Computers & Geosciences Vol. 15, No. 3, pp. 325–332, 1989
Printed in Great Britain

0098-3004/89 $3.00 + 0.00
Pergamon Press plc

DECLUS: A FORTRAN 77 PROGRAM FOR DETERMINING OPTIMUM SPATIAL DECLUSTERING WEIGHTS

CLAYTON DEUTSCH

Computer Services, Placer Dome Inc., 1600-1055 Dunsmuir Street, Vancouver, B.C., Canada V7X 1P1

(*Received* 18 *April* 1988; *accepted* 22 *July* 1988)

Abstract—Most data collected in the Earth Sciences are clustered preferentially. The clustering may be in high or low "grade" zones or the data may be clustered in areas accessible easily to sampling. Because all statistical and geostatistical analysis requires a distribution that is representative of the *entire* area of interest, a declustering procedure is necessary.

This paper presents a FORTRAN 77 program to compute declustering weights by a modified cell declustering procedure. An example is given and the results are compared to polygonal declustering and global kriging.

Key Words: Declustering, Unbiased distributions.

INTRODUCTION

This paper is concerned with the problem of determining a method to decluster spatially clustered data. Most measurements made in the geological sciences intend to inform some spatial domain. Usually some areas within the domain of interest have been sampled more extensively than others. In this situation, the data must be declustered to obtain a representative histogram or distribution.

Clustering occurs when the domain of interest is sampled preferentially. For example, the high-grade portion of an orebody may have twice the number of samples that are present in a lower grade part of the orebody. If all of the samples are combined with equal influence to determine the distribution, the high-grade part will have too much influence. In fact, the samples in the high-grade zone should be given one-half the influence of the samples in the low-grade zone. In this example the area with high grade (saturation, temperature, etc.) has been sampled preferentially. Another situation is when more samples are taken because they are accessible. For example, more samples of reservoir rock are available near a drilling platform than are available away from the platform.

In most instances it is undesirable or impossible to sample on a regular grid. However, if the volume has been sampled regularly, the equal weighted histogram provides the best estimate of the distribution. If the volume has *not* been sampled regularly, then we assume that there exists a set of weights that can be applied to the data to obtain a representative distribution. A datum in an area that is sampled sparsely will receive more weight than a datum in a densely sampled area. The weights then are used as frequencies of occurrence to generate the histogram or frequency distribution. *Optimal* declustering weights are defined as the weights which provide the most representative histogram.

The goal of this declustering exercise is to obtain a single overall distribution that is representative of the spatial domain in question. If the goal is a local declustered distribution representative of a subarea or block within the larger domain, one should consider indicator kriging (Journel, 1983).

The program presented in this paper uses a modified version of the cell declustering procedure described by Journel (1983). In addition to the cell declustering procedure, there are other methods to decluster data including global kriging and polygonal declustering. These two alternative methods have been considered and an example will be shown comparing the results from all three methods. A modification of the cell declustering method is proposed due to the success of the method and its computational simplicity.

THE METHOD OF CELL-DECLUSTERING

The method was proposed first by Journel (1983) as an effective method to compute spatial declustering weights. Since 1983 the method has been used widely and proven effective. The idea is to split the area of interest into a number of subareas (L) which contain the clusters of data and apply an equal weighting within each subarea followed by an average of the subarea means. The technique is illustrated by a two dimensional example in Figure 1.

Each datum falling into cell a_l receives a weight of $1/n_l$. Where a_l is the volume of cell l and n_l is the number of data within cell l. The area a_l is typically three and possibly four dimensional (with time being the fourth dimension). Within each subarea a_l the data are given an equal weight. This indicates that no declustering is done within the subareas.

A cell a_l without data does not contribute to the declustering weights. This indicates that the area informed by the declustered distribution is the area of cells that have at least one datum. This is an advan-

