Computers & Geosciences Vol. 16, No. 5, pp. 705–716, 1990
Printed in Great Britain.

0098-3004/90 $3.00 + 0.00

EFFICIENCY OF DIFFERENT EQUATION SOLVERS IN COKRIGING

JAMES R. CARR[1] and DONALD E. MYERS[2]

[1]Department of Geological Sciences, University of Nevada-Reno, Reno, NV 89557 and
[2]Department of Mathematics, University of Arizona, Tucson, AZ 85721, U.S.A.

(*Received* 22 *August* 1989; *accepted* 11 *December* 1989)

Abstract—When the system of equations for cokriging is written in matrix form the sample–sample covariance matrix may be considered either as an $mn \times mn$ matrix of scalar entries, where n is the number of sample locations and m is the number of variables, or as an $n \times n$ matrix whose entries are $m \times m$ matrices. Similarly, the point-sample covariance matrix may be considered as m column vectors or as a single column whose entries are $m \times m$ matrices. The formulation in the original program assumed that the submatrix structure should be preserved, but this is not necessary. The scalar matrix formulation allows for the use of a standard Gaussian elimination to reduce the matrix to diagonal form or for reduction to upper triangular form together with back substitution. Both methods result in significant reductions in computing time.

Key Words: Cokriging, COKRIG, Gauss elimination, Banded equation solution.

INTRODUCTION

Programs for solving kriging equations generally use a Gauss elimination algorithm for the reduction of the sample–sample covariance matrix and the point-sample covariance vector. In developing a computer code for cokriging (Carr, Myers, and Glass, 1985), Gauss elimination was not used at first because of the nature of the sample–sample covariance matrix and point-sample covariance vector as well as memory limitations. In cokriging, each entry in these arrays is a square matrix whose dimension is the number of variables. The initial program for solving the cokriging equations used an algorithm which maintained this submatrix structure.

Originally, Carr, Myers, and Glass (1985) used an extension of the algorithm known as ART (algebraic reconstruction technique from Herman, Lent, and Rowland, 1973). This method is iterative and is adapted easily to equation solving wherein submatrix structure is present. It was soon determined that in this form ART converges slowly; at least 200 iterations are required per estimate. Moreover, ART requires a beginning estimate for the solution; Carr, Myers, and Glass (1985) use an estimate of zero (more precisely, the null vector for weights is used). A different initial guess based on the point-sample covariance values reduces the number of iterations required.

Because the ART algorithm is slow when the initial solution is the null vector, a modified Gauss elimination technique was incorporated. In kriging, the sample–sample covariance matrix has the largest numerical values on the diagonal. In the Gauss elimination algorithm, these diagonal values are used to normalize and reduce this matrix, and the point-sample covariance vector, as a prelude to back substitution. This reduction requires division operations using diagonal covariance entries. In cokriging, the diagonal entries in the sample–sample covariance matrix are matrices, and it was desired to maintain this submatrix structure in original computer code development. A global Gauss elimination algorithm, subroutine EQSOLV in the original program, was used in which a second Gauss elimination algorithm performs the normalization and reduction using the diagonal square matrices.

Subsequently, it was noted that the submatrix structure need not be maintained in the sample–sample covariance matrix and the point-sample covariance column vector. The sample–sample covariance matrix can be treated as a large matrix of scalar entries and reduced as such in a Gauss elimination procedure. The point-sample column vector likewise is treated as m column vectors, where m is the number of variables being estimated. These vectors are reduced individually. Back substitution using each of the column vectors and the reduced covariance matrix yields the solution for the weights, one column vector at a time. It then is of interest to compare the efficiency of the two equation solvers.

COKRIGING EQUATIONS

As shown in Myers (1982), the ordinary cokriging system of equations can be written as:

$$UY = D \tag{1}$$

where U is the sample–sample covariance matrix:

$$U = \begin{bmatrix} \bar{C}(x_1 - x_1) & \dots & \bar{C}(x_1 - x_N) & I \\ \vdots & & \vdots & \vdots \\ \bar{C}(x_N - x_1) & \dots & \bar{C}(x_N - x_N) & I \\ I & \dots & I & 0 \end{bmatrix}. \quad (2)$$

Y is the solution vector:

$$Y = \begin{Bmatrix} \Gamma_1 \\ \vdots \\ \Gamma_N \\ \bar{\mu} \end{Bmatrix} \quad (3)$$

and D is the point-sample covariance vector:

$$D = \begin{Bmatrix} \bar{C}(x_1 - x_0) \\ \vdots \\ \bar{C}(x_N - x_0) \\ I \end{Bmatrix}. \quad (4)$$

In each of these arrays, I is an identity matrix and N is the number of nearest known sample locations used for estimation.

As noted in Myers (1988), the coefficient matrix (sample–sample covariance matrix) is the same whether estimating all variables or only one. This reinforces the approach utilized in the new equation solver described herein.

