Environment and Planning A, 1980, volume 12, pages 1073-1081

# Algorithm 9: Simulation of autocorrelation for aggregate data

M F Goodchild
Department of Geography, The University of Western Ontario, London, Canada
Referee S Openshaw
Department of Town and Country Planning, University of Newcastle, Newcastle upon Tyne, England
Received 24 July 1979, in revised form 5 January 1980

*Language.* Subroutine PRESC is written in ANSI Fortran.

**Description and purpose**

In the last few years a considerable volume of literature has accumulated on the zoning or aggregation problem. Its effects have been analyzed in the context of spatial interaction (Masser and Brown, 1978), location allocation (Goodchild, 1979), and ecological correlation (Openshaw, 1978). Each field, and particularly the last, recognizes the importance of the spatial autocorrelation of parameters between zones in controlling the severity of aggregation effects. In general, if neighbouring zones have similar characteristics then aggregation effects will be relatively weak.

It is assumed that autocorrelation will be expressed by some form of the Cliff and Ord (1973) generalization of the Moran statistic

$$I' = \frac{n \sum_{i,j} (x_i - \bar{x})(x_j - \bar{x}) w_{ij}}{\sum_{i,j} w_{ij} \sum_{i} (x_i - \bar{x})^2} ,$$

where

$n$ is the number of zones,
$x_i$ is the value of variable $x$ in zone $i$, and
$w_{ij}$ is a weight.

The weights may take the form of a binary matrix, that is, the $i$th, $j$th element takes the value 1 if $i$ and $j$ share a common boundary and 0 otherwise, or some decreasing function of the distance between $i$ and $j$, or an increasing function of the length of common boundary. Whereas the first may be simplistic, the advantages of the latter two have to be weighed against the arbitrariness of any specific choice of functions. The weights may also be standardized such that $\sum_j w_{ij} = 1$.

Although it is easy to measure the spatial autocorrelation of an observed set of data, the relationships between the index and various stochastic processes are usually complex. Cliff and Ord analyzed two such processes in developing null hypotheses for $I'$; in the N process zone values are sampled randomly from a Gaussian distribution estimated from the observed data, whereas in the R process the observed zone values are randomly permuted.

Cliff and Ord (1973, Appendix 1) were able to simulate positive autocorrelation by the following autoregressive process. Let the value in zone $i$ be influenced by the values in other zones in a linear manner: viz

$$x_i = \rho \sum_j v_{ij} x_j + u_i ,$$

where
$v_{ij}$ is a weight,
$u_i$ is an error term, having a Gaussian distribution with zero mean, and
$\rho$ is a constant parameter indirectly determining the strength of autocorrelation.
In matrix notation

$$\boldsymbol{x} = (\mathbf{I} - \rho\mathbf{v})^{-1}\boldsymbol{u} \ .$$

It is clear that a realization of the process can be generated by obtaining a vector $\boldsymbol{u}$ and solving for $\boldsymbol{x}$; however, $I'$ could not be controlled directly since its dependence on $\rho$, and also on $\mathbf{w}$ and $\mathbf{v}$, was not understood.

Openshaw (1978) was concerned with the effects of spatial autocorrelation on bivariate ecological models of the form

$$y_i = a + bx_i + e_i \ ,$$

where $a$, $b$ are model parameters, and $e_i$ is an error term.
To analyze or simulate this problem would require a stochastic process capable of producing two variables with prescribed autocorrelations, $I'_x$ and $I'_y$, and a prescribed correlation $R_{xy}$. In the apparent absence of a suitable model Openshaw treated the problem as the minimization of a function of $2n$ variables; find $x$ and $y$ to minimize

$$F(x,y) = (I'_x - I^*_x)^2 + (I'_y - I^*_y)^2 + (S^*_x)^2 + (S^*_y)^2 + (K^*_x)^2 + (K^*_y)^2 + (R_{xy} - R^*_{xy})^2 + (a - a^*)^2 + (b - b^*)^2 \ ,$$

where
$^*$ indicates an observed value of a parameter,
$x, y$ are positive,
$S$ and $K$ denote skewness and kurtosis, respectively, and
$I'_x$, $I'_y$, $R_{xy}$, $a$, and $b$ are prescribed.
The skewness and kurtosis are introduced to give the simulated values a near Gaussian distribution. Openshaw and Taylor (1978) reported that with a quasi-Newton procedure the minimization for 99 zones required 920 seconds of IBM 370/168 time.

