
SWR C3040
SWR C3050
SWR C3060
SWR C3070
SWR C3080
SWR C3090
SWR C3100
SWR C3110
SWR C3120
SWR C3130
SWR C3140
SWR C3150
SWR C3160

FILE: SWRITE FORTRAN 4 PURDUE / LARS 3031

```

IBEG=1          SWR01590
C               SWR01600
C               SWR01610
C FACTOR COVARIANCE MATRICES           SWR01620
C               SWR01630
C               SWR01640
C               SWR01650
C               SWR01660
C               SWR01670
C               SWR01680
C               SWR01690
C               SWR01700
C               SWR01710
C               SWR01720
C               SWR01730
C               SWR01740
C               SWR01750
C               SWR01760
C               SWR01770
C               SWR01780
C               SWR01790
C               SWR01800
C               SWR01810
C               SWR01820
C               SWR01830
C               SWR01840
C               SWR01850
C               SWR01860
C               SWR01870
C               SWR01880
C               SWR01890
C               SWR01900
C               SWR01910
C               SWR01920
C               SWR01930
C               SWR01940
C               SWR01950
C               SWR01960
C               SWR01970
C               SWR01980
C               SWR01990
C               SWR02000
C               SWR02010
C               SWR02020
C               SWR02030
C               SWR02040
C               SWR02050
C               SWR02060
C               SWR02070
C               SWR02080
C               SWR02090
C               SWR02100
C               SWR02110
C               SWR02120
C               SWR02130
C               SWR02140
C               SWR02150
C               SWR02160
C               SWR02170
C               SWR02180
C               SWR02190
C               SWR02200
C               SWR02210
C               SWR02220
C               SWR02230
C               SWR02240
C               SWR02250
C               SWR02260
C               SWR02270
C               SWR02280
C               SWR02290
C               SWR02300
C               SWR02310
C               SWR02320
C               SWR02330
C               SWR02340
C               SWR02350
C               SWR02360
C               SWR02370

DO 30 IX=1,NOPOLL          SWR01590
IDONE=IBEG+NOCOMP-1        SWR01600
K=0                         SWR01610
DO 20 IY=IBEG, IDONE       SWR01620
K=K+1                       SWR01630
20 A(K)=Z(IY)              SWR01640
CALL MFSDA, NOCHAN, EPS, IER  SWR01650
IF; IER.EQ.-1, GO TO 300    SWR01660
IF; IER.GE.11 GO TO 310    SWR01670
K=0                         SWR01680
DO 25 IY=IBEG, IDONE       SWR01690
K=K+1                       SWR01700
25 Z(IY)=A(K)              SWR01710
30 IBEG=IBEG+NOCOMP        SWR01720
C               SWR01730
C GENERATE STARTING POINTS  SWR01740
C               SWR01750
C               SWR01760
C               SWR01770
C               SWR01780
C               SWR01790
C               SWR01800
C               SWR01810
C               SWR01820
C               SWR01830
C               SWR01840
C               SWR01850
C               SWR01860
C               SWR01870
C               SWR01880
C               SWR01890
C               SWR01900
C               SWR01910
C               SWR01920
C               SWR01930
C               SWR01940
C               SWR01950
C               SWR01960
C               SWR01970
C               SWR01980
C               SWR01990
C               SWR02000
C               SWR02010
C               SWR02020
C               SWR02030
C               SWR02040
C               SWR02050
C               SWR02060
C               SWR02070
C               SWR02080
C               SWR02090
C               SWR02100
C               SWR02110
C               SWR02120
C               SWR02130
C               SWR02140
C               SWR02150
C               SWR02160
C               SWR02170
C               SWR02180
C               SWR02190
C               SWR02200
C               SWR02210
C               SWR02220
C               SWR02230
C               SWR02240
C               SWR02250
C               SWR02260
C               SWR02270
C               SWR02280
C               SWR02290
C               SWR02300
C               SWR02310
C               SWR02320
C               SWR02330
C               SWR02340
C               SWR02350
C               SWR02360
C               SWR02370

29 WRITE(16,31)          SWR01590
31 FORMAT(5X,'DO YOU WANT TO SPECIFY THE STARTING POINTS FOR THE //5X') SWR01600
$, 'RANDOM NUMBER GENERATOR? (TYPE YES OR NO)' SWR01610
READ(16,32)INPUT          SWR01620
32 FORMAT(4I1)             SWR01630
IF; INPUT.EQ.NO! GO TO 36  SWR01640
IF; INPUT.EQ.YES! GO TO 33  SWR01650
GO TO 29                  SWR01660
33 DO 39 IX=1,NOCHAN      SWR01670
WRITE(16,41)IX            SWR01680
41 FORMAT(5X,'SPECIFY STARTING POINT FOR CHANNEL ',I3/5X,'(TYPE A NINE') SWR01690
$, 'DIGIT ODD NUMBER)')     SWR01700
READ(16,42)ISTART(IX)      SWR01710
42 FORMAT(19)              SWR01720
39 CONTINUE                SWR01730
GO TO 43                  SWR01740
36 CALL GTSERL:ISERL       SWR01750
ISERL=:ISERL/10)*8+1       SWR01760
DO 40 I=1,NOCH             SWR01770
ISERL=ISERL+10000000       SWR01780
ISTART(I)=ISERL            SWR01790
40 CONTINUE                SWR01800
43 WRITE(6,34)             SWR01810
34 FORMAT(//5X,'STARTING POINTS FOR RANDOM NUMBER GENERATOR'//) SWR01820
DO 44 I=1,NOCHAN          SWR01830
WRITE(6,35)I,ISTART(I)     SWR01840
35 FORMAT(5X,'STARTING POINT FOR CHANNEL ',I2,' IS ',I9) SWR01850
44 CONTINUE                SWR01860
C               SWR01870
C READ CLASSIFICATIONS      SWR01880
C               SWR01890
C               SWR01900
C               SWR01910
C               SWR01920
C               SWR01930
C               SWR01940
C               SWR01950
C               SWR01960
C               SWR01970
C               SWR01980
C               SWR01990
C               SWR02000
C               SWR02010
C               SWR02020
C               SWR02030
C               SWR02040
C               SWR02050
C               SWR02060
C               SWR02070
C               SWR02080
C               SWR02090
C               SWR02100
C               SWR02110
C               SWR02120
C               SWR02130
C               SWR02140
C               SWR02150
C               SWR02160
C               SWR02170
C               SWR02180
C               SWR02190
C               SWR02200
C               SWR02210
C               SWR02220
C               SWR02230
C               SWR02240
C               SWR02250
C               SWR02260
C               SWR02270
C               SWR02280
C               SWR02290
C               SWR02300
C               SWR02310
C               SWR02320
C               SWR02330
C               SWR02340
C               SWR02350