EQUATION SOLVERS

The original procedure in Carr, Myers, and Glass (1985) for computation of the sample–sample covariance matrix uses considerable disk access. The coefficient matrix is formed and written row by row to disk. Prior to solution, this matrix is brought into core (for the modified Gauss elimination approach: subroutine EQSOLV in Carr, Myers, and Glass, 1985). This disk I/O procedure was necessary to conform to the ART equation solvers. Because disk I/O is less efficient than in core computation of the sample–sample covariance matrix, examples are presented comparing the efficiency of the original disk I/O procedure with a modified procedure which forms this matrix in core. New or modified versions of original subroutines are listed in Appendix 2.

Numerical results

Three studies are presented. The first compares the computational efficiency of the two approaches (maintaining submatrix structure or treating all matrices as having scalar entries) using the original method of disk I/O formation of the sample–sample covariance matrix. The second repeats this comparison using in core formation of this covariance matrix. Finally, a third study is presented to show the increase in efficiency of Gauss elimination equation solution for the method in which submatrix structure is not retained. Each study uses the data set presented in Appendix 1, and preliminary output for each study is shown in Figure 1. Computational efficiency is given in Table 1.

All computations reported in this table were obtained on an IBM PC/XT computer upgraded with a 12 MHz 80286 Turbo Card with 80287-10 math coprocessor and hard disk. All programs were compiled using Microsoft FORTRAN Optimizing Compiler Version 4.0. References to IBM and Microsoft are made for information purposes only and do not imply any endorsements.

Only the method which does not maintain the submatrix structure is considered in the third and final study (the fifth column of Table 1 headed III/VI). It is clear from Table 1 that this is the most efficient method for solving the cokriging system. These results are obtained by modifying the Gauss elimination method to produce a modified, banded

Table 1. Number of locations at which estimates are made per minute; two variables are estimated per location in this example

	With Appendices as indicated					
COKRIG	ORIG	2	3/4	3/5	3/6	3/GAUSSJ
W/O coprocessor	2	3	3	8	9	—
With coprocessor	6	37	19	79	90	45

Explanation

ORIG: original version of COKRIG

2: original version of COKRIG with disk I/O but matrices treated as having scalar entries

3/4: in-core formation of covariance matrices, but submatrix structure maintained in equation solution

3/5: in-core formation of covariance matrices, but matrices treated as having scalar entries

3/6: in-core formation of covariance matrices, modified banded algorithm used for equation solution

3/GAUSSJ: test algorithm to compare solution accuracy

CPU configuration: IBM PC/XT with a 12 MHz 80286 Turbo Card; an 80287-10 (10 MHz) chip was added for the math coprocessor results shown. A math coprocessor emulator option was used for all compilings; a maximum of 10 closest data locations are used to obtain each estimate. A sector search was not used to locate these 10 closest locations

```
**********          CO-KRIGING PROGRAM          **********

                    NO OF ROWS IN KRIGED ARRAY =          10
                    NO OF COLS IN KRIGED ARRAY =          10
                    MAXIMUM Y COORDINATE       =     250.000
                    MAXIMUM X COORDINATE       =     250.000
                    INCREMENT ON X             =      25.000
                    INCREMENT ON Y             =      25.000

                    A TOTAL OF  2 VARIABLE(S) WILL BE ESTIMATED

        ********* VARIOGRAM AND CROSS-VARIOGRAM PARAMETERS    *******

                    SINGLE VARIABLE (VARIOGRAM) PARAMETERS
VARIABLE  NUGGET     SILL       RANGE      ANGLE       RATIO      INFLUENCE   MODEL

   1      9.200     15.400     14.000      .000      1.000     1000.000         2
   2       .034      .062      5.000      .000      1.000     1000.000         2

                    INTER-VARIABLE (CROSS-VARIOGRAM) PARAMETERS
VARIABLE  NUGGET     SILL       RANGE      ANGLE       RATIO      INFLUENCE   MODEL

   1      9.230     15.500     14.000      .000      1.000     1000.000         2

                              ***  INPUT DATA  ***
        X-COORD    Y-COORD                          DATA VALUES

      134.170    96.720     3.100     1.000
      131.430    92.280     4.500     1.000
      116.900    91.720     4.500     1.000
      133.280    92.280     3.500     1.000
      127.720    93.390    10.500     1.000
      123.810    97.170     3.300     1.000
      125.870    93.390    11.500     1.000
      128.180    93.390     9.600     1.000
      132.400    91.170     4.000     1.000
      127.720    92.280     9.000     1.000
      133.210   102.280     7.000     1.000
      131.440    90.050     5.750     1.000
      133.280    92.390     2.100      .000
      120.590    93.950     5.500     1.000
      132.360    91.170     5.000     1.000
      115.220    93.060     4.000     1.000
      143.860   100.390     3.600     1.000
      112.210   102.280     8.000     1.000
      141.590    94.500     4.200     1.000
      119.210    92.610     5.300     1.000
      119.110    92.610     3.000     1.000
      116.900    91.830     3.700     1.000
      111.920   103.400     5.600     1.000
      112.180   106.510    26.000     1.000
      128.370    92.610     7.300     1.000
      121.460   101.950     4.000     1.000
      116.810    91.830     5.200     1.000
      128.460    93.390     5.100     1.000
      128.600    98.950     3.000     1.000
      132.550    63.370     2.500      .000
      133.520    57.810     1.400      .000
      130.470    96.720     5.500     1.000
      129.570    93.390     7.600     1.000
      120.120    80.600     4.500     1.000
      112.490   102.280    12.000     1.000
      124.900    98.950     4.000     1.000
      120.680    93.390     7.000     1.000
      133.240    97.840     3.800     1.000
      131.420    93.390     4.500     1.000
      124.640    96.720     4.000     1.000
      124.730    96.720     4.000     1.000
      116.990    91.720     4.250     1.000
      131.420    93.170     5.400     1.000
      122.720    93.390     6.800     1.000
      129.790    87.940     5.200     1.000
      128.270    93.390    10.500     1.000

                    **** ** ***   KRIGING RESULTS   *** ** ****
ROW COL    NORTH      WEST                          DATA ESTIMATES       VARIANCE
```