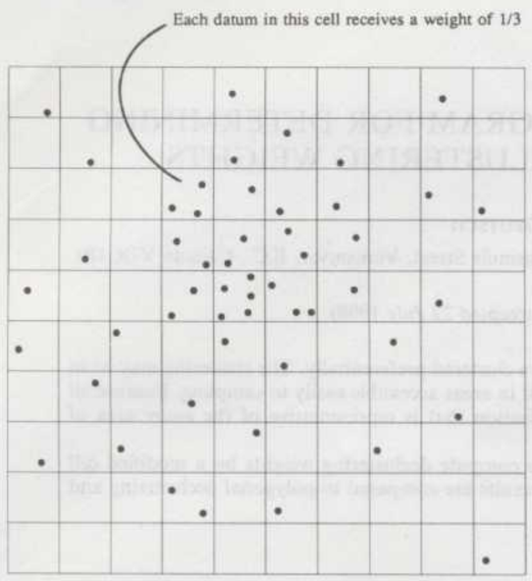


Figure 1. Two-dimensional example of cell declustering is shown. First, regular grid is constructed over area of interest. Weight applied to each datum is inverse of number of data falling in its cell.

tage because the influence of data near the edges are restricted to a maximum weight of 1/1. It also can be a disadvantage if the edges of the domain are defined clearly because the area contributing to the declustered distribution may not be the clearly defined domain. This disadvantage is mitigated by the fact that if the domain is defined clearly then the entire domain should have samples.

There are two parameters that must be defined before the declustering weights can be calculated:

(1) The cell size (i.e., rectangular size in each dimension).

(2) The origin of the cell network.

When the cell size is too small each cell will contain at most one datum, therefore, all the data will be weighted equally. Alternatively, if the cell size is too large, all the data will be in a single cell again causing the data to be weighted equally.

The method that is used to select the optimum cell size depends on the type of clustering. When the data are clustered at random (i.e., more data have been taken in accessible areas) the cell size is selected such that there is approximately one datum per cell in the sparse areas.

If it is known that the high- or low-grade areas have been "over" sampled, then the cell size can be selected such that weights provide the minimum (or maximum) weighted mean of the data. To select the minimum or maximum (as appropriate) a number of cell sizes must be considered. Usually, a plot of the weighted mean vs the cell size is created and the appropriate cell size can be selected from the graph. The cell size does not have to be square (or cubic) and separate graphs for each cell anisotropy must be constructed. Alternatively, the results can be displayed on a contour map where the cell size in two directions are used as the coordinates. If there are extreme high or low values in the data, these plots can be erratic. This is caused by the location of the origin of the cell network.

If the cell size is held constant and the origin of the cell network is moved, then the declustering weights can change considerably. This problem will be solved by systematically moving the origin for a fixed cell size. The weights obtained for each change of the origin can be averaged to obtain a unique set of weights for that cell size. This makes the declustering procedure independent of the origin. The program DECLUS allows for any number of origin offsets; however, five different offsets are usually enough.

PROGRAM DESCRIPTION

The program will decluster up to three dimensional data; however, it would be straightforward to expand the capabilities of the program. The program DECLUS reads the input parameters from an input file and creates two output files: a file containing the optimal weights and a summary file.

An example of a parameter file is shown on Figure 2. The parameters are read in free-format (the file names and the formats are read with an "a20" format) allowing the user to document the parameter file. The parameters are as follows:

- *datafl*—name of the file where the data are stored.
- *icolx, icoly, icolz, icolvr*—the column (in the data file) for the *x*, *y*, and *z* coordinates and the attribute value. If the data are two dimensional, the column number for the *z* coordinate can be set to zero or negative.
- *fmtin*—the input format. The input format may be set to (*free*) which causes the data to be read in free format.
- *znull*—missing value code. Whenever a datum with a grade of *znull* is encountered the attribute value is not considered.
- *sumfl*—name of a file for the summary.
- *weightfl*—name of a file for the optimal declustering weights. Each record will have the *x*, *y*, and *z* coordinates, the attribute value, and the declustering weight (in that order).
- *fmtout*—the output format. The output format may be set to (*free*) which causes the data to be written in free format.
- *anisy* and *anisz*—the anisotropy of the cell size in the *y* and *z* directions. For example, if *anisy* is set to 2.0, then the cell size in the *y* direction will be 2.0 times the cell size in the *x* direction.
- *minmax*—min/max switch telling whether a minimum or maximum declustered mean should be taken as the optimum declustering. Note: if only one cell size is selected, then this is not relevant.

```
                         PARAMETERS FOR DECLUS

datafl                                 'name of the data file
1  2  0  3                             'column for x,y,z, and variable
(free)                                 'format of input data (free) if free format
-1.0                                   'missing value code
cell_sum                               'name for a summary file
cell_out                               'name for an output file
(free)                                 'format of output (free) if free format
1.0    1.0                             'anisotropy for y and z
min                                    'looking for a minimum or maximum (min/max)
20     1.0   20.0                      'number of cell sizes, min size, max size
5                                      'number of origin offsets
```

Figure 2. Example of prepared parameter file.