C               SWR02360
C               SWR02370

IDREC(1)=TAPEND          SWR02180
IDREC(2)=JFILE            SWR02190
IDREC(3)=RUNNO            SWR02200
NOLD = IDREC(5)           SWR02210
IDREC(5)= NOCHAN          SWR02220
IDREC(6)= 4*(NOPNTS + S1/4) SWR02230
NOSAM = IDREC(6)          SWR02240
IDREC(7)= FLGT             SWR02250
DO 141 II=1,3              SWR02260
IDREC(II+16)= DATE(II)    SWR02270
141 CONTINUE                SWR02280
IDREC(20)= NOLINE          SWR02290
DO 145 II= 1, NOCHAN       SWR02300
INEW = FETVC3(II)          SWR02310
DO 145 II2 = 1,5            SWR02320
FRQCAL(II2,II)= FRQCAL(II2,INEW) SWR02330
C 145 CONTINUE                SWR02340
C               SWR02350
C               SWR02360
C               SWR02370
C               SWR02380
C               SWR02390
C               SWR02400
C               SWR02410
C               SWR02420
C               SWR02430
C               SWR02440
C               SWR02450
C               SWR02460
C               SWR02470
C               SWR02480
C               SWR02490
C               SWR02500
C               SWR02510
C               SWR02520
C               SWR02530
C               SWR02540
C               SWR02550
C               SWR02560
C               SWR02570
C               SWR02580
C               SWR02590
C               SWR02600
C               SWR02610
C               SWR02620
C               SWR02630
C               SWR02640
C               SWR02650
C               SWR02660
C               SWR02670
C               SWR02680
C               SWR02690
C               SWR02700
C               SWR02710
C               SWR02720
C               SWR02730
C               SWR02740
C               SWR02750
C               SWR02760
C               SWR02770
C               SWR02780
C               SWR02790
C               SWR02800
C               SWR02810
C               SWR02820
C               SWR02830
C               SWR02840
C               SWR02850
C               SWR02860
C               SWR02870
C               SWR02880
C               SWR02890
C               SWR02900
C               SWR02910
C               SWR02920
C               SWR02930
C               SWR02940
C               SWR02950
C               SWR02960
C               SWR02970
C               SWR02980
C               SWR02990
C               SWR03000
C               SWR03010
C               SWR03020
C               SWR03030
C               SWR03040
C               SWR03050
C               SWR03060
C               SWR03070
C               SWR03080
C               SWR03090
C               SWR03100
C               SWR03110
C               SWR03120
C               SWR03130
C               SWR03140
C               SWR03150
C               SWR03160
C               SWR03170
C               SWR03180
C               SWR03190
C               SWR03200
C               SWR03210
C               SWR03220
C               SWR03230
C               SWR03240
C               SWR03250
C               SWR03260
C               SWR03270
C               SWR03280
C               SWR03290
C               SWR03300
C               SWR03310
C               SWR03320
C               SWR03330
C               SWR03340
C               SWR03350
C               SWR03360
C               SWR03370
C               SWR03380
C               SWR03390
C               SWR03400
C               SWR03410
C               SWR03420
C               SWR03430
C               SWR03440
C               SWR03450
C               SWR03460
C               SWR03470
C               SWR03480
C               SWR03490
C               SWR03500
C               SWR03510
C               SWR03520
C               SWR03530
C               SWR03540
C               SWR03550
C               SWR03560
C               SWR03570
C               SWR03580
C               SWR03590
C               SWR03600
C               SWR03610
C               SWR03620
C               SWR03630
C               SWR03640
C               SWR03650
C               SWR03660
C               SWR03670
C               SWR03680
C               SWR03690
C               SWR03700
C               SWR03710
C               SWR03720
C               SWR03730
C               SWR03740
C               SWR03750
C               SWR03760
C               SWR03770
C               SWR03780
C               SWR03790
C               SWR03800
C               SWR03810
C               SWR03820
C               SWR03830
C               SWR03840
C               SWR03850
C               SWR03860
C               SWR03870
C               SWR03880
C               SWR03890
C               SWR03900
C               SWR03910
C               SWR03920
C               SWR03930
C               SWR03940
C               SWR03950
C               SWR03960
C               SWR03970
C               SWR03980
C               SWR03990
C               SWR04000
C               SWR04010
C               SWR04020
C               SWR04030
C               SWR04040
C               SWR04050
C               SWR04060
C               SWR04070
C               SWR04080
C               SWR04090
C               SWR04100
C               SWR04110
C               SWR04120
C               SWR04130
C               SWR04140
C               SWR04150
C               SWR04160
C               SWR04170
C               SWR04180
C               SWR04190
C               SWR04200
C               SWR04210
C               SWR04220
C               SWR04230
C               SWR04240
C               SWR04250
C               SWR04260
C               SWR04270
C               SWR04280
C               SWR04290
C               SWR04300
C               SWR04310
C               SWR04320
C               SWR04330
C               SWR04340
C               SWR04350
C               SWR04360
C               SWR04370
C               SWR04380
C               SWR04390
C               SWR04400
C               SWR04410
C               SWR04420
C               SWR04430
C               SWR04440
C               SWR04450
C               SWR04460
C               SWR04470
C               SWR04480
C               SWR04490
C               SWR04500
C               SWR04510
C               SWR04520
C               SWR04530
C               SWR04540
C               SWR04550
C               SWR04560
C               SWR04570
C               SWR04580
C               SWR04590
C               SWR04600
C               SWR04610
C               SWR04620
C               SWR04630
C               SWR04640
C               SWR04650
C               SWR04660
C               SWR04670
C               SWR04680
C               SWR04690
C               SWR04700
C               SWR04710
C               SWR04720
C               SWR04730
C               SWR04740
C               SWR04750
C               SWR04760
C               SWR04770
C               SWR04780