Figure 1. This printout of preliminary information results for examples shown in Figures.

algorithm. This method is described in the next section.

Analogy between cokriging and finite element analysis

The Gauss elimination algorithm is used in many finite element programs, but is modified for banded matrices. Increased memory and equation solving efficiency is the result. Appendix 5 is rewritten for banded matrices after a computer algorithm given in Cook (1974, p. 45). Use of the banded matrix approach for kriging was proposed by Davis and Culhane (1984). Furthermore, Davis and Grivet (1984) propose using an LU decomposition and provide a FORTRAN subroutine for this method. The modified banded equation solver described next extends the methods previously described. Davis and Culhane (1984) correctly note that the banded matrix algorithm as implemented in finite element programs is useful only for certain covariance models and simple kriging. The modified banded algorithm presented herein, however, is applicable to any diagonalized, symmetric matrix. The code is given in Appendix 6.

In finite element analysis, arrays are formed to yield the system $AX = B$. If the finite element analysis is used to solve Hooke's law, A is a square, symmetric, diagonalized matrix of system stiffnesses, B is a column vector of forces, and X is the solution vector of deflections. A "band" (and hence the term, banded equation solver) is the area of array A on either side of the diagonal where nonzero entries are located.

In geostatistics, the kriging or cokriging system also is an $AX = B$ problem. Furthermore, the matrix A is a square, symmetric, and diagonalized matrix of covariance values. The term, "band", however, really does not have meaning for this covariance matrix because, in many instances, all entries in this matrix are nonzero. Therefore, the algorithm presented in Cook (1974, p. 45) is modified slightly to use the entire upper triangle and diagonal of the covariance matrix rather than a banded region of the upper triangle. The bandwidth is allowed to change in this modification, whereas it is a constant in finite element analysis.

This modification is presented in Appendix 6. This equation solver, used in COKRIG with AFORM replaced by Appendix 3, yields nine estimates in 1 min without a math coprocessor, 90 estimates per minute using a coprocessor. Therefore, this third and final study represents the greatest efficiency.

DISCUSSION OF SUBROUTINES

The program reported in Carr, Myers, and Glass (1985) is COKRIG. No modifications are required to reproduce the first column of Table 1 using the data of Appendix 1. To obtain the second column in this table, COKRIG is modified by simply replacing the original subroutines, EQSOLV and SCALG, with Appendix 2.

To produce the results shown in the third column of Table 1, COKRIG is modified by replacing the original subroutine, AFORM, by Appendix 3; and, the original subroutine, EQSOLV, is replaced by Appendix 4. Note that original subroutines SCALG and MATMUL are required when using Appendix 4.

To produce the results shown in the fourth column of Table 1, COKRIG is modified by replacing the original subroutine, AFORM, by Appendix 3; then, the original subroutines, EQSOLV and SCALG, are replaced by Appendix 5.

Finally, to produce the results shown in the fifth column of Table 1, COKRIG is modified by replacing the original subroutine, AFORM, by Appendix 3; then, the original subroutines, EQSOLV and SCALG, are replaced by Appendix 6.

If using the version of COKRIG from Carr and Myers (1986), do not delete subroutine, SCALG, as described. This subroutine is required for the ART algorithm in this 1986 version of COKRIG.