A recent paper by Haining (1978) explored the equivalences between autoregressive and moving average models. In the latter case a vector $\boldsymbol{u}$ is generated, usually by sampling a Gaussian distribution, and some sort of spatial averaging, equivalent to a filter, is applied to obtain $\boldsymbol{x}$. The characteristics of the filter determine the autocorrelation structure of the data. However, Haining discussed processes on a lattice, and it is not clear how they can be adapted to give useful results for irregular areas, particularly when $x_i$ is an average or summation over a finite area, rather than a point sample.

It is clear that although this is an important and exciting area the literature on spatial processes has not advanced to the point where interzonal autocorrelation can be readily simulated by realizing a stochastic process, particularly in the case of Openshaw's bivariate problem. The purpose of the present paper is to describe a method of simulation which was developed for the same purpose as Openshaw's, to study the effects of autocorrelation on spatial models, and which seems to possess a number of important advantages: it is much faster, and treats many of the required parameters as constraints rather than as parts of the objective function. It differs in that the specified parameters $a$, $b$, and $R_{xy}$ define a population from which actual values are sampled, so that for small samples the values obtained may differ substantially from those specified. It may be necessary to run the procedure a number of times, therefore, and select the data with the best results.

## Examples

*Univariate case* Suppose we require a vector $x$ sampled from a Gaussian distribution, mean $\mu_x$ and standard deviation $\sigma_x$, and with autocorrelation $I'_x$. All of the requirements except the last can be met by one of a number of standard methods for sampling the Gaussian distribution. We may therefore concentrate on how these values should be arranged among the zones so as to yield the required $I'_x$. Clearly one possibility is to examine all $n!$ arrangements. Any systematic search of the combinatorial problem is undesirable since it is likely to create systematic biasses in the eventual solution. The proposed method is therefore in a sequence of steps as follows:

*Step 1* Assign values to zones and compute $I'_x$.
*Step 2* Select a random pair of zones.
*Step 3* Compute the change in $I'_x$ if the values in the zones are swapped.
*Step 4* If the new $I'_x$ is closer to the target, make the swap. If not, go to *step 2*.
*Step 5* If the new $I'_x$ is within a tolerance of the target, or if too many unsuccessful tries have been made, stop. If not, go to *step 2*.

The program has three modes of operation in the univariate case. In mode 1, values are sampled from a Gaussian distribution. In mode 2 the user supplies a vector of values, which are then randomly permuted among the zones before the swap process begins. Mode 3 is as mode 2 except that the permutation step is omitted.

*Bivariate case* Because the correlation between the two variables, and also the $a$ and $b$ coefficients, are all invariant under the spatial rearrangement of pairs of values, an approach similar to that for the univariate case can be used for the bivariate case. Values of $x$ are input or generated in one of three ways, corresponding to modes 1, 2, and 3 above, but denoted in the bivariate case by modes 4, 5, and 6, respectively. A second variable $y$ is then generated from the values of $a$ and $b$ supplied by the user, and by sampling an error term from a Gaussian distribution of zero mean and standard deviation calculated from the $R_{xy}$ specified. The criterion in *step 4* for the bivariate case is that the greater of the differences between $I'_x$ and $I'_y$ and their respective targets be less than before.

*General case* Let $z_i$ denote the value of $x_i$ which has been adjusted to zero mean, $z_i = x_i - \bar{x}$. Rearrangement of values can only change the term $\sum_{i,j} z_i z_j w_{ij}$ in the expression for $I'_x$. When zones $m$ and $n$ are swapped, the change in this term is given by

$$\Delta_{mn} = 2(x_n - x_m)(S_m - S_n - x_n w_{nm} + x_m w_{mn}) ,$$

where $S_m = \sum_k x_k w_{mk}$.