C               SWR04790
C               SWR04800
C               SWR04810
C               SWR04820
C               SWR04830
C               SWR04840
C               SWR04850
C               SWR04860
C               SWR04870
C               SWR04880
C               SWR04890
C               SWR04900
C               SWR04910
C               SWR04920
C               SWR04930
C               SWR04940
C               SWR04950
C               SWR04960
C               SWR04970
C               SWR04980
C               SWR04990
C               SWR05000
C               SWR05010
C               SWR05020
C               SWR05030
C               SWR05040
C               SWR05050
C               SWR05060
C               SWR05070
C               SWR05080
C               SWR05090
C               SWR05100
C               SWR05110
C               SWR05120
C               SWR05130
C               SWR05140
C               SWR05150
C               SWR05160
C               SWR05170
C               SWR05180
C               SWR05190
C               SWR05200
C               SWR05210
C               SWR05220
C               SWR05230
C               SWR05240
C               SWR05250
C               SWR05260
C               SWR05270
C               SWR05280
C               SWR05290
C               SWR05300
C               SWR05310
C               SWR05320
C               SWR05330
C               SWR05340
C               SWR05350
C               SWR05360
C               SWR05370
C               SWR05380
C               SWR05390
C               SWR05400
C               SWR05410
C               SWR05420
C               SWR05430
C               SWR05440
C               SWR05450
C               SWR05460
C               SWR05470
C               SWR05480
C               SWR05490
C               SWR05500
C               SWR05510
C               SWR05520
C               SWR05530
C               SWR05540
C               SWR05550
C               SWR05560
C               SWR05570
C               SWR05580
C               SWR05590
C               SWR05600
C               SWR05610
C               SWR05620
C               SWR05630
C               SWR05640
C               SWR05650
C               SWR05660
C               SWR05670
C               SWR05680
C               SWR05690
C               SWR05700
C               SWR05710
C               SWR05720
C               SWR05730
C               SWR05740
C               SWR05750
C               SWR05760
C               SWR05770
C               SWR05780
C               SWR05790
C               SWR05800
C               SWR05810
C               SWR05820
C               SWR05830
C               SWR05840
C               SWR05850
C               SWR05860
C               SWR05870
C               SWR05880
C               SWR05890
C               SWR05900
C               SWR05910
C               SWR05920
C               SWR05930
C               SWR05940
C               SWR05950
C               SWR05960
C               SWR05970
C               SWR05980
C               SWR05990
C               SWR06000
C               SWR06010
C               SWR06020
C               SWR06030
C               SWR06040
C               SWR06050
C               SWR06060
C               SWR06070
C               SWR06080
C               SWR06090
C               SWR06100
C               SWR06110
C               SWR06120
C               SWR06130
C               SWR06140
C               SWR06150
C               SWR06160
C               SWR06170
C               SWR06180
C               SWR06190
C               SWR06200
C               SWR06210
C               SWR06220
C               SWR06230
C               SWR06240
C               SWR06250
C               SWR06260
C               SWR06270
C               SWR06280
C               SWR06290
C               SWR06300
C               SWR06310
C               SWR06320
C               SWR06330
C               SWR06340
C               SWR06350
C               SWR06360
C               SWR06370
C               SWR06380
C               SWR06390
C               SWR06400
C               SWR06410
C               SWR06420
C               SWR06430
C               SWR06440
C               SWR06450
C               SWR06460
C               SWR06470
C               SWR06480
C               SWR06490
C               SWR06500
C               SWR06510
C               SWR06520
C               SWR06530
C               SWR06540
C               SWR06550
C               SWR06560
C               SWR06570
C               SWR06580
C               SWR06590
C               SWR06600
C               SWR06610
C               SWR06620
C               SWR06630
C               SWR06640
C               SWR06650
C               SWR06660
C               SWR06670
C               SWR06680
C               SWR06690
C               SWR06700
C               SWR06710
C               SWR06720
C               SWR06730
C               SWR06740
C               SWR06750
C               SWR06760
C               SWR06770
C               SWR06780
C               SWR06790
C               SWR06800
C               SWR06810
C               SWR06820
C               SWR06830
C               SWR06840
C               SWR06850
C               SWR06860
C               SWR06870
C               SWR06880
C               SWR06890
C               SWR06900
C               SWR06910
C               SWR06920
C               SWR06930
C               SWR06940
C               SWR06950
C               SWR06960
C               SWR06970
C               SWR06980
C               SWR06990
C               SWR07000
C               SWR07010
C               SWR07020
C               SWR07030
C               SWR07040
C               SWR07050
C               SWR07060
C               SWR07070
C               SWR07080
C               SWR07090
C               SWR07100
C               SWR07110
C               SWR07120
C               SWR07130
C               SWR07140
C               SWR07150
C               SWR07160
C               SWR07170
C               SWR07180
C               SWR07190
C               SWR07200
C               SWR07210
C               SWR07220
C               SWR07230
C               SWR07240
C               SWR07250
C               SWR07260
C               SWR07270
C               SWR07280
C               SWR07290
C               SWR07300
C               SWR07310
C               SWR07320
C               SWR07330
C               SWR07340
C               SWR07350
C               SWR07360
C               SWR07370
C               SWR07380
C               SWR07390
C               SWR07400
C               SWR07410
C               SWR07420
C               SWR07430
C               SWR07440
C               SWR07450
C               SWR07460
C               SWR07470
C               SWR07480