Four additional modifications to the main program of COKRIG are required to reproduce the results in Table 1 when Appendices 3–6 are used. These modifications follow:

(1) The labeled common, FORM, must be changed to:

```
COMMON/FORM/XMEAS(100,5),
ATEMP(100,100)
```

(this is statement COK00890 in Carr, Myers, and Glass, 1985, p. 116);

(2) The array, ATEMP(5,100) must be deleted from the DIMENSION statement at COK00910 (Carr, Myers, and Glass, 1985, p. 116);

(3) The call to subroutine AFORM must be changed to:

```
CALL AFORM(YCORD,XCORD,
NHOLE,INIT)
```

(this is statement COK02800, Carr, Myers, and Glass, 1985, p. 118);

(4) The call to subroutine EQSOLV must be changed to:

```
IF(ISOLV.EQ.1) CALL EQSOLV(XTEMP)
```

(this is statement COK02820, Carr, Myers, and Glass, 1985, p. 118).

A final study is summarized to document the numerical accuracy of the equation solution subroutines presented in Appendices 2, 5, and 6. A Gauss elimination equation solver, GAUSSJ, is presented in Numerical Recipes (Press and others, 1987, p. 28–29). The program, COKRIG, was modified using Appendix 3 and GAUSSJ, then applied to the data of Appendix 1 to document equation solution accuracy. The principle of duplication of results is used for this verification; that is, if two different subroutines yield the same results, and one of these subroutines is a published, standard subroutine such as GAUSSJ,

From Appendix VI:

1	1	237.500	12.500	7.897	1.002	18.873
1	2	237.500	37.500	8.011	1.001	19.022
1	3	237.500	62.500	7.836	1.002	18.546
1	4	237.500	87.500	8.298	1.003	18.869
1	5	237.500	112.500	7.541	1.001	18.381

From GAUSSJ:

1	1	237.500	12.500	7.897	1.002	18.873
1	2	237.500	37.500	8.011	1.001	19.022
1	3	237.500	62.500	7.836	1.002	18.546
1	4	237.500	87.500	8.298	1.003	18.869
1	5	237.500	112.500	7.541	1.001	18.381

Figure 2. Comparison of estimates yielded by two equation solvers: Appendix 6 and GAUSSJ.

the reliability of the other subroutine is substantiated. Application of COKRIG to the data of Appendix 1 using Appendix 6 is compared to that using GAUSSJ in Figure 2. Only the first five estimated values are shown, but results using both equation solvers are identical. Although only Appendix 6 is compared to GAUSSJ, Appendices 2 and 5 yield the same estimated values.

The use of standard subroutines, such as published in Press and others (1987); assures accuracy and uniformity in published software. For the applications presented here, specialty subroutines are presented, but are tested against the subroutine, GAUSSJ (Press and others, 1987, p. 28–29). The subroutine, GAUSSJ, is not modified easily for banded equation solution, which is shown here to be the most efficient form of equation solution for cokriging.

CONCLUSION

Submatrix structure is an integral part of the cokriging system of equations. This structure is maintained when forming the sample–sample covariance matrix and the point-sample covariance vector. The modified version of the subroutine, AFORM, listed in Appendix 3 maintains submatrix structure when forming these arrays.

It is not necessary, however, to maintain the submatrix structure throughout equation solving as in the original version of COKRIG. By forming the sample–sample covariance matrix in core, treating this matrix as a large matrix of scalar entries, and using only the upper triangle and diagonal of this matrix for equation solution, a substantial improvement in execution speed for COKRIG is realized.

REFERENCES

Carr, J. R., Myers, D. E., and Glass, C. E., 1985, Cokriging —a computer program: Computers & Geosciences, v. 11, no. 2, p. 111–127.

Carr, J. R., and Myers, D. E., 1986, Multivariate geostatistical analysis for the earth sciences: Short Course Notes, Univ. Missouri-Rolla, 117 p.

Cook, R. D., 1974, Concepts and applications of finite element analysis: John Wiley & Sons, New York, 402 p.

Davis, M. W., and Culhane, P. G., 1984, Contouring very large datasets using kriging, *in* Verly, G., and others, eds., Geostatistics for natural resources characterization: D. Reidel, Dordrecht, The Netherlands, v. 2, p. 599–619.

Davis, M. W., and Grivet, C., 1984, Kriging in a global neighborhood: Jour. Math. Geology, v. 16, no. 3, p. 249–266.

Herman, G. T., Lent, A., and Rowland, S. W., 1973, ART: mathematics and applications: Jour. Theor. Biology, v. 42, no. 1, p. 1–32.

Myers, D. E., 1982, Matrix formulation of co-kriging: Jour. Math. Geology, v. 14, no. 3, p. 249–257.

Myers, D. E., 1988, Multivariate geostatistics for environmental monitoring: Sciences de la Terre, v. 27, p. 411–427.

Press, W. H., Flannery, B. P., Teukolsky, S. A., and Vetterling, W. T., 1987, Numerical recipes: the art of scientific computing: Cambridge Univ. Press, New York, 818 p.