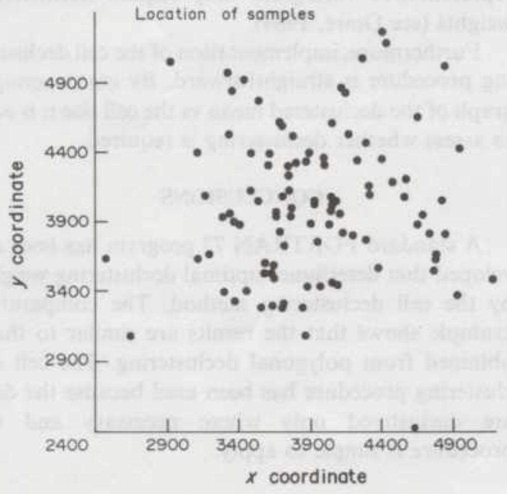
- *ncell*, *cmin*, and *cmax*—number of cell sizes that will be considered and the minimum and maximum bounds that will be used to determine the cell sizes (i.e., *ncell* cell sizes equally spaced between *cmin* and *cmax*).
- *noff*—number of origin offsets that will be considered.

The program was developed on a SUN 3/50 workstation and also tested on an IBM XT compatible (Microsoft FORTRAN 77 compiler). To the best of the author's knowledge the program adheres to the ANSI FORTRAN 77 standard. The program will identify the usual errors and notify the user of the probable cause.

EXAMPLE

A comparative example is presented to illustrate how the declustering weights obtained from DECLUS compare with other techniques. The two-dimensional example is taken from a gold mineralization containing 100 data. Figure 3 shows a location map and an equal weighted histogram of the sample values. We know that the samples are clustered in high-grade areas. Therefore, the declustered mean will be less than the unweighted mean 0.2286.

Figure 4 shows the results obtained from DECLUS for two situations: if a single origin is used and if the declustering weights are averaged for five origin offsets. The results obtained by averaging the weights are more stable than if just one origin is used. Not only are the results more stable but the minimum declustered mean is not sensitive to small changes in



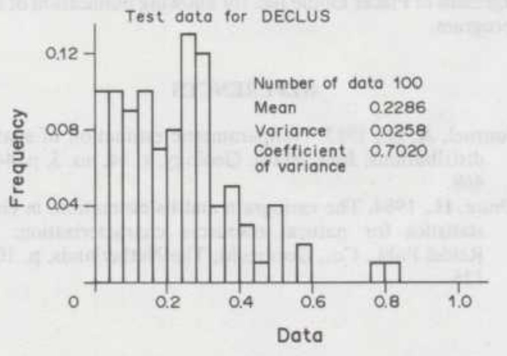
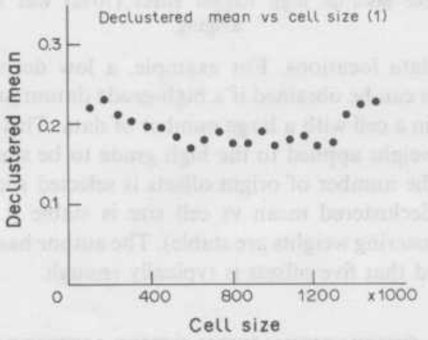


Figure 3. Location map and equal weighted histogram for test data.



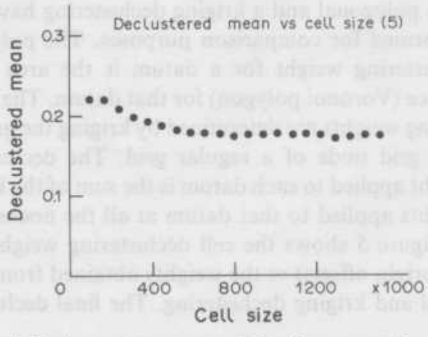
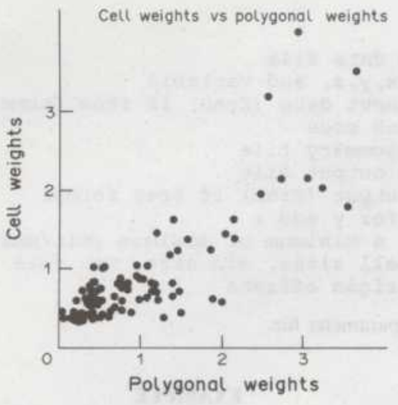


Figure 4. Declustered mean vs cell size for one origin and for five origin offsets. When declustering weights are averaged for number of origin offsets results are more stable. Declustered mean is minimum for cell size of 600.