Thus *step 4* can be reduced to a simple test of $\Delta_{mn}$. In *step 5* the values of the vector $S$ must be updated, as well as $x_m$ and $x_n$.

The subroutine given below provides storage for the vectors $x$ and $y$, and also $S$, and for the matrix of weights **w**. In cases where the weights matrix is sparse and core storage is at a premium, the program might be rewritten to store only the nonzero weights, together with a system of pointers.

## Structure

*Subroutine* PRESC(N, NMAX, W, MODE, XTARG, NTRYM, DSEED, XAVE, XSD, XIN, YTARG, A, B, RXY, IWRITE, XAUTO, YAUTO, NSW, X, Y, IFAULT, SX, SY, SRT)

*Arguments*

All arguments except DSEED are single precision with type in accordance with the usual Fortran naming convention. DSEED is double precision.

*Input*

| | |
|---|---|
| N | The number of zones. |
| NMAX | The dimensions of W in the calling program. |
| W | A matrix of weights, with dimensions N by N. W(I, J) is the influence of $j$ at $i$. The diagonal terms should be zero. |
| MODE | The mode of operation, with range 1 to 6 (see **Examples**). |
| XTARG | The target for $x$ autocorrelation. |
| NTRYM | The maximum number of unsuccessful tries at a swap. |
| DSEED | A double precision seed for the random number generators (see *Auxiliary algorithms* below). |
| XAVE, XSD | The mean and standard deviation of Gaussian distribution sampled for $x$ (modes 1 and 4 only, dummy otherwise. See *Output*). |
| XIN | A vector of length N containing $x$ values (modes 2, 3, 5, and 6 only, dummy otherwise). |
| YTARG | The target for $y$ autocorrelation (modes 4 to 6 only, dummy otherwise). |
| A, B, RXY | The parameters for the regression model (modes 4 to 6 only, dummy otherwise). |
| IWRITE | If greater than 0, this logical unit is used to print useful statistics. |

*Output*

| | |
|---|---|
| XAVE, XSD | The sample mean and standard deviation of $x$. |
| XAUTO | The final autocorrelation for $x$. |
| YAUTO | The final autocorrelation for $y$ (modes 4 to 6 only, dummy otherwise). |
| NSW | The number of successful swaps made. |
| X | The final values. |
| Y | The final values (modes 4 to 6 only, dummy otherwise). |
| IFAULT = 1 | If execution terminated because NTRYM has been exceeded. |
| = 0 | For normal termination. |

*Working space*

| | |
|---|---|
| SX, SY, SRT | Vectors of length N. |

*Auxiliary algorithms*

The subroutine calls two external routines, GGUBS and GGNML (IMSL, 1979), which generate vectors of independent random numbers with uniform (random numbers in the range 0 to 1·0) and Gaussian (random normally distributed numbers with mean 0·0 and standard deviation 1·0) distributions respectively. Users can substitute their own subroutines here, and use whatever random number procedures they have available.

**Restrictions**

The positive and negative limits of $I'_x$ and $I'_y$ vary widely, and this depends on the pattern of weights. The expression for $I'_x$ can be manipulated to give

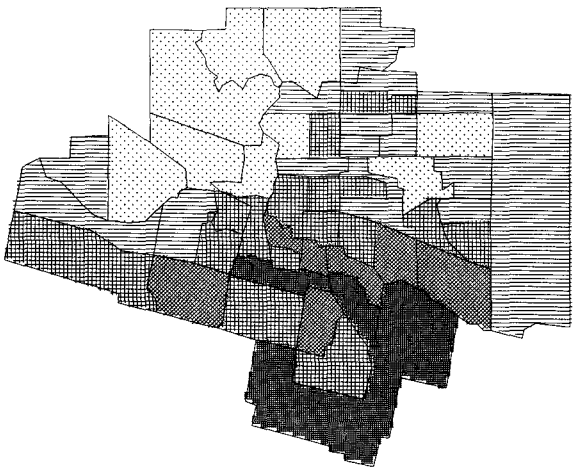
$$I'_x = \frac{n\sum_i (z_i^2 \sum_j w_{ij})}{\sum_{i,j} w_{ij} \sum_i z_i^2} - \frac{n\sum_{i,j}(x_i - x_j)^2 w_{ij}}{2\sum_{i,j} w_{ij} \sum_i z_i^2}.$$

The second term is similar to the Geary coefficient, and becomes very small when $I'_x$ approaches its upper limit. The first term is a weighted average which tends to 1 if $\sum_j w_{ij}$ is constant, that is if each zone has the same total weight or if the weights are standardized. The upper limits for the data sets in table 1 varied from +0·9 to +1·4.