[Appendices overleaf

APPENDIX 1

```
0,2,0,1
10,10,250.0,250.0,25.0,25.0
1,2,9.20,15.4,14.,0.0,1.0,1000.0
2,2,0.034,.062,5.0,0.0,1.0,1000.0
1,2,2,9.23,15.5,14.0,0.0,1.0,1000.0
3.1,1.0,96.72,134.17
4.5,1.0,92.28,131.43
4.5,1.0,91.72,116.90
3.5,1.0,92.28,133.28
10.5,1.0,93.39,127.72
3.3,1.0,97.17,123.81
11.5,1.0,93.39,125.87
9.6,1.0,93.39,128.18
4.0,1.0,91.17,132.40
9.0,1.0,92.28,127.72
7.0,1.0,102.28,133.21
5.75,1.0,90.05,131.44
2.1,0.0,92.39,133.28
5.5,1.0,93.95,120.59
5.0,1.0,91.17,132.36
4.0,1.0,93.06,115.22
3.6,1.0,100.39,143.86
8.0,1.0,102.28,112.21
4.2,1.0,94.50,141.59
5.3,1.0,92.61,119.21
3.0,1.0,92.61,119.11
3.7,1.0,91.83,116.90
5.6,1.0,103.40,111.92
26.0,1.0,106.51,112.18
7.3,1.0,92.61,128.37
4.0,1.0,101.95,121.46
5.2,1.0,91.83,116.81
5.1,1.0,93.39,128.46
3.0,1.0,98.95,128.60
2.5,0.0,63.37,132.55
1.4,0.0,57.81,133.52
5.5,1.0,96.72,130.47
7.6,1.0,93.39,129.57
4.5,1.0,80.60,120.12
12.0,1.0,102.28,112.49
4.0,1.0,98.95,124.90
7.0,1.0,93.39,120.68
3.8,1.0,97.84,133.24
4.5,1.0,93.39,131.42
4.0,1.0,96.72,124.64
4.0,1.0,96.72,124.73
4.25,1.0,91.72,116.99
5.4,1.0,93.17,131.42
6.8,1.0,93.39,122.72
5.2,1.0,87.94,129.79
10.5,1.0,93.39,128.27
0.0,0.0,0.0,0.0
```

APPENDIX 2

```
      SUBROUTINE EQSOLV(XMEAS, XTEMP)
      COMMON /FILES/ IUNIT
      COMMON /PARM/ MROW, MCOL, MVAR, MTOT
      DIMENSION ATEMP(100,100), XTEMP(100,5), XMEAS(100,5)
C.....
C.....THIS GAUSS ELIMINATION EQUATION SOLVER REPLACES
C.....SUBROUTINE EQSOLV, CARR, ET. AL., 1985, P. 124.
C.....
      REWIND IUNIT
      DO 10 I = 1,MROW
      DO 10 J = 1,MVAR
      JB = (I - 1) * MVAR + J
      READ(IUNIT) (ATEMP(JB,JK), JK = 1,MTOT)
10    CONTINUE
      DO 15 I = 1,MTOT
      DO 15 J = 1,MVAR
      ATEMP(I,MTOT+J) = XMEAS(I,J)
15    CONTINUE
```

```
C
C......BEGIN GAUSS DECOMPOSITION
C......STEP 1:  FORWARD REDUCTION
C
       NSIZE = MTOT
       MP = NSIZE + MVAR
       DO 100 N = 1,NSIZE
       I = N + 1
          DO 90 L = 1,NSIZE
          IF (L .NE. N) THEN
             C = ATEMP(L,N) / ATEMP(N,N)
             DO 80 K = I,MP
80           ATEMP(L,K) = ATEMP(L,K) - C*ATEMP(N,K)
          ENDIF
90        CONTINUE
100    CONTINUE
C
C......STEP 2:  SOLVE FOR COKRIGING WEIGHTS
C
       DO 300 I = 1,MVAR
       DO 300 M = 1,NSIZE
          XTEMP(M,I) = ATEMP(M,NSIZE+I) / ATEMP(M,M)
300    CONTINUE
       RETURN
       END
```