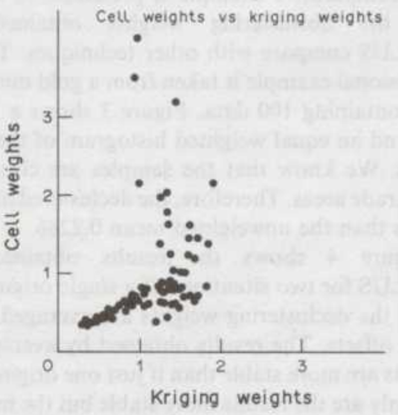


Figure 5. Cell declustering weights are plotted against kriging weights and polygonal weights. Kriging weights are less variable because high nugget effect (70%) was used in kriging.

the data locations. For example, a low declustered mean can be obtained if a high-grade datum just falls within a cell with a large number of data. This causes the weight applied to the high grade to be small.

The number of origin offsets is selected such that the declustered mean vs cell size is stable (i.e., the declustering weights are stable). The author has determined that five offsets is typically enough.

COMPARISON WITH OTHER METHODS

A polygonal and a kriging declustering have been performed for comparison purposes. The polygonal declustering weight for a datum is the area of influence (Voronoi polygon) for that datum. The global kriging weights are determined by kriging the grade at each grid node of a regular grid. The declustering weight applied to each datum is the sum of the kriging weights applied to that datum at all the nodes.

Figure 5 shows the cell declustering weights (for five origin offsets) vs the weights obtained from polygonal and kriging declustering. The final declustered means are shown here:

Equal weighted mean	0.229
Cell declustered mean	0.172
Polygonal declustered mean	0.173
Kriging declustered mean	0.197

All of the declustered means are similar. Although the true mean is unknown when a declustering exercise is performed, we are assuming that data close together are somewhat redundant. In this example, we conclude that the equal weighted mean likely over-estimates the true mean by 20–25%. The kriging declustered mean is higher than either the cell declustered or the polygonal declustered because a high nugget effect was used in the kriging.

When using polygonal declustering and global kriging, one has to decide the range of influence for the data. For example, the influence of data near the borders must be restricted, otherwise, they will receive too much weight. When the data are distributed sparsely, the cell declustering procedure assigns an equal weight (i.e., each datum is alone within a cell). This is an advantage of the cell declustering procedure; the declustering is only performed where necessary. One further disadvantage of using kriging as a declustering procedure is that a variogram is required prior to kriging. But the calculation of a representative variogram may require declustering weights (see Omre, 1984).

Furthermore, implementation of the cell declustering procedure is straightforward. By constructing a graph of the declustered mean vs the cell size it is easy to assess whether declustering is required.

CONCLUSIONS

A standard FORTRAN 77 program has been developed that determines optimal declustering weights by the cell declustering method. The comparative example shows that the results are similar to those obtained from polygonal declustering. The cell declustering procedure has been used because the data are declustered only where necessary and the procedure is simple to apply.

Acknowledgment—The author would like to thank the management of Placer Dome Inc. for allowing publication of this program.

REFERENCES

Journel, A. G., 1983, Nonparametric estimation of spatial distributions: Jour. Math. Geology, v. 14, no. 3, p. 445–468.

Omre, H., 1984, The variogram and its estimation, *in* Geostatistics for natural resources characterization: D. Reidel Publ., Co., Dordrecht, The Netherlands, p. 107–125.