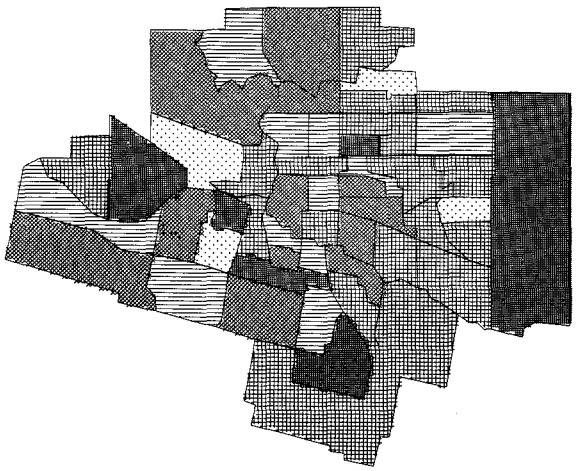
If the user tries to force $I'_x$ beyond these limits the program will terminate because of the maximum set for the number of tries. In the bivariate case the limits for $I'_x$, $I'_y$, and $R_{xy}$ are interdependent, and again depend on the distribution of $\sum_j w_{ij}$.

**Time**

Two examples of simulations are shown in figure 1. Both were generated by means of mode 1, which gives a Gaussian distribution, and with prescribed autocorrelations of 0·75 and −0·25. Both targets were reached to within three decimal places. Contiguity weights were used. In the positive case the process gives rise to a prominent peak, because similar values tend to be located close to each other. The location of the peak on the southern edge of the city is a matter of chance—in reality, of course, many parameters are most intense in the central business district. The negative case shows a checkerboard pattern of low and high values.



(a)



(b)

**Figure 1.** Simulated spatial autocorrelations for London, Ontario Census Tracts ($n = 51$) for autocorrelations of (a) +0·75 and (b) −0·25.

Table 1 shows a sample of performances for various sizes of problems of the univariate case, where contiguity weights and a Gaussian distribution were used. The $n = 22$ example consisted of the Planning Districts of London, Ontario; the others are Census Tract maps of London, Edmonton, and the west part of Montreal respectively. Times were approximately 0·25 milliseconds per try on a CDC Cyber 73/14, independently of $n$, which allows all these problems to be solved in much less than a second.

**Table 1.** Tries for various target autocorrelations, univariate case, and contiguity weights.

| $n$ | $I'_x = +0{\cdot}2$ | $I'_x = +0{\cdot}5$ | $I'_x = -0{\cdot}2$ | $I'_x = -0{\cdot}5$ |
|---|---|---|---|---|
| 22 | 20 | 35 | 50 | 350 |
| 51 | 10 | 70 | 60 | 450 |
| 79 | 50 | 300 | 20 | 230 |
| 165 | 50 | 210 | 60 | 750 |

**References**

Cliff A D, Ord J K, 1973 *Spatial Autocorrelation* (Pion, London)

Goodchild M F, 1979 "The aggregation problem in location allocation" *Geographical Analysis* **11** 240-255

Haining R P, 1978 "The moving average model for spatial interaction" *Transactions of the Institute of British Geographers, New Series* **3** 202-225

IMSL, 1979 *IMSL Library Reference Manual* seventh edition (International Mathematical and Statistical Libraries, Inc., Houston, Texas)

Masser I, Brown P J B (Eds), 1978 *Spatial Representation and Spatial Interaction* (Martinus Nijhoff, Leiden, Holland)

Openshaw S, 1978 "An empirical study of some zone-design criteria" *Environment and Planning A* **10** 781-794