APPENDIX 3

```
      SUBROUTINE AFORM (YCORD,XCORD,KHOLE,INIT)
      COMMON /DAT2/ X(500),Y(500),DAT(500,5)
      COMMON /PARM/ MROW,MCOL,MVAR,MTOT
      COMMON /AMAT/ IKRIG,JUNSAM(500),KCOUNT
      COMMON  /VAR/ CO(5),C(5),RANGE(5),MODEL(5),ANIS(5),
     2              RATIO(5),RINFLU(5)
      COMMON /CVAR/ CCO(10),CC(10),CRANGE(10),CMODEL(10),
     2              CANIS(10),CRNFLU(10),CRATIO(10)
      COMMON /FORM/ XMEAS(100,5),ATEMP(100,100)
      DIMENSION     KHOLE(INIT),IPOS(5)
      INTEGER CMODEL
      M1 = INIT
      M2 = INIT
C
C     SUBROUTINE TO FORM INTERSAMPLE COVARIANCE MATRIX
C
      DO 1000 II = 1,M1
      NI = KHOLE(II)
      I7 = (II - 1) * MVAR
      DO 750  JJ = 1,M2
      IF (JJ .LT. II)  GO TO 750
      KPOS = 0
      NK = KHOLE(JJ)
      DIFX = X(NI) - X(NK)
      DIFY = Y(NI) - Y(NK)
      K7 = (JJ - 1) * MVAR
      DISTAN = SQRT((DIFX*COS(ANIS(1)) + DIFY*SIN(ANIS(1)))**2
     2          + (RATIO(1)*(DIFY*COS(ANIS(1)) - DIFX *
     3          SIN(ANIS(1))))**2)
      DO 500 KK = 1,MVAR
      ITOT = I7 + KK
      JTOT = K7 + KK
      DO 500 LL = 1,MVAR
      LTOT = I7 + LL
      NTOT = K7 + LL
      IF (LL .EQ. KK)   THEN
         ATEMP(ITOT,NTOT) = COVAR(DISTAN,LL)
         ATEMP(NTOT,ITOT) = ATEMP(ITOT,NTOT)
      ELSEIF (LL .GT. KK)  THEN
         KPOS = KPOS + 1
         ATEMP(ITOT,NTOT) = CROSS(DISTAN,KPOS,KK,LL)
         ATEMP(LTOT,JTOT) = ATEMP(ITOT,NTOT)
         ATEMP(NTOT,ITOT) = ATEMP(ITOT,NTOT)
         ATEMP(JTOT,LTOT) = ATEMP(LTOT,JTOT)
      ENDIF
500   CONTINUE
750   CONTINUE
```

```
C
C       FORM THE IDENTITY MATRICES
C
        M5 = INIT * MVAR
        DO 900 MM = 1,MVAR
        M6 = I7 + MM
        DO 900 NM = 1,MVAR
        M7 = M5 + NM
        IF (MM .EQ. NM) ATEMP(M6,M7) = 1.0
        IF (MM .NE. NM) ATEMP(M6,M7) = 0.0
        ATEMP(M7,M6) = ATEMP(M6,M7)
900     CONTINUE
1000    CONTINUE
           M5 = INIT * MVAR + 1
        DO 1100 I = M5,MTOT
        DO 1100 J = M5,MTOT
1100    ATEMP(I,J)  =   0.0
C
C       MODIFY ATEMP FOR UNDERSAMPLING
C
        IF (IKRIG .EQ. 1)  THEN
           DO 1480 IAB = 1,M2
           NK = KHOLE(IAB)
           IMOD = 0
           DO 1420 IAC = 1,KCOUNT
           NA = JUNSAM(IAC)
           IF (NK .EQ. NA) IMOD = 1
1420       CONTINUE
           IF (IMOD .EQ. 1)  THEN
              JOUNT = 0
              DO 1440 IAD = 1,MVAR
              IF (DAT (NK,IAD) .EQ. 0.0)  THEN
                 JOUNT = JOUNT + 1
                 IPOS(JOUNT) = IAD
              ENDIF
1440          CONTINUE
              K10 = (IAB - 1) * MVAR
              DO 1450 IAE = 1,JOUNT
              KZ2 = K10 + IPOS(IAE)
              DO 1450 IAF = 1,MTOT
              ATEMP(KZ2,IAF) = 0.0
1450          ATEMP(IAF,KZ2) = 0.0
           ATEMP(KZ2,KZ2) = 10000000000.0
           ENDIF
1480       CONTINUE
        ENDIF
C
C       FORM THE MEASUREMENT VECTOR
C
        DO 2000 II = 1,M1
        K7 = KHOLE(II)
        DIFX = XCORD - X(K7)
        DIFY = YCORD - Y(K7)
        KPOS = 0
        K12 = (II - 1) * MVAR
        DISTAN = SQRT((DIFX*COS(ANIS(1)) + DIFY*SIN(ANIS(1)))**2
     2           + (RATIO(1)*(DIFY*COS(ANIS(1)) - DIFX *
     3           SIN(ANIS(1))))**2)
        DO 2000 JJ = 1,MVAR
        KZ = K12 + JJ
        DO 2000 KK = 1,MVAR
        KW = K12 + KK
        IF (KK .EQ. JJ)  THEN
           XMEAS(KZ,KK) = COVAR(DISTAN,KK)
        ELSEIF (KK .GT. JJ)  THEN
           KPOS = KPOS + 1
           XMEAS(KZ,KK) = CROSS(DISTAN,KPOS,JJ,KK)
           XMEAS(KW,JJ) = XMEAS(KZ,KK)
        ENDIF
2000    CONTINUE
C
C       MODIFY THE MEASUREMENT VECTOR FOR UNDERSAMPLING
C
        IF (IKRIG .EQ. 1)  THEN
           DO 2350 II = 1,M1
           IMOD = 0
           K7 = KHOLE(II)
           DO 2200 JJ = 1,KCOUNT
```

```
            NA = JUNSAM(JJ)
2200        IF (K7 .EQ. NA) IMOD = 1
            IF (IMOD .EQ. 1)  THEN
               JOUNT = 0
               DO 2250 KK = 1,MVAR
               IF (DAT(K7,KK) .EQ. 0.0)  THEN
                  JOUNT = JOUNT + 1
                  IPOS(JOUNT) = KK
               ENDIF
2250           CONTINUE
               K12 = (II - 1) * MVAR
               DO 2260 LL = 1,JOUNT
               K8 = K12 + IPOS(LL)
               DO 2260 MM = 1,MVAR
               XMEAS(K8,MM) = 0.0
2260           CONTINUE
            ENDIF
2350        CONTINUE
         ENDIF
C
C        LAST ENTRY IN THE MEASUREMENT VECTOR IS AN IDENTITY MATRIX
C
         N1 = M1 * MVAR
         DO 2500 II = 1,MVAR
         N3 = N1 + II
         DO 2500 JJ = 1,MVAR
         IF (JJ .EQ. II)  THEN
            XMEAS(N3,JJ) = 1.0
         ELSEIF (JJ .NE. II)  THEN
            XMEAS(N3,JJ) = 0.0
         ENDIF
2500     CONTINUE
         RETURN
         END
```