C               SWR07490
C               SWR07500
C               SWR07510
C               SWR07520
C               SWR07530
C               SWR07540
C               SWR07550
C               SWR07560
C               SWR07570
C               SWR07580
C               SWR07590
C               SWR07600
C               SWR07610
C               SWR07620
C               SWR07630
C               SWR07640
C               SWR07650
C               SWR07660
C               SWR07670
C               SWR07680
C               SWR07690
C               SWR07700
C               SWR07710
C               SWR07720
C               SWR07730
C               SWR07740
C               SWR07750
C               SWR07760
C               SWR07770
C               SWR07780
C               SWR07790
C               SWR07800
C               SWR07810
C               SWR07820
C               SWR07830
C               SWR07840
C               SWR07850
C               SWR07860
C               SWR07870
C               SWR07880
C               SWR07890
C               SWR07900
C               SWR07910
C               SWR07920
C               SWR07930
C               SWR07940
C               SWR07950
C               SWR07960
C               SWR07970
C               SWR07980
C               SWR07990
C               SWR08000
C               SWR08010
C               SWR08020
C               SWR08030
C               SWR08040
C               SWR08050
C               SWR08060
C               SWR08070
C               SWR08080
C               SWR08090
C               SWR08100
C               SWR08110
C               SWR08120
C               SWR08130
C               SWR08140
C               SWR08150
C               SWR08160
C               SWR08170
C               SWR08180
C               SWR08190
C               SWR08200
C               SWR08210
C               SWR08220
C               SWR08230
C               SWR08240
C               SWR08250
C               SWR08260
C               SWR08270
C               SWR08280
C               SWR08290
C               SWR08300
C               SWR08310
C               SWR08320
C               SWR08330
C               SWR08340
C               SWR08350
C               SWR08360
C               SWR08370
C               SWR08380
C               SWR08390
C               SWR08400
C               SWR08410
C               SWR08420
C               SWR08430
C               SWR08440
C               SWR08450
C               SWR08460
C               SWR08470
C               SWR08480
C               SWR08490
C               SWR08500
C               SWR08510
C               SWR08520
C               SWR08530
C               SWR08540
C               SWR08550
C               SWR08560
C               SWR08570
C               SWR08580
C               SWR08590
C               SWR08600
C               SWR08610
C               SWR08620
C               SWR08630
C               SWR08640
C               SWR08650
C               SWR08660
C               SWR08670
C               SWR08680
C               SWR08690
C               SWR08700
C               SWR08710
C               SWR08720
C               SWR08730
C               SWR08740
C               SWR08750
C               SWR08760
C               SWR08770
C               SWR08780
C               SWR08790
C               SWR08800
C               SWR08810
C               SWR08820
C               SWR08830
C               SWR08840
C               SWR08850
C               SWR08860
C               SWR08870
C               SWR08880
C               SWR08890
C               SWR08900
C               SWR08910
C               SWR08920
C               SWR08930
C               SWR08940
C               SWR08950
C               SWR08960
C               SWR08970
C               SWR08980
C               SWR08990
C               SWR09000
C               SWR09010
C               SWR09020
C               SWR09030
C               SWR09040
C               SWR09050
C               SWR09060
C               SWR09070
C               SWR09080
C               SWR09090
C               SWR09100
C               SWR09110
C               SWR09120
C               SWR09130
C               SWR09140
C               SWR09150
C               SWR09160
C               SWR09170
C               SWR09180
C               SWR09190
C               SWR09200
C               SWR09210
C               SWR09220
C               SWR09230
C               SWR09240
C               SWR09250
C               SWR09260
C               SWR09270
C               SWR09280
C               SWR09290
C               SWR09300
C               SWR09310
C               SWR09320
C               SWR09330
C               SWR09340
C               SWR09350
C               SWR09360
C               SWR09370
C               SWR09380
C               SWR09390
C               SWR09400
C               SWR09410
C               SWR09420
C               SWR09430
C               SWR09440
C               SWR09450
C               SWR09460
C               SWR09470
C               SWR09480
C               SWR09490
C               SWR09500
C               SWR09510
C               SWR09520
C               SWR09530
C               SWR09540
C               SWR09550
C               SWR09560
C               SWR09570
C               SWR09580
C               SWR09590
C               SWR09600
C               SWR09610
C               SWR09620
C               SWR09630
C               SWR09640
C               SWR09650
C               SWR09660
C               SWR09670
C               SWR09680
C               SWR09690
C               SWR09700
C               SWR09710
C               SWR09720
C               SWR09730
C               SWR09740
C               SWR09750
C               SWR09760
C               SWR09770
C               SWR09780
C               SWR09790
C               SWR09800
C               SWR09810
C               SWR09820
C               SWR09830
C               SWR09840
C               SWR09850
C               SWR09860
C               SWR09870
C               SWR09880
C               SWR09890
C               SWR09900
C               SWR09910
C               SWR09920
C               SWR09930
C               SWR09940
C               SWR09950
C               SWR09960
C               SWR09970
C               SWR09980
C               SWR09990
C               SWR10000
C               SWR10010
C               SWR10020
C               SWR10030
C               SWR10040
C               SWR10050
C               SWR10060
C               SWR10070
C               SWR10080
C               SWR10090
C               SWR10100
C               SWR10110
C               SWR10120
C               SWR10130
C               SWR10140
C               SWR10150
C               SWR10160
C               SWR10170
C               SWR10180
