A COMPARISON OF ALGORITHMS FOR LEAST SQUARES ESTIMATES OF PARAMETERS IN THE LINEAR MODEL

by

CHUL H. AHN

B. S., Seoul National University, 1976

A MASTER'S REPORT

submitted in partial fulfillment of the requirements for the degree

MASTER OF SCIENCE

Department of Statistics

KANSAS STATE UNIVERSITY
Manhattan, Kansas

1981

Approved by:

[Signature]
Major Professor
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Acknowledgements</td>
<td>11</td>
</tr>
<tr>
<td>I. Introduction</td>
<td>1</td>
</tr>
<tr>
<td>II. Gaussian Elimination</td>
<td>2</td>
</tr>
<tr>
<td>1. Triangular Decomposition</td>
<td>2</td>
</tr>
<tr>
<td>2. Matrix Inversion</td>
<td>5</td>
</tr>
<tr>
<td>3. Formulation of $A^{-1} = U^{-1}L^{-1}$</td>
<td>6</td>
</tr>
<tr>
<td>4. Back Substitution</td>
<td>7</td>
</tr>
<tr>
<td>III. Cholesky Method</td>
<td>8</td>
</tr>
<tr>
<td>1. Cholesky Decomposition</td>
<td>9</td>
</tr>
<tr>
<td>2. Matrix Inversion</td>
<td>11</td>
</tr>
<tr>
<td>3. Back Substitution</td>
<td>13</td>
</tr>
<tr>
<td>IV. Sweep Operator</td>
<td>14</td>
</tr>
<tr>
<td>V. Comparison Based on the Number of Arithmetic Operations</td>
<td>19</td>
</tr>
<tr>
<td>VI. Results and Discussion</td>
<td>20</td>
</tr>
<tr>
<td>1. Condition Number of $X'X$</td>
<td>20</td>
</tr>
<tr>
<td>2. Generating $X'X$ matrix</td>
<td>21</td>
</tr>
<tr>
<td>3. Accuracy</td>
<td>22</td>
</tr>
<tr>
<td>4. Execution Time</td>
<td>23</td>
</tr>
<tr>
<td>References</td>
<td>25</td>
</tr>
<tr>
<td>Appendices</td>
<td>27</td>
</tr>
<tr>
<td>Abstract</td>
<td></td>
</tr>
</tbody>
</table>
THIS BOOK CONTAINS NUMEROUS PAGES WITH DIAGRAMS THAT ARE CROOKED COMPARED TO THE REST OF THE INFORMATION ON THE PAGE. THIS IS AS RECEIVED FROM CUSTOMER.
ACKNOWLEDGEMENTS

To Dr. K. E. Kemp I wish to express my gratitude for his guidance and advice in the preparation of this report. Gratitude is also expressed to Dr. D. E. Johnson and Dr. R. M. Rubison for their serving on the advisory committee and to Dr. G. A. Milliken for allowing me to use his computer program for generating a correlation matrix.

To my parents and wife I wish to express my appreciation. They have provided tremendous support.
I. Introduction

Most linear models applications require a solution for the normal equations: $X'Xb = X'Y$, which frequently requires an inverse of the $X'X$ matrix. This report compares three methods of finding the inverse of $X'X$ and the solution vector, $b$: Gaussian elimination, Cholesky method, and Sweep operator. Relative efficiencies for each method are computed and are based on the number of arithmetic operations and computation time each method requires compared to others. The accuracy of the computed inverse, and hence the solution vector as well, is obtained empirically.
II. Gaussian Elimination

Gaussian elimination is the process of subtracting suitable multiples of each row of the system of equation from all succeeding rows to produce a matrix with zeros below and to the left of the main diagonal. The method first developed by Gauss early in the nineteenth century was used to solve systems of linear equations. But it can also be used to find the inverse of a matrix. From the normal equations: \( X^TXb = X^TY \), let \( X^TX = A \) and \( X^TY = C \). Then the problem is to find the inverse, \( A^{-1} \), and the solution, \( b \), in \( Ab = C \). The following steps are required to find \( A^{-1} \) and \( b \).