APPENDIX

Program DECLUS

```
C-----------------------------------------------------------------------
C        DECLUS: A THREE DIMENSIONAL CELL DECLUSTERING PROGRAM
C        ****************************************************
C
C INPUT:
C
C  THE INPUT PARAMETERS ARE READ FROM A PARAMETER FILE WHICH THE USER
C    MUST PREPARE IN ADVANCE OF RUNNING THE PROGRAM. FOR EXAMPLE:
C
C LINE#    |START OF LINE
C   1
C   2                 PARAMETERS FOR DECLUS
C   3
C   4      DATAFL                'NAME OF THE DATA FILE
C   5      1  2  0  3            'COLUMN FOR X,Y,Z, AND VARIABLE
C   6      (FREE)                'FORMAT OF INPUT DATA (FREE) IF FREE FORMAT
C   7      -1.0                  'MISSING VALUE CODE
C   8      CELL_SUM              'NAME FOR A SUMMARY FILE
C   9      CELL_OUT              'NAME FOR AN OUTPUT FILE
C  10      (FREE)                'FORMAT OF OUTPUT (FREE) IF FREE FORMAT
C  11      1.0   1.0             'ANISOTROPY FOR Y AND Z
C  12      MIN                   'MINIMUM OR MAXIMUM (MIN/MAX)?
C  13      10  1.0  25.0         'NUMBER OF CELL SIZES, MIN SIZE, MAX SIZE
C  14      5                     'NUMBER OF ORIGIN OFFSETS
C
C OUTPUT:
C
C A SUMMARY FILE WITH THE THE CELL SIZE (IN THE X DIRECTION) AND THE
C  DECLUSTERED MEAN IS CREATED (SUMMARY FILE NOTED ABOVE). ALSO A
C  WEIGHT FILE IS CREATED WITH THE POINT COORDINATES, GRADE, AND THE
C  DECLUSTERED MEAN.
C
C PARAMETERS:
C
C     NDIM  MUST BE AS LARGE OR LARGER THAN THE NUMBER OF DATA.
C     NDIM2 MUST BE AS LARGE OR LARGER THAN THE MAXIMUM NUMBER OF CELLS.
C
      PARAMETER(NDIM=300,NDIM2=10000)
C
C AUTHOR: C. DEUTSCH                                  DATE: JANUARY 1988
C-----------------------------------------------------------------------
      REAL      X(NDIM),Y(NDIM),Z(NDIM),WT(NDIM),VR(NDIM)
      REAL      B(20),WTOPT(NDIM),CELLWT(NDIM2)
      INTEGER   INDEX(NDIM)
      CHARACTER DATAFL*20,PARFL*20,SUMFL*20,WTFL*20,FMTIN*20,FMTOUT*20
      CHARACTER MINMAX*3
      LOGICAL   FREE,DIM3,MIN
      DATA      XMIN,YMIN,ZMIN/ 9999999., 999999., 999999/
      DATA      XMAX,YMAX,ZMAX/-9999999.,-999999.,-999999/
C
C FIND OUT WHERE THE PARAMETERS ARE AND READ PARAMETERS:
C
      WRITE(*,*) 'ENTER THE NAME OF THE PARAMETER FILE:'
      READ(*,100) PARFL
      OPEN(1,FILE=PARFL,ERR=51)
      GO TO 50
 51   WRITE(*,*) 'THE FILE ',PARFL,' DOES NOT EXIST! ...TRY AGAIN'
      STOP
 50   READ(1,*)
      READ(1,*)
      READ(1,*)
      READ(1,100)        DATAFL
      READ(1,*)          ICOLX,ICOLY,ICOLZ,ICOLVR
      READ(1,100)        FMTIN
      READ(1,*)          ZNULL
      READ(1,100)        SUMFL
      READ(1,100)        WTFL
      READ(1,100)        FMTOUT
      READ(1,*)          ANISY,ANISZ
      READ(1,102)        MINMAX
      READ(1,*)          NCELL,CMIN,CMAX
      READ(1,*)          NOFF
      ROFF = REAL(NOFF)
 100      FORMAT(A20)
 102      FORMAT(A3)
          CLOSE(1)
C
C SOME QUICK ERROR CHECKING:
C
          IF(NOFF.LT.1) THEN
               WRITE(*,*) 'ERROR: NOFF < 1'
               STOP
          ENDIF
          IF(CMIN.LE.0.0) THEN
               WRITE(*,*) 'ERROR: CMIN < 0.0'
               STOP