C               SWR10190
C               SWR10200
C               SWR10210
C               SWR10220
C               SWR10230
C               SWR10240
C               SWR10250
C               SWR10260
C               SWR10270
C               SWR10280
C               SWR10290
C               SWR10300
C               SWR10310
C               SWR10320
C               SWR10330
C               SWR10340
C               SWR10350
C               SWR10360
C               SWR10370
C               SWR10380
C               SWR10390
C               SWR10400
C               SWR10410
C               SWR10420
C               SWR10430
C               SWR10440
C               SWR10450
C               SWR10460
C               SWR10470
C               SWR10480
C               SWR10490
C               SWR10500
C               SWR105
```

FILE: SWRITE FORTRAN 4 PURDUE / LARS 3031

```

DATA FLGT /*SIM */
EPS=1.E-5
C ***** LOAD TAPES AND READ PARAMETERS *****
C ***** WRITE(16,500)
500 FORMAT(//5X,'SPECIFY TAPE NUMBER ON WHICH RESULTS FILE IS LOCATED',/5X,'ITYPE EIGHT DIGIT TAPE NUMBER')
READ(16,505)INTAP
505 FORMAT(18)
WRITE(16,510,'SPECIFY FILE NUMBER AT WHICH RESULTS FILE IS LOCATED',/5X,'ITYPE THREE DIGIT FILE NUMBER')
READ(16,515)IFILE
515 FORMAT(13)
CALL MMTAPE(INTAP,IFILE,C)
WRITE(16,570,'SPECIFY THE TAPE NUMBER ONTO WHICH SIMULATED DATA IS TO BE WRITTEN',/5X,'(TYPE EIGHT DIGIT TAPE NUMBER)')
READ(16,575)TAPENO
575 FORMAT(18)
WRITE(16,580,'SPECIFY FILE NUMBER AT WHICH SIMULATED DATA IS TO BE WRITTEN',/5X,'(TYPE THREE DIGIT FILE NUMBER)')
READ(16,585)JFILE
585 FORMAT(13)
WRITE(16,590,'SPECIFY THE RUN NUMBER FOR THE SIMULATED DATA RUN',/5X,'(TYPE EIGHT DIGIT RUN NUMBER)')
1 READ(16,575)RUNNO
CALL MOUNT(TAPENO,12,'R')
MARG=JFILE-1
IF(MARG.LE.0) GO TO 3
DO 3 LIP=1,MARG
CALL TOPFF(12)
3 CONTINUE
5 READ(111)
IF(I.NE.1) GO TO 31C
READ(111)I,J,NOCLAS,NOCHAN,NOFLDS,NOPOCL,{FETVC3:IX1,IX=1,NOCHAN}
NOCH=(NOCHAN+1)/2*2
NOCOMP=NOCHAN*(NOCHAN+1)/2
ISTOP=NOCOMP*NOPPOOL
IEND=ISTOP+NOCHAN*NOPPOOL
15 READ(111,I,J,K)
IF(I.LT.3) GO TO 15
IF(K.NE.EDS) GO TO 15
READ(111)I,J,Z,IX1,IX=1,IEND
DO 17 IX=1,IEND
Z2:IX)=Z:IX1
17 CONTINUE
45 READ(111,AREANO,NOPNTS,NOLINE,INFO,IREC)
NOFET3=NOCHAN
IF(I.NE.5) GO TO 45
WRITE(6,520)
520 FORMAT(1H1///5X,'*****')
WRITE(6,525)
525 FORMAT(5X,'+DATA SIMULATION USING MCCABES EQUATION+')
WRITE(6,530)
530 FORMAT(5X,'++++++')
WRITE(6,535) RUNNO,IREC(3)
535 FORMAT(//5X,'SIMULATED DATA RUN IS ',I9,' FROM RUN ',I9)
WRITE(6,537) INFO(4),INFO(5),INFO(7),INFO(8)
537 FORMAT(5X,'LINE ',I5,' TO LINE ',I5,' AND COLUMN ',I5,' TO COLUMN ',I5)
$51
WRITE(6,540)INTAP,IFILE
540 FORMAT(//5X,'INPUT RESULTS FILE IS ON TAPE ',I9,' FILE ',I4)
WRITE(6,545)TAPENO,JFILE
545 FORMAT(//5X,'SIMULATED DATA IS ON TAPE ',I9,' FILE ',I4)
WRITE(6,550)
550 FORMAT(5X,'CHANNELS USED')
DO 560 IX=1,NOCHAN
WRITE(6,555){FETVC3:IX1,FQCAL1,IX1,FQCAL2,IX1}
555 FORMAT(5X,I2,2X,F5.2,'-',F5.2)
560 CONTINUE
CALL GTDATE:DATE
WRITE(6,565)DATE
565 FORMAT(//5X,'DATE OF SIMULATION IS ',3A4)

```

FILE: SWRITE FORTRAN A PURDUE / LARS 3C31

```

***** WRITTEN BY: BILL PFAFF ***** SWR0010
***** EDITED BY: MARWAN MUASHER JUNE 14, 1980 ***** SWR0020
***** THIS PROGRAM GENERATES SIMULATED DATA BASED ON A ***** SWR0030
***** CLASSIFICATION MAP OR A GROUND TRUTH MAP. EACH PIXEL ***** SWR0040
***** GENERATED THUS COMES FROM A KNOWN CLASS DISTRIBUTION. THE ***** SWR0050
***** METHOD USED IS AS FOLLOWS: ***** SWR0060
***** 1. A GOOD CLASSIFICATION IS CHOSEN AS A BASE FOR ***** SWR0070
***** SIMULATED DATA ***** SWR0080
***** 2. FROM THIS CLASSIFICATION WE KNOW THE NUMBER OF CLASSES, THE ***** SWR0090
***** CLASS STATISTICS, AND THE CLASS OF EACH PIXEL IN THE ***** SWR0100
***** AREA CLASSIFIED. ***** SWR0110
***** 3. A STREAM OF UNIFORM RANDOM NUMBERS IS GENERATED FOR ***** SWR0120
***** EACH CHANNEL. THEY ARE CHANGED TO NORMAL (0,1) DEVIATES. ***** SWR0130
***** 4. FOR EACH PIXEL, A RANDOM N-DIMENSIONAL VECTOR IS TRANSFORMED TO ***** SWR0140
***** BE DISTRIBUTED ACCORDING TO THE CLASS STATISTICS OF THAT ***** SWR0150
***** PIXEL. THIS IS THE SIMULATED DATA VECTOR. ***** SWR0160
***** 5. AS EACH LINE IS COMPLETED, IT IS WRITTEN TO AN OUTPUT TAPE. ***** SWR0170
***** TO RUN THE PROGRAM, YOU NEED TO HAVE THE FOLLOWING ***** SWR0180
***** EXEC FILE ON YOUR DISK: ***** SWR0190
***** GETDISK LARSYS ***** SWR0200
***** GETDISK DVSYS ***** SWR0210
***** GLOBAL TXTLIB CMSLIB FORTRAN SSP370 ***** SWR0220
***** FILEDEF 6 PRINTER ***** SWR0230
***** FILEDEF 16 TERMINAL ***** SWR0240
***** FILEDEF 12 TAP2 ***** SWR0250
***** FILEDEF 11 TAP1 :RECFM VS LRECL 1570 BLKSIZE 1570 ***** SWR0260
***** LOAD SWRITE GL0COM MMTAPE TAP0P 3CDVAL GTSERL GTDATE MFSD ***** SWR0270
***** RANDU WRTMTX ***** SWR0280
***** START SWRITE ***** SWR0290
***** THE PROGRAM WILL ASK FOR INFORMATION SUCH AS ***** SWR0300
***** TAPE NUMBERS, FILE NUMBERS, ..ETC. FROM HERE ON, IT ***** SWR0310
***** SHOULD BE EASY TO FOLLOW. ***** SWR0320
***** VARIABLES USED IN TPRINT ***** SWR0330
***** A = COVARIANCE STORAGE FOR FACTORING ***** SWR0340
***** AREANO= AREA NUMBER OF CLASSIFICATION ***** SWR0350
***** B = COVARIANCE STORAGE FOR MULTIPLICATION ***** SWR0360
***** DATA = DATA POINT STORAGE ***** SWR0370
***** DATVAL=LINE NUMBER AND ROLL PARAMETER ***** SWR0380
***** ICAL = CALIBRATION INFORMATION ***** SWR0390
***** IDREC = IDENTIFICATION RECORD STORAGE ***** SWR0400
***** ISTART=STARTING POINTS FOR GAUSS ***** SWR0410
***** LOGDAT=DATA POINTS IN LOGICAL FORMAT ***** SWR0420
***** NOCHAN=NUMBER OF CHANNELS IN CLASSIFICATION ***** SWR0430
***** NOCLAS=NUMBER OF CLASSES IN ORIGINAL STATISTICS ***** SWR0440
***** NOFLDS=NUMBER OF TEST FIELDS ***** SWR0450
***** NOPOLD=NUMBER OF POOLED CLASSES ***** SWR0460
***** PNTCLS=CLASSIFICATIONS ARRAY ***** SWR0470
***** Z = STATISTICS STORAGE ***** SWR0480
***** **** INITIALIZATION **** SWR0490
***** C
      INTEGER*2 I2, INTDAT, ICAL(3), ILIN(2), PNTCLS(1000), ISTAT(4),
      $          FETVC3:301
      $      LOGICAL*1 L1,21,LOGDAT(21),LCAL(6), DATCOLT:120001
      $      REAL*4 A(78),A2(12),Z(2700),B(12),I21,DATA(12),
      $          RMEAN(30,12),RVAR(30,12,12),Z2(2700),FKCAL(5,30)
      $      INTEGER*4 ISTART(12),EOS,INFO(17),AREANO, IDREC(200), TAPENO, THREE,
      $          CLAPNT(30), IMEAN(30,12),IVAR(30,12,12), YES,NO, DATE(3)
      $      INTEGER*4 RUNNC,FLG
      EQUIVALENCE (I2,L1),(INTDAT,LOGDAT),(ICAL,LCAL),(LNHRT,ILIN)
      EQUIVALENCE (FKCAL(1,1),IDREC(51))
      DATA EOS,S,AM /'EOS','1.C,0.D /
      DATA YES,NO,THREE /'YES ','NO ','3'/

```

APPENDIX III

```
*****
* Read data from LARS Results Tape
* and simulate data from it
* using the Box Muller relationship
*
*****  

* Swrite.c has been translated from
* the Lars Fortran Version of Swrite into
* the language 'c' for the Unix O.S.