APPENDIX 4

```
         SUBROUTINE EQSOLV (XTEMP)
         COMMON /PARM/ MROW,MCOL,MVAR,MTOT
         COMMON /FORM/ XMEAS(100,5),ATEMP(100,100)
         DIMENSION     XTEMP(100,5)
         DIMENSION     TEMP(5,5),TEMP1(5,5),TEMP2(5,5),TEMP3(5,5)
C
C        SUBROUTINE FOR MODIFIED GAUSS ELIMINATION
C
         ISET = 1
         DO 15 I = 1,MTOT
         DO 15 J = 1,MVAR
         N = MTOT + J
         ATEMP(I,N) = XMEAS(I,J)
15       CONTINUE
C
C        BEGIN GAUSS DECOMPOSITION
C
         MN = MROW + 1
         DO 100 I = 1,MROW
         IP = I + 1
         KB1 = (I - 1) * MVAR
         DO 20 KJ = 1,MVAR
         KB = KB1 + KJ
         DO 20 KK = 1,MVAR
         KC = KB1 + KK
20       TEMP(KJ,KK) = ATEMP(KB,KC)
         DUM1 = TRACE(TEMP)
         DO 100 J = 1,MROW
         IF (I - J)  40,100,40
40       CONTINUE
         KC1 = (J - 1) * MVAR
         DO 50 KJ = 1,MVAR
         KB = KC1 + KJ
         DO 50 KK = 1,MVAR
         KC = KB1 + KK
50       TEMP1(KJ,KK) = ATEMP(KB,KC)
         IF (MVAR .EQ. 1)  THEN
            DO 70 KJ = 1,MVAR
            DO 70 KK = 1,MVAR
```

```
70          TEMP1(KJ,KK) = - TEMP1(KJ,KK)  /  DUM1
          ELSE
             CALL SCALG(TEMP,TEMP1,TEMP3)
             DO 72 KJ = 1,MVAR
             DO 72 KK = 1,MVAR
72           TEMP1(KJ,KK) = - TEMP3(KJ,KK)
          ENDIF
          DO 90 K = IP,MN
          JB1 = (K - 1) * MVAR
          DO 80 KA = 1,MVAR
          JA = KB1 + KA
          DO 80 KB = 1,MVAR
          JB = JB1 + KB
80        TEMP2(KA,KB) = ATEMP(JA,JB)
          IF (ISET .EQ. 0)  THEN
             CALL MATMUL(TEMP1,TEMP2,TEMP3,MVAR,MVAR,MVAR)
          ELSE
             CALL MATMUL(TEMP2,TEMP1,TEMP3,MVAR,MVAR,MVAR)
          ENDIF
          DO 90 KA = 1,MVAR
          JA = KC1 + KA
          DO 90 KB = 1,MVAR
          JB = JB1 + KB
90        ATEMP(JA,JB) = ATEMP(JA,JB) + TEMP3(KA,KB)
100       CONTINUE
          DO 200 I = 1,MROW
          KB1 = (I - 1) * MVAR
          DO 150 KJ = 1,MVAR
          KB = KB1 + KJ
          DO 150 KK = 1,MVAR
          KC = KB1 + KK
150       TEMP(KJ,KK) = ATEMP(KB,KC)
          DUM2 = TRACE(TEMP)
          DO 160 J = 1,MVAR
          JA = KB1 + J
          DO 160 K = 1,MVAR
          JB = MTOT + K
160       TEMP2(J,K) = ATEMP(JA,JB)
          IF (MVAR .EQ. 1)  THEN
             DO 170 J = 1,MVAR
             JA = KB1 + J
             DO 170 K = 1,MVAR
             JB = MTOT + K
170          XTEMP(JA,K) = ATEMP(JA,JB) / DUM2
          ELSE
             CALL SCALG(TEMP,TEMP2,TEMP3)
             DO 185 J = 1,MVAR
             JA = KB1 + J
             DO 185 K = 1,MVAR
185          XTEMP(JA,K) = TEMP3(J,K)
          ENDIF
200       CONTINUE
          RETURN
          END
```