Openshaw S, Taylor P J, 1979 "A million or so correlation coefficients: three experiments on the modifiable areal unit problem" in *Statistical Applications in the Spatial Sciences* Ed. N Wrigley (Pion, London) pp 127-144

## Subroutine PRESC

```
      SUBROUTINE PRESC(N,NMAX,W,MODE,XTARG,NTRYM,DSEED,XAVE,XSD,XIN,
     1YTARG,A,B,RXY,IWRITE,XAUTO,YAUTO,NSW,X,Y,IFAULT,SX,SY,SRT)
C
C         THIS ROUTINE SIMULATES AUTOCORRELATION FOR AGGREGATE DATA
C
C         ARGUMENTS
C         ON INPUT
C         N NUMBER OF ZONES
C         NMAX DIMENSIONS OF W ARRAY IN CALLING ROUTINE
C         W MATRIX OF WEIGHTS, DIMENSIONS N BY N. W(I,J) IS
C           THE INFLUENCE OF J AT I
C         MODE MODE OF OPERATION, RANGE 1 TO 6
C         XTARG TARGET FOR X AUTOCORRELATION
C         NTRYM MAXIMUM NUMBER OF UNSUCCESSFUL TRIES AT A SWAP
C         DSEED DOUBLE PRECISION SEED FOR RANDOM NUMBER GENERATORS
C         XAVE,XSD MEAN AND STANDARD DEVIATION OF GAUSSIAN
C           DISTRIBUTION SAMPLED FOR X (MODES 1 AND 4 ONLY, DUMMY
C           OTHERWISE)
C         XIN VECTOR OF LENGTH N CONTAINING INPUT X VALUES
C           (MODES 2, 3, 5 AND 6 ONLY, DUMMY OTHERWISE)
C         YTARG TARGET FOR Y AUTOCORRELATION (MODES 4 TO 6 ONLY,
C           DUMMY OTHERWISE)
C         A,B,RXY PARAMETERS FOR THE REGRESSION MODEL (MODES 4
C           TO 6 ONLY, DUMMY OTHERWISE)
C         IWRITE IF GREATER THAN 0, USE THIS LOGICAL UNIT TO
C           PRINT USEFUL OUTPUT
C         ON RETURN
C         XAVE,XSD SAMPLE MEAN AND STANDARD DEVIATION FOR X
C         XAUTO FINAL AUTOCORRELATIONS FOR X
C         YAUTO FINAL AUTOCORRELATIONS FOR Y (MODES 4 TO 6 ONLY,
C           DUMMY OTHERWISE)
C         NSW NUMBER OF SWAPS MADE
C         X FINAL X VALUES
C         Y FINAL Y VALUES (MODES 4 TO 6 ONLY, DUMMY OTHERWISE)
C         IFAULT IS FAULT NUMBER
C         WORKING SPACE
C         SX,SY,SRT VECTORS OF LENGTH N
C
C         FAULTS
C         IFAULT =0 ON NORMAL TERMINATION
C                =1 IF EXECUTION TERMINATED BECAUSE NTRYM EXCEEDED
C
C         AUXILIARY ALGORITHMS
C         GGUBS AND GGNML ARE (IMSL) ROUTINES TO GENERATE VECTORS OF
C         INDEPENDENT RANDOM NUMBERS WITH UNIFORM AND GAUSSIAN
C         DISTRIBUTIONS RESPECTIVELY
C
      DIMENSION X(N),Y(N),SX(N),W(NMAX,NMAX),SRT(N),SY(N),U(2),XIN(N)
      DOUBLE PRECISION DSEED,DSEEDO
C
      DIFFM=0