          ENDIF
          MIN  = .TRUE.
          IF(MINMAX.EQ.'max'.OR.MINMAX.EQ.'MAX') MIN = .FALSE.
          DIM3 = .TRUE.
          IF(ICOLZ.LE.0) DIM3 = .FALSE.
C
C OPEN THE NECESSARY FILES:
C
          OPEN(1,ERR=53,FILE=DATAFL,STATUS='OLD')
          GO TO 54
 53       WRITE(*,*) '  THE FILE ',DATAFL,' DOES NOT EXIST - TRY AGAIN'
          STOP
 54       OPEN(3,FILE=SUMFL)
          OPEN(2,FILE=WTFL)
C
C READ IN THE DATA:
C
          FREE = .FALSE.
          IF(FMTIN.EQ.'(FREE)') FREE = .TRUE.
 58       NCOL=MAX0(ICOLY,ICOLX,ICOLZ,ICOLVR)
          NT = 0
 1        NT =  NT + 1
          IF(FREE) THEN
                  READ(1,*,END=2)(B(J),J=1,NCOL)
          ELSE
                  READ(1,FMTIN,END=2)(B(J),J=1,NCOL)
          ENDIF
          VR(NT) = B(ICOLVR)
          IF(VR(NT).EQ.ZNULL) THEN
                  NT = NT - 1
                  GO TO 1
          ENDIF
          X(NT)  = B(ICOLX)
          Y(NT)  = B(ICOLY)
          Z(NT)  = 0.0
          IF(DIM3) Z(NT) = B(ICOLZ)
          GO TO 1
 2        NT = NT - 1
          IF(NT.LE.1) THEN
                  WRITE(*,*) 'TOO FEW DATA TO DECLUSTER: ',NT
                  STOP
          ENDIF
          WRITE(*,*) 'THE NUMBER OF DATA FOUND = ',NT
          CLOSE(1)
C
C CHECK DIMENSION OF DATA ARRAY:
C
          IF(NT.GT.NDIM) THEN
                  WRITE(*,200) NT,NDIM
 200              FORMAT('ERROR: ARRAY DIMENSIONS TOO SMALL CHECK NDIM'/
     +                   ' NUMBER OF DATA = ',I6, ' ARRAY SIZE = ',I6)
                  STOP
          ENDIF
C
C COMPUTE MIN, MAX, AND AVERAGE:
C
          VRAV = 0.0
          DO 5 I=1,NT
                  WTOPT(I) = 1.0
                  VRAV  = VRAV + VR(I)
                  IF(X(I).LT.XMIN) XMIN=X(I)
                  IF(X(I).GT.XMAX) XMAX=X(I)
                  IF(Y(I).LT.YMIN) YMIN=Y(I)
                  IF(Y(I).GT.YMAX) YMAX=Y(I)
                  IF(DIM3) THEN
```

```
                     IF(Z(I).LT.ZMIN) ZMIN=Z(I)
                     IF(Z(I).GT.ZMAX) ZMAX=Z(I)
                ENDIF
 5       CONTINUE
         VRAV = VRAV / REAL(NT)
         WRITE(*,*) '   MINIMUM AND MAXIMUM X = ',XMIN,XMAX
         WRITE(*,*) '   MINIMUM AND MAXIMUM Y = ',YMIN,YMAX
         IF(DIM3)WRITE(*,*) '   MINIMUM AND MAXIMUM Z = ',ZMIN,ZMAX
         WRITE(*,*) 'THE UNWEIGHTED MEAN          = ',VRAV
C
C INITIALIZE THE "BEST" WEIGHT VALUES:
C
         VROP = VRAV
         BEST = 0.0
         WRITE(3,300)BEST,VROP
 300     FORMAT(2(F15.3,2X))
C
C DEFINE A "LOWER" ORIGIN TO USE FOR THE CELL SIZES:
C
         XO1 = XMIN - 0.01
         YO1 = YMIN - 0.01
         ZO1 = ZMIN - 0.01
C
C DEFINE THE INCREMENT FOR THE CELL SIZE:
C
         XINC = (CMAX-CMIN) / REAL(NCELL)
         YINC = ANISY * XINC
         ZINC = ANISZ * XINC
C
C LOOP OVER "NCELL+1" CELL SIZES IN THE GRID NETWORK:
C
         XCS = CMIN - XINC
         YCS = CMIN - YINC
         ZCS = CMIN - ZINC
         DO 6 LP=1,NCELL+1
            XCS = XCS + XINC
            YCS = YCS + YINC
            ZCS = ZCS + ZINC
C
C        INITIALIZE THE WEIGHTS TO ZERO:
C
         DO 13 I=1,NT
 13      WT(I) = 0.0
C