APPENDIX 5

```
          SUBROUTINE EQSOLV(XTEMP)
          COMMON /FORM/ XMEAS(100,5), ATEMP(100,100)
          COMMON /PARM/ MROW, MCOL, MVAR, MTOT
          DIMENSION XTEMP(100,5)
C.....
C.....THIS GAUSS ELIMINATION EQUATION SOLVER REPLACES
C.....SUBROUTINE EQSOLV, CARR, ET. AL., 1985, P. 124.
C.....
          DO 15 I = 1,MTOT
          DO 15 J = 1,MVAR
          ATEMP(I,MTOT+J) = XMEAS(I,J)
15        CONTINUE
C
C.....BEGIN GAUSS DECOMPOSITION
C.....STEP 1:  FORWARD REDUCTION
C
```

```
      NSIZE = MTOT
      MP = NSIZE + MVAR
      DO 100 N = 1,NSIZE
      I = N + 1
         DO 90 L = 1,NSIZE
         IF (L .NE. N) THEN
            C = ATEMP(L,N) / ATEMP(N,N)
            DO 80 K = I,MP
80          ATEMP(L,K) = ATEMP(L,K) - C*ATEMP(N,K)
         ENDIF
90       CONTINUE
100   CONTINUE
C
C.....STEP 2:  SOLVE FOR COKRIGING WEIGHTS
C
      DO 300 I = 1,MVAR
      DO 300 M = 1,NSIZE
         XTEMP(M,I) = ATEMP(M,NSIZE+I) / ATEMP(M,M)
300   CONTINUE
      RETURN
      END
```

APPENDIX 6

```
      SUBROUTINE EQSOLV( XTEMP)
      COMMON /FORM/ XMEAS(100,5), ATEMP(100,100)
      COMMON /PARM/ MROW, MCOL, MVAR, MTOT
      DIMENSION XTEMP(100,5)
      DIMENSION BUF(100), XSTOR(100,5)
C.....
C.....THIS BANDED GAUSS ELIMINATION EQUATION SOLVER REPLACES
C.....SUBROUTINE EQSOLV, CARR, ET. AL., 1985, P. 124.
C.....
      DO 5 I = 1,MTOT
      DO 5 J = 1,MVAR
      XSTOR(I,J) = XMEAS(I,J)
5     CONTINUE
C
C.....BEGIN GAUSS DECOMPOSITION
C.....STEP 1:  FORWARD REDUCTION
C
      NSIZE = MTOT
      MBAND = MTOT + 1
      DO 100 N = 1,NSIZE
         LL = N + 1
      IF (ATEMP(N,N) .EQ. 0.0)  GO TO 100
         DO 90 L = LL,MBAND
            C = ATEMP(N,L) / ATEMP(N,N)
            J = L - 1
            DO 80 K = L,MBAND
            J = J + 1
80          ATEMP(L,J) = ATEMP(L,J) - C*ATEMP(N,K)
            ATEMP(N,L) = C
90       CONTINUE
100   CONTINUE
C
C.....STEP 2:  REDUCE THE MEASUREMENT VECTOR
C
      DO 200 K = 1,MVAR
      DO 200 N = 1,NSIZE
         LL = N + 1
         DO 190 L = LL,MBAND
         IF (ATEMP(N,L) .NE. 0.0)  THEN
            XMEAS(L,K) = XMEAS(L,K) - ATEMP(N,L) *
     2                   XMEAS(N,K)
         ENDIF
190      CONTINUE
      IF (ATEMP(N,N) .EQ. 0.0) GO TO 200
      XMEAS(N,K) = XMEAS(N,K) / ATEMP(N,N)
200   CONTINUE
C
C.....STEP 3:  SOLVE FOR COKRIGING WEIGHTS
C
      DO 205 I = 1,NSIZE
      DO 205 J = 1,MVAR
      XTEMP(I,J) = XMEAS(I,J)
```

```
205     CONTINUE
        DO 300 I = 1,MVAR
        DO 300 M = 2,NSIZE
        N = NSIZE + 1 - M
        LL = N + 1
        DO 290 L = LL,MBAND
        IF (ATEMP(N,L) .NE. 0.0)  THEN
           XTEMP(N,I) = XTEMP(N,I) - ATEMP(N,L) *
     2                  XTEMP(L,I)
        ENDIF
290     CONTINUE
300     CONTINUE
C
C       RESTORE THE MEASUREMENT VECTOR
C
        DO 1000 I = 1,MTOT
        DO 1000 J = 1,MVAR
        XMEAS(I,J) = XSTOR(I,J)
1000    CONTINUE
        RETURN
        END
```