* run on the DEC PDP-11/45
*
*****  

* Variables used in Swrite:  

*  

*   a      == Covariance storage for factoring
*   b      == Covariance storage for multiplication
*   data   == Data point storage
*   nochan == Number of channels in Classification
*   noclas == Number of classes in original statistics
*   nrool  == Number of rooled classes
*   rntcls == Classifications array
*   z      == Statistics storage
*
*****  

* compile with cc swrite.c -lrc -lfp /usr/lib/pdslib
*
*****  

* Initialize all variables used in swrite
* External variables are available to all functions
* within which they are declared
*/
main()
{
    extern int noclas, nochan, nrool, fdi, nropts, noline, renum;
    extern int ier, fd2;
    extern float ers, upper[5], lower[5];
    int j, k, fetwo3[5], debus, infot17, rntcls[1000], jj, ipol;
    int ii, idone, iy, ma, mb, mc;
    int noch, necomp, istor, iend, ix, ibes;
    int lnumrt, i2, icount, iz, ia, intdat, istat[6], ip, io, it, nine, no, in;
    int nosam, nos, pfd1;
    int fd3, fd4, err5;
    char buf[1500], *delim, *c1, *c2, *c3, datout[2100], obuf[2100];
    char rname[30], ttl[41];
    char mufile[30], cmfile[30], orname[30];
    float z[610], remnt, r;
    float b[6][6];
    double rmean[31][6];
    float data[13], z2[610];
    float clarntr[35];
    float mobuf[30];
    double ivar[31][6][6], revar, imean[31][6], tifloat, t2float, remean, semean;
    double temp1, temp2, temp0, a[20], a2[6];
    double sqrt(), log(), cos(), fmod(), s, t, f;
/* begin main program
* if debus is set to minus one the following is
```

```
* Printed out for verification on user terminal:  
*      nochan, nocomp, fetvc3, info, nntcls, noprnts, noline, z, upper, lower;  
*/  
debus = 1;  
eps = .00001;  
/* read records 2,4,5,&6 off results tape*/  
/* skip records 1,3 */  
readito5(buf, fetvc3, z, info);  
noch = (((nochan+1)/2)*2);  
nocomp = (nochan*(nochan+1)/2);  
istop = nocomp * nocomp;  
iend = nochan * nocomp + istop;  
nosam = 4 * ((noprnts + 9)/4);  
if(nochan>5) {  
    printf("Number of channels is %d but internal", nochan);  
    printf(" storage only allows 5\n");  
    printf("Execution terminated abnormally \n");  
    exit();  
}  
if(nocomp>29) {  
    printf("Number of pooled classes is %d but internal", nocomp);  
    printf(" storage only allows 29\n");  
    printf("Execution terminated abnormally \n");  
    exit();  
}  
if(noprnts>=1000) {  
    printf("Number of points per line is %d but internal", noprnts);  
    printf(" storage only allows 1000\n");  
    printf("Execution terminated abnormally \n");  
    exit();  
}  
printf("nochan=%d, nocomp=%d\n", nochan, nocomp);  
if(debus == -1) {  
    for(k=0; k<nochan; ++k)  
        printf("fetvc3[%d] = %d \n", k, fetvc3[k]);  
    for(k=0; k<iend; k = k+10) {  
        for(j=k; j<=k+9; ++j)  
            printf(" %f", z[j]);  
        printf("\n");  
    }  
    printf("\nField size: ");  
    printf("\n      line %d to %d with interval %d",
           info[4], info[5], info[6]);  
    printf("\n      cols %d to %d with interval %d\n",
           info[7], info[8], info[9]);  
    printf("Number of lines classified is %d\n", noline);  
    printf("Number of points classified per line is %d\n", noprnts);  
    for(j=0; j<nochan; ++j) {  
        printf("upper[%d] = %f ", j, upper[j]);  
        printf("lower[%d] = %f \n", j, lower[j]);  
    }  
}  
read6rec(buf, nntcls);  
if(debus == -1) {  
    printf("\nFirst line of record %d follows\n", recnum);  
    for(j=0; j<noprnts; j=j+10) {  
        for(k=j; k<=j+9; ++k)  
            printf(" %d", nntcls[k]);  
        printf("\n");  
    }  
}
```

```
>
/* make a copy of statistics array z */
for(ix=1; ix<=iend; ++ix)
    z2[ix] = z[ix-1];
/* Create output file */
sets("\nSpecify linewriter output file pathname: ", orname);
fd2 = creat(ormame, 0777);
if(fd2<0)
    printf("Cannot creat lr output file %s\n", orname);
rfd1 = 1;
sets("Specify simulated data PDS file pathname: ", rname);
sets("enter a 40 char title:\n", ttl);
sets("\nSpecify mean vector file pathname: ", mvfile);
sets("\nSpecify covariance matrix file pathname: ", cmfile);
rdsoren(rfd1, rname);
putfmt(rfd1, 1, nosam, noline, 8, nochan);
putttl(rfd1, ttl);
fd3 = creat(mvfile, 0777);
if(fd3<0)
    printf("Cannot creat mvfile %s (MAIN)\n", mvfile);
fd4 = creat(cmfile, 0777);
if(fd4<0)
    printf("Cannot creat cmfile %s (MAIN)\n", cmfile);
/* write output page one */
delim = " ++++++++\n";
cl = " +Data Simulation Using Box-Muller Relationship+";
printf(fd2, "\n\n\n\n%e\n%e\n%e\n", delim, cl, delim);
printf(fd2, "\n Line %d to line %d with interval %d \n",
       info[4], info[5], info[6]);
printf(fd2, "\n Column %d to column %d with interval %d \n",
       info[7], info[8], info[9]);
printf(fd2, "\n channels used \n");
for(ix=1; ix<=nochan; ++ix)
    printf(fd2, "%d %f - %f\n", fetwo3[ix-1], lower[ix-1],
           upper[ix-1]);
/* output new page character */
printf(fd2, "\014");
/* Factor covariance matrices */
ibes = 1;
for(ix=1; ix<=nocoal; ++ix) {
    idone = ibes + nocomp - 1;
    k = 0;
    for(iy=ibes; iy<=idone; ++iy) {
        k = k + 1;
        a[k] = z[iy-1];
    }
}
/* call function to perform factorine */
mfsd(a);
if(ier == -1)
    printf(fd2, "Error -1\n");
if(ier>=1)
    printf(fd2, "Error st %d\n");
if((ier == -1) || (ier >= 1))
    exit(1);
k = 0;
for(iy=ibes; iy<=idone; ++iy) {
    k = k + 1;
    z[iy-1] = a[k];
}
ibes = ibes + nocomp;
```

```
    }
/* initialize random number generator */
srand(1);
/* initialize arrays */
for(ma=1; ma<=noclas; ++ma) {