      DSEEDO=DSEED
      IFAULT=0
      IF(MODE.NE.1.AND.MODE.NE.4)GO TO 2
C
C         MODE=1 OR 4. SAMPLE X FROM A GAUSSIAN DISTRIBUTION
C
      CALL GGNML(DSEED,N,X)
      DO 1 J=1,N
    1 X(J)=X(J)*XSD+XAVE
      GO TO 7
    2 IF(MODE.EQ.3.OR.MODE.EQ.6)GO TO 5
C
C         MODE=2 OR 5. GENERATE A VECTOR OF UNIFORM RANDOM NUMBERS
C         AND SORT X
C
      CALL GGUBS(DSEED,N,SRT)
      DO 4 J=1,N
      SMIN=2.0
      DO 3 K=1,N
      IF(SRT(K).GT.SMIN)GO TO 3
      SMIN=SRT(K)
      KA=K
    3 CONTINUE
      X(J)=XIN(KA)
```

```
    4 SRT(KA)=3.0
      GO TO 7
C
C         MODE=3 OR 6
C
    5 DO 6 J=1,N
    6 X(J)=XIN(J)
C
C         COMPUTE THE SAMPLE MEAN AND STANDARD DEVIATION
C
    7 FN=N
      SUM=0.0
      SUMSQ=0.0
      DO 8 J=1,N
      SUM=SUM+X(J)
    8 SUMSQ=SUMSQ+X(J)**2
      XAVE=SUM/FN
      XSD=SQRT(SUMSQ/FN-XAVE**2)
      IF(IWRITE.GT.0)WRITE(IWRITE,27)MODE,XTARG,NTRYM,DSEED0,XAVE,
     1XSD,(J,X(J),J=1,N)
C
C         REDUCE X TO ZERO MEAN
C
      DO 9 J=1,N
    9 X(J)=X(J)-XAVE
C
C         COMPUTE SUMS AND INITIAL AUTOCORRELATIONS
C
      SW=0.0
      DO 10 J=1,N
      SX(J)=0.0
      DO 10 K=1,N
      SW=SW+W(J,K)
   10 SX(J)=SX(J)+X(K)*W(J,K)
      XCON=1.0/(SW*XSD**2)
      XSUM=0.0
      DO 11 J=1,N
   11 XSUM=XSUM+X(J)*SX(J)
      XAUTO=XSUM*XCON
      IF(IWRITE.GT.0)WRITE(IWRITE,28)XAUTO
      IF(MODE.LT.4)GO TO 16
C
C         MODE=4 TO 6, GENERATE Y FROM THE REGRESSION MODEL
C
      SE=XSD*B*SQRT(1.0-RXY**2)/RXY
      CALL GGNML(DSEED,N,Y)
      SUM=0.0
      SUMSQ=0.0
      DO 12 J=1,N
      Y(J)=Y(J)*SE+A+B*(X(J)+XAVE)
      SUM=SUM+Y(J)
   12 SUMSQ=SUMSQ+Y(J)**2
      YAVE=SUM/FN
      YSD=SQRT(SUMSQ/FN-YAVE**2)
      DO 13 J=1,N
      SRT(J)=Y(J)
   13 Y(J)=Y(J)-YAVE
      YCON=1.0/(SW*YSD**2)
      DO 14 J=1,N
      SY(J)=0.0
      DO 14 K=1,N
   14 SY(J)=SY(J)+Y(K)*W(J,K)
      YSUM=0.0
      DO 15 J=1,N
   15 YSUM=YSUM+Y(J)*SY(J)
      YAUTO=YSUM*YCON
      IF(IWRITE.GT.0)WRITE(IWRITE,29)YTARG,A,B,RXY,YAUTO,
     1(J,SRT(J),J=1,N)
C
C         BEGIN SWAP ALGORITHM
C

   16 NTRY=-1
      NSW=0
      XDIFF=ABS(XAUTO-XTARG)
      IF(MODE.GT.3)YDIFF=ABS(YAUTO-YTARG)
      IF(MODE.GT.3)DIFFM=AMAX1(XDIFF,YDIFF)
      IF(IWRITE.GT.0)WRITE(IWRITE,30)
```

```
   17 CALL GGUBS(DSEED,2,U)
      IM=INT(U(1)*FN)+1
      IN=INT(U(2)*FN)+1
      NTRY=NTRY+1
      IF(NTRY.GE.NTRYM)GO TO 18