1. \( A \) is decomposed into two triangular matrices, \( L \) and \( U \)
2. \( L \) and \( U \) are inverted.
3. \( A^{-1} \) is formed by premultiplying \( L^{-1} \) by \( U^{-1} \)
4. \( b \) is obtained from \( Ly = C \) and \( Ub = y \).

Step 1 is known as triangular decomposition, the most commonly used variant of Gaussian elimination.

1. Triangular Decomposition

A matrix \( A \) is first decomposed into a lower triangular matrix with unit diagonal, \( L \), and an upper triangular matrix, \( U \), in the following manner:

\[
A = LU \quad (1)
\]

The computation of \( L \) and \( U \) is illustrated by the following example. Consider the case of a 4 x 4 matrix. We have to find the elements \( l_{ij} \) and \( u_{ij} \) corresponding to known elements for which the following matrix equation holds.
\[
\begin{bmatrix}
1 & 0 & 0 & 0 \\
1_{21} & 1 & 0 & 0 \\
1_{31} & 1_{32} & 1 & 0 \\
1_{41} & 1_{42} & 1_{43} & 1
\end{bmatrix}
\begin{bmatrix}
u_{11} & u_{12} & u_{13} & u_{14} \\
0 & u_{22} & u_{23} & u_{24} \\
0 & 0 & u_{33} & u_{34} \\
0 & 0 & 0 & u_{44}
\end{bmatrix}
= 
\begin{bmatrix}
a_{11} & a_{12} & a_{13} & a_{14} \\
a_{12} & a_{22} & a_{23} & a_{24} \\
a_{13} & a_{23} & a_{33} & a_{34} \\
a_{14} & a_{24} & a_{34} & a_{44}
\end{bmatrix}
\] (2)

From (2) we have 10 independent equations.

\[u_{11} = a_{11}\]
\[u_{12} = 1_{21} u_{11} = a_{12}\]
\[u_{13} = 1_{31} u_{11} = a_{13}\]
\[u_{14} = 1_{41} u_{11} = a_{14}\]

(3)

giving successively \(u_{11}, u_{12}, u_{13}, u_{14}, 1_{21}, 1_{31},\) and \(1_{41}\);

\[1_{21} u_{12} + u_{22} = a_{22}\]
\[1_{21} u_{13} + u_{23} = 1_{31} u_{12} + 1_{32} u_{22} = a_{23}\]
\[1_{21} u_{14} + u_{24} = 1_{41} u_{12} + 1_{42} u_{22} = a_{24}\]

(4)

giving successively \(u_{22}, u_{23}, u_{24}, 1_{32},\) and \(1_{42}\);

\[1_{31} u_{13} + 1_{32} u_{23} + u_{33} = a_{33}\]
\[1_{31} u_{14} + 1_{32} u_{24} + u_{34} = 1_{41} u_{13} + 1_{42} u_{23} + 1_{43} u_{33} = a_{34}\]

(5)

giving successively \(u_{33}, u_{34},\) and \(1_{43}\);

\[1_{41} u_{14} + 1_{42} u_{24} + 1_{43} u_{34} + u_{44} = a_{44}\]

(6)

giving \(u_{44}\).

The basic steps in the triangular decomposition are

\[u_{ij} = a_{ij} - \sum_{k=1}^{i-1} l_{ik} k_{kj} \quad i = 1, \ldots, j-1 \]
\[j = 1, \ldots, n\]

(7)
\[ l_{ij} = \frac{a_{ij} - \sum_{k=1}^{j-1} l_{ik} u_{kj}}{u_{jj}} \quad i = 1, \ldots, n \]
\[ j = 1, \ldots, i-1 \]  
(8)

From (7) and (8), \((i-1)\) multiplications and \((i-1)\) additions are needed to compute \(u_{ij}\) and \(j\) multiplications and \((j-1)\) additions to compute \(l_{ij}\).