    clarnl[ma] = 0.0;
    for(mb=1; mb<=nochan; ++mb) {
        imean[ma][mb] = 0.0;
        rmean[ma][mb] = 0.0;
        for(mc=1; mc<=nocomp; ++mc) {
            ivar[ma][mb][mc] = 0.0;
        }
    }
}
t = 25.0;
lnwrt = 0;
/* this while loop is repeated for each line in the classification */
while(reclnum == 6) {
    lnwrt = lnwrt + 1;
    s = lnwrt;
    f = fmod(s, t);
    if(f == 0.0)
        printf("%d Lines out of %d are completed\n",
               lnwrt, noline);
    i2 = 0;
    icoount = 4;
    for(ix=1; ix<=nortns; ++ix) {
        icoount = icoount + 1;
        i2 = nrtcls[ix-1];
        ipol = (i2-1) * nochan;
        ibes = (i2-1) * nocomp;
        k = ibes;
        for(iy=1; iy<=nochan; ++iy) {
            for(iz=1; iz<=iy; ++iz) {
                k = k + 1;
                b[iy][iz] = z[k-1];
                if(iy != iz)
                    b[iz][iy] = 0.0;
            }
        }
        for(iy=1; iy<=nochan; ++iy) {
            a2[iy] = rand();
            a[iy] = rand();
            a2[iy] = a2[iy] / 32767.0;
            a[iy] = a[iy] / 32767.0;
            a[iy] = sqrt(-2.0 * log(a2[iy])) * cos(6.28318 * a[iy]);
        }
        clarnl[i2] = clarnl[i2] + 1.0;
        for(iy=1; iy<=nochan; ++iy) {
            data[iy] = 0.0;
            ia = nrool * nocomp + ipol + iy;
            for(iz=1; iz<=nochan; ++iz)
                data[iy] = data[iy] + b[iy][iz] * a[iz];
            data[iy] = data[iy] + z[ia-1];
            intdat = data[iy] + .5;
            if(intdat<0) intdat = 0;
            if(intdat>255) intdat = 255;
            istat[iy] = intdat;
            pos = (iy-1)*nosam + icoount - 5;
            if(pos>2100) printf("datout internal buffer full\n");
        }
    }
}
```

```
datout[Pos1] = intdat & 0377;
for(iz=1; iz<=6; ++iz)
    datout[Pos+iz] = 0;
}
for(ii=1; ii<=nochan; ++ii) {
    imean[i2][ii] = imean[i2][ii] + istat[ii];
    for(jj=ii; jj<=nochan; ++jj) {
        temp0 = istat[ii];
        temp1 = istat[jj];
        ivar[i2][ii][jj] = ivar[i2][ii][jj] + temp0 * temp1;
    }
}
i = 0;
for(iy=0; iy<nosam; ++iy) {
    for(iz=0; iz<nochan; ++iz)
        obuf[li++ = datout[nosam*iz + iy];
}
pds1Pos(pfd1, lnrwt-1);
if(putline(pfd1, lnrwt-1, obuf, nosam*nochan) != 0)
    printf("Putline to PDS output failed\n");
read6rec(buf, pntcls);
}
/* end of while loop */
/*
for(i2=1; i2<=18; ++i2) {
    printf(fd2, "\n %d \n", i2);
    for(ii=1; ii<=nochan; ++ii) {
        for(jj=ii; jj<=nochan; ++jj) {
            printf(fd2, " %f ", ivar[i2][ii][jj]);
        }
    }
    printf(fd2, "\n");
}
for(i2=1; i2<=18; ++i2) {
    printf(fd2, "\n");
    for(ii=1; ii<=nochan; ++ii) {
        printf(fd2, " %f \n", imean[i2][ii]);
    }
}
*/
for(ip=1; ip<=noclas; ++ip) {
    for(io=1; io<=nochan; ++io) {
        if(clapnt[ip]>0.0) {
            t1float = imean[ip][io];
            t2float = clapnt[ip];
            rmean[ip][io] = t1float/t2float;
        }
        for(it=io; it<=nochan; ++it) {
            if(clapnt[ip]>1.0) {
                rept = clapnt[ip];
                revar = ivar[ip][io][it];
                remean = imean[ip][io];
                semean = imean[ip][it];
                temp0 = revar/(rept-1.0);
                temp1 = remean/rept;
                temp2 = semean/(rept-1.0);
                ivar[ip][io][it] = temp0 - (temp1*temp2);
                ivar[ip][it][io] = ivar[ip][io][it];
            }
        }
    }
}
```

```
>
>
/* output results */
for(ip=1;ip<=noclas;++ip) {
    printf(fd2,"      Class number %d      %f      points\n\n\n",
           ip,clapnt[ip]);
    printf(fd2,"                                     Actual      Simulated\n");
    printf(fd2,"                                     Mean      mean\n");
    for(ix=1;ix<=nochan;++ix) {
        nine = nocomp * noclas + (ip - 1) * nochan;
        printf(fd2,"      Channel %d (%.3f - %.3f)      %.3f      %.3f\n",
               ix,fetvc3[ix-1],lower[ix-1],upper[ix-1],z2[nine+ix],
               rmean[ip][ix]);
        mobuf[ix-1] = rmean[ip][ix];
    }
    err5 = write(fd3,mobuf,4*nochan);
    if(err5<0)
        printf("Error occurred writing Mean Vector file (MAIN) \n");
    printf(fd2,"\\n\\n\\n\\n\\n      Actual Covariance Matrix\\n");
    for(no=1;no<=nocomp;++no) {
        nine = (ip-1) * nocomp;
        a[no] = z2[nine+no];
    }
    wrmtx(a,fetvc3);
    printf(fd2,"\\n\\n\\n\\n      Simulated Covariance Matrix\\n");
    no = 0;
    for(io=1;io<=nochan;++io) {
        for(in=1;in<=io;++in) {
            no = no + 1;
            a[no] = ivar[ip][io][in];
            mobuf[no-1] = ivar[ip][io][in];
        }
    }
    wrmtx(a,fetvc3);
    err5 = write(fd4,mobuf,4*nocomp);
    if(err5<0)
        printf("Error occurred writing cm file (MAIN) \n");
    printf(fd2,"\\014");
}
/* cexit terminates activities on open files and flushes
 * the output buffer.  cexit is part of the c library */
printf("swrite.c is finished\n");
printf("\\n The simulated data (in PDS format) is at %s \\n",pname);
printf("The linewriter output file is at %s \\n",opname);
printf("The Mean vector file is at %s \\n",mvfile);
printf("The Covariance Matrix file is at %s \\n",cmfile);
pdsclose(pfd1);
cexit();
}
/* end of main program */
*****  
/* function to read records 1 thru 5 follows */
readto5(buf,fetvc3,word,info)
int *fetvc3,*info;
char *buf;
float *word;
{
    extern int noclas,nochan,nocomp,fd1,noents,noline;
    extern float upper[5],lower[5];
```

```
int err1, j, k, reenum, statsize;
char tbuf[4];
float ibmdec();
fd1 = open("/dev/rmt0", O);
if(fd1<0)
    printf("Cannot open 9 trk tape file (READ1T05) \n");