      IF(IM.EQ.IN)GO TO 17
      DELX=2.0*(X(IN)-X(IM))*(SX(IM)-SX(IN)-X(IN)*W(IM,IN)+X(IM)
     1*W(IN,IM))
      XSUM2=XSUM+DELX
      XNDIFF=ABS(XSUM2*XCON-XTARG)
      IF(MODE.GT.3)GO TO 19
      IF(XNDIFF.LT.XDIFF)GO TO 20
      GO TO 17
   18 IF(IWRITE.GT.0)WRITE(IWRITE,31)NTRY
      IFAULT=1
      GO TO 24
   19 DELY=2.0*(Y(IN)-Y(IM))*(SY(IM)-SY(IN)-Y(IN)*W(IM,IN)+Y(IM)
     1*W(IN,IM))
      YSUM2=YSUM+DELY
      YDIFF=ABS(YSUM2*YCON-YTARG)
      DIFFNM=AMAX1(XNDIFF,YDIFF)
      IF(DIFFNM.GE.DIFFM)GO TO 17
   20 NSW=NSW+1
      XDIFF=XNDIFF
      XSUM=XSUM2
      IF(MODE.GT.3)DIFFM=DIFFNM
      IF(MODE.GT.3)YSUM=YSUM2
C
C          UPDATE THE SUMS AND MAKE THE SWAPS
C
      DO 21 K=1,N
   21 SX(K)=SX(K)+(W(K,IM)-W(K,IN))*(X(IN)-X(IM))
      T=X(IM)
      X(IM)=X(IN)
      X(IN)=T
      XAUTO=XSUM*XCON
      IF(MODE.LE.3.AND.IWRITE.GT.0)WRITE(IWRITE,32)IM,IN,NTRY,XAUTO
      IF(MODE.LE.3)GO TO 23
      DO 22 K=1,N
   22 SY(K)=SY(K)+(W(K,IM)-W(K,IN))*(Y(IN)-Y(IM))
      T=Y(IM)
      Y(IM)=Y(IN)
      Y(IN)=T
      YAUTO=YSUM*YCON
      IF(IWRITE.GT.0)WRITE(IWRITE,32)IM,IN,NTRY,XAUTO,YAUTO
   23 NTRY=-1
      IF(MODE.LE.3.AND.XDIFF.GT.0.0001)GO TO 17
      IF(MODE.GT.3.AND.DIFFM.GT.0.0001)GO TO 17
C
C          PRINT FINAL RESULTS
C
   24 IF(IWRITE.GT.0)WRITE(IWRITE,33)NSW
      DO 25 J=1,N
      X(J)=X(J)+XAVE
      IF(MODE.GT.3)GO TO 25
      IF(IWRITE.GT.0)WRITE(IWRITE,34)J,X(J)
      GO TO 26
   25 Y(J)=Y(J)+YAVE
      IF(IWRITE.GT.0)WRITE(IWRITE,34)J,X(J),Y(J)
   26 CONTINUE
      RETURN
   27 FORMAT(46HOPROGRAM TO GENERATE SPECIFIED AUTOCORRELATION/5HOMODE,
     1I3/13H TARGET FOR X,F8.4/28H LIMIT TO UNSUCCESSFUL TRIES,I5/
     229H RANDOM NUMBER GENERATOR SEED,D12.4/12H SAMPLE MEAN,E12.4/
     326H SAMPLE STANDARD DEVIATION,E12.4/13HOINITIAL DATA/
     4(I5,E12.4))
   28 FORMAT(30HOINITIAL MORAN AUTOCORRELATION,F8.4)
   29 FORMAT(13HOTARGET FOR Y,F8.4/8H MODEL A,E12.4,6H AND B,E12.4/
     112H CORRELATION,F8.4/26H INITIAL Y AUTOCORRELATION,F8.4/
     217HOINITIAL Y VALUES/(I5,E12.4))
   30 FORMAT(1HO,19X,7HUNSUCC./46H   SWAP M  WITH N    TRIES  AUTOCORREL
     1ATION(S))
   31 FORMAT(17HO***STOPPED AFTER,I5,34H UNSUCCESSFUL TRIES AT IMPROVEME
     1NT)
   32 FORMAT(3I8,2F10.4)
   33 FORMAT(15HOFINISHED AFTER,I5,6H SWAPS/13HOFINAL VALUES)
   34 FORMAT(I10,2E12.4)
      END
```