In the case of a 4x4 matrix,

\[
\bar{U} = \begin{bmatrix}
0 & 0 & 0 & 0 \\
0 & 1 & 1 & 1 \\
0 & 0 & 2 & 2 \\
0 & 0 & 0 & 3
\end{bmatrix} \quad \bar{L} = \begin{bmatrix}
0 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 \\
1 & 2 & 0 & 0 \\
1 & 2 & 3 & 0
\end{bmatrix}
\]

where each element \(\bar{u}_{ij}\) and \(\bar{l}_{ij}\) denotes the number of multiplications required to compute its own.

From (7) and (8) the number of multiplications required to decompose \(A\) into \(L\) and \(U\) is given by:

\[
\sum_{j=1}^{n} \sum_{i=1}^{j} (i-1) + \sum_{i=1}^{n} \sum_{j=1}^{i-1} j = \frac{n^3}{3} - \frac{n}{3}
\]
(9)

Similarly, the number of additions is given by:

\[
\sum_{j=1}^{n} \sum_{i=1}^{j} (i-1) + \sum_{i=1}^{n} \sum_{j=1}^{i-1} (j-1) = \frac{n^3}{3} - \frac{n^2}{2} + \frac{n}{6}
\]
(10)
2. Matrix Inversion

If $U^{-1}$ has the elements $v_{ij}$, then by definition of the inverse we have

$$
\sum_{k=1}^{n} u_{ik} v_{kj} = \delta_{ij} = \begin{cases} 
1 & \text{if } i=j \\
0 & \text{otherwise}
\end{cases}
$$

e.g. if $n=4$, we have to find the $v_{ij}$ such that

$$
\begin{bmatrix}
    u_{11} & u_{12} & u_{13} & u_{14} \\
    0 & u_{22} & u_{23} & u_{24} \\
    0 & 0 & u_{33} & u_{34} \\
    0 & 0 & 0 & u_{44}
\end{bmatrix}
\begin{bmatrix}
    v_{11} & v_{12} & v_{13} & v_{14} \\
    0 & v_{22} & v_{23} & v_{24} \\
    0 & 0 & v_{33} & v_{34} \\
    0 & 0 & 0 & v_{44}
\end{bmatrix}
= 
\begin{bmatrix}
    1 & 0 & 0 & 0 \\
    0 & 1 & 0 & 0 \\
    0 & 0 & 1 & 0 \\
    0 & 0 & 0 & 1
\end{bmatrix}
$$

To calculate $v_{ij}$ we proceed in the following manner.

First for $i=1,...,n$ calculate $v_{ii} = 1/u_{ii}$. Then calculate $v_{in}$ in the order $i=n,...,1$. The next step is to calculate $v_{i,n-1}$ in the order $i=(n-1),...,1$, etc.

The elements of $U^{-1}$ are, thus, given by

$$
v_{ii} = \frac{1}{u_{ii}} \tag{11}
$$

$$
v_{ij} = -v_{ii} \sum_{p=i+1}^{j} u_{ip} v_{pj} \quad , \quad i \neq j \tag{12}
$$

The calculation of $n$ reciprocals used in (11), is performed as part of the decomposition of $A$ into $L$ and $U$. To compute $v_{ij}$, $i \neq j$, in (12), requires $(j-i+1)$ multiplications and $(j-i-1)$ additions which gives the number of multiplications as
\[
\sum_{j=1}^{n} \sum_{i=1}^{j-1} (j-i+1) = \frac{n(n-1)(n+4)}{6}
\]  \hspace{1cm} (13)

Similarly the number of additions is
\[
\sum_{j=1}^{n} \sum_{i=1}^{j-1} (j-i-1) = \frac{n(n-1)(n-2)}{6}
\]  \hspace{1cm} (14)

The same analysis is applicable to \( L^{-1} \) but as the diagonal elements of \( L \) are unity the number of multiplications is reduced by \( n(n-1) \).

To invert \( L \) requires
\[
\frac{n(n-1)(n-2)}{6} \quad \text{multiplications}
\]  \hspace{1cm} (15)

\[
\frac{n(n-1)(n-2)}{6} \quad \text{additions}
\