/* skip record 1 */
err1 = read(fd1, buf, 1500);
if(err1 == -1)
    printf("Error occurred in reading record 1 (READ1T05) \n");
/* read record 2 */
err1 = read(fd1, buf, 1500);
if(err1 == -1)
    printf("Error occurred reading record 2 (READ1T05) \n");
noclas = buf[19];
nochan = buf[23];
nopool = buf[31];
for(j=0, k=33; j<nochan; ++j, k=k+4)
    fetwo3[j] = buf[k];
for(k=32+4*nochan, j=0; j<2*nochan; ++j, k=k+4) {
    tbuf[0] = buf[k];
    tbuf[1] = buf[k+1];
    tbuf[2] = buf[k+2];
    tbuf[3] = buf[k+3];
    if(j<nochan)
        lower[j] = ibmdec(tbuf);
    else upper[j-nochan] = ibmdec(tbuf);
}
/* skip record 3 */
err1 = read(fd1, buf, 1500);
if(err1 == -1)
    printf("Error occurred reading record 3 (READ1T05) \n");
reenum = buf[11];
while(reenum == 3) {
    err1 = read(fd1, buf, 1500);
    if(err1 == -1)
        printf("Error occurred reading record three (READ1T05)\n");
    reenum = buf[11];
}
/* Handle record 4 */
statsize = nochan*nopool + (nochan*(nochan+1)/2)*nopool;
for(k=0, j=16; k<statsize; ++k, j=j+4) {
    tbuf[0] = buf[j];
    tbuf[1] = buf[j+1];
    tbuf[2] = buf[j+2];
    tbuf[3] = buf[j+3];
    wordd[k] = ibmdec(tbuf);
}
err1 = read(fd1, buf, 1500);
if(err1 == -1)
    printf("Error occurred reading record 5 (READ1T05) \n");
nopnts = (buf[18]<<8 | (buf[19] & 0377));
noline = (buf[22]<<8 | (buf[23] & 0377));
info[4] = (buf[38]<<8 | (buf[39] & 0377));
info[5] = (buf[42]<<8 | (buf[43] & 0377));
info[6] = (buf[46]<<8 | (buf[47] & 0377));
info[7] = (buf[50]<<8 | (buf[51] & 0377));
info[8] = (buf[54]<<8 | (buf[55] & 0377));
info[9] = (buf[58]<<8 | (buf[59] & 0377));
}
```

```
int noelas, nopool, nochan, fd1, noverts, noline;
float upper[5], lower[5];
float ibmdec(buf)
    char *buf;
{
    int k, l, m, tint;
    char nbuf[2], tchar, temp3, byte[4], sian;
    float word;
    m = 0;
    sian = 0000;
    if(buf[0] < 0)
        sian = 0200;
    tint=((buf[0] & 0177) - 64) * 4;
    while(buf[1] > 0) {
        buf[1]=((buf[1]<<1) & 0376);
        ++m;
    }
    k = 8-m;
    nbuf[0]=buf[2];
    nbuf[1]=buf[3];
    for(l=1; l<=k; ++l) {
        nbuf[0]=((nbuf[0]>>1) & 0177);
        nbuf[1]=((nbuf[1]>>1) & 0177);
    }
    for(l=1; l<=m; ++l) {
        buf[2]=((buf[2]<<1) & 0376);
        buf[3]=((buf[3]<<1) & 0376);
    }
    buf[1]=(buf[1] + nbuf[0]);
    buf[2]=(buf[2] + nbuf[1]);
    tint = tint-m+128;
    if(tint < 0) tchar = 0000;
    else if(tint > 255) tchar = 0377;
    else tchar = tint;
    buf[1]=(buf[1] & 0177);
    temp3 = tchar<<7;
    temp3 = temp3 & 0200;
    buf[1] = temp3 | buf[1];
    buf[0] = tchar>>1;
    buf[0] = buf[0] & 0177;
    buf[0] = buf[0] | sian;
    byte[0] = buf[1] & 0377;
    byte[1] = buf[0] & 0377;
    byte[2] = buf[3] & 0377;
    byte[3] = buf[2] & 0377;
    pack(byte, &word);
    return(word);
}
/* the function pack takes the 4 8-bit character bytes
 * that have been rearranged by ibmdec and packs
 * them into a floating point word */
pack(byte, cword)
    char *byte, *cword;
{
    int j;
    for(j=0; j<4; ++j)
        cword[j] = byte[j];
}
*****read6rec(buf, nntcls)
/* function to read record 6 follows */
read6rec(buf, nntcls)
```

```
int *pntols;
char *buf; 
extern int nopts, reenum, fd1;
int j, n, err1, k;
n = 0;
err1=read(fd1, buf, 1500);
if(err1 == -1)
    printf("Error occurred reading record 6\n");
reenum=buf[11];
for(j=20, k=0; k<norts; k++, j=j+2) {
    pntols[n]=(buf[j+1] & 0377);
    ++n;
}
}

int reenum;
/***********************/
/* Mfsd factors the matrices passed by the main program.
 * Mfsd is a direct translation from the Lars
 * fortran version of the same name */
mfsd(a)
double *a; 
extern int nochan, ier;
extern float eps;
int kpiv, k, ind, lend, i, lanf, l, lind;
double tol, dsum, dpiw, r, fabs();
double done, sqrt(), x;
done = 1.0;
ier = 0;
kpiv = 0;
for(k=1; k<=nochan; ++k) {
    kpiv = kpiv+k;
    ind = kpiv;
    lend = k-1;
    r = eps*a[kpiv];
    tol = fabs(r);
    for(i=k; i<=nochan; ++i){
        dsum = 0.0;
        if(lend != 0) {
            for(l=1; l<=lend; ++l) {
                lanf = kpiv-l;
                lind = ind-l;
                dsum = (dsum+(a[lanf]*a[lind]));
            }
        }
        dsum = a[ind]-dsum;
        if((i-k) == 0) {
            if(((dsum-tol) <= 0.0) && (dsum > 0.0) && (ier<=0))
                ier = k-1;
            if(((dsum-tol) <= 0.0) && (dsum <= 0.0)) {
                ier = -1;
                printf("Error -1\n");
                exit(1);
            }
            x = dsum;
            dpiw = sqrt(x);
            a[kpiv] = dpiw;
            dpiw = done/dpiw;
        }
        if((i-k) != 0)
            a[ind] = dsum*dpiw;
    }
}
```

```
    ind = ind+i;
}
}

int ier;
float eps;
/************* */
/* the write matrix function writes to the output
 * file the lower half of the nochan*nochan covariance matrix
 * which is passed through the parameter a. */
wrmtx(a, fd2, eps);
    int *fetw3;
    double *a;
extern int fd2, nochan;
extern float upper[5], lower[5];
int j, k, m;
m = 0;
printf(fd2, "\n      Spectral      ");
for(j=0; j<nochan; ++j)
    printf(fd2, "%7.5f -      ", lower[j]);
printf(fd2, "\n      Band      ");
for(j=0; j<nochan; ++j)
    printf(fd2, "%7.5f      ", upper[j]);
for(j=0; j<nochan; ++j) {
    printf(fd2, "\n\n      %7.5f -\n", lower[j]);
    printf(fd2, "      %7.5f      ", upper[j]);
    for(k=0; k<j; ++k) {
        printf(fd2, "%7.3f      ", a[m]);
    }
}
}
int fd2;
/************* */
sets(p, s)
char *p, *s;
{
    char c, *p1;
    printf(p);
    p1 = s;
    while((c = setchar()) != '\n')
        *p1++ = c;
    *p1 = 0;
}
```