C        DETERMINE THE MAXIMUM NUMBER OF GRID CELLS IN THE NETWORK:
C
         NCELLX = INT((XMAX-(XO1-XCS))/XCS)+1
         NCELLY = INT((YMAX-(YO1-YCS))/YCS)+1
         IF(DIM3) THEN
                NCELLZ = INT((ZMAX-(ZO1-ZCS))/ZCS)+1
         ELSE
                NCELLZ = 1
         ENDIF
         NCELLT = NCELLX*NCELLY*NCELLZ
C
C        CHECK THE ARRAY NDIM2 DIMENSIONS:
C
         IF(NCELLT.GT.NDIM2) THEN
                WRITE(*,201) NCELLT,NDIM2
 201            FORMAT('ERROR: ARRAY DIMENSIONS TOO SMALL CHECK NDIM2'/
     +             ' NUMBER OF CELLS = ',I6, ' ARRAY SIZE = ',I6,/
     +             ' CHECK FOR OUTLIERS - INCREASE CMIN AND/OR NDIM2')
                STOP
         ENDIF
C
C        LOOP OVER ALL THE ORIGIN OFFSETS SELECTED:
C
         XFAC = XCS/ROFF
         YFAC = YCS/ROFF
         ZFAC = ZCS/ROFF
         DO 20 KP=1,NOFF
            XO = XO1 - (REAL(KP)-1.0)*XFAC
            YO = YO1 - (REAL(KP)-1.0)*YFAC
            ZO = ZO1 - (REAL(KP)-1.0)*ZFAC
C
C               INITIALIZE THE CUMULATIVE WEIGHT INDICATORS:
C
         DO 8 I=1,NCELLT
 8      CELLWT(I) = 0.0
C
C               DETERMINE WHICH CELL EACH DATUM IS IN:
C
        DO 9 I=1,NT
                ICELLX = INT((X(I) - XO)/XCS) + 1
                ICELLY = INT((Y(I) - YO)/YCS) + 1
                IF(DIM3) THEN
                      ICELLZ = INT((Z(I) - ZO)/ZCS) +1
                ELSE
                      ICELLZ = 1
                ENDIF
                ICELL  = ICELLX + (ICELLY-1)*NCELLX
     +                          + (ICELLZ-1)*NCELLY*NCELLX
                INDEX(I)       = ICELL
 9              CELLWT(ICELL) = CELLWT(ICELL) + 1.0
C
C               ACCUMULATE WEIGHT TO EACH DATUM:
C
        DO 10 I=1,NT
                IPOINT = INDEX(I)
 10             WT(I) = WT(I) + (1.0 / CELLWT(IPOINT))
 20     CONTINUE
C
C               COMPUTE THE WEIGHTED AVERAGE FOR THIS CELL SIZE:
C
        SUMW  = 0.0
        SUMWG = 0.0
        DO 14 I=1,NT
                SUMW  = SUMW  + WT(I)
 14             SUMWG = SUMWG + WT(I)*VR(I)
        VRCR  = SUMWG / SUMW
        WRITE(3,300)XCS,VRCR
C
C                SEE IF THIS WEIGHTING IS OPTIMAL:
C
        IF((MIN.AND.VRCR.LT.VROP).OR.(.NOT.MIN.AND.VRCR.GT.VROP)) THEN
                BEST = XCS
                VROP = VRCR
                DO 11 I=1,NT
 11             WTOPT(I) = WT(I)
        ENDIF
C
C                LOOP OVER ALL CELL SIZES:
C
 6      CONTINUE
        CLOSE(3)
C
C WRITE OUT THE RESULTS
C
        WRITE(*,*) 'THE DECLUSTERED MEAN      = ',VROP
        FREE = .FALSE.
        IF(FMTOUT.EQ.'(FREE)') FREE = .TRUE.
        DO 12 I = 1,NT
        IF(DIM3) THEN
                IF(FREE) THEN
                             WRITE(2,*) X(I),Y(I),Z(I),VR(I),WTOPT(I)
                ELSE
                             WRITE(2,FMTOUT) X(I),Y(I),Z(I),VR(I),WTOPT(I)
                ENDIF
        ELSE
                IF(FREE) THEN
                             WRITE(2,*) X(I),Y(I),VR(I),WTOPT(I)
                ELSE
                             WRITE(2,FMTOUT) X(I),Y(I),VR(I),WTOPT(I)
                ENDIF
        ENDIF
 12     CONTINUE
        CLOSE(2)
C
C ALL FINISHED:
C
 98     STOP
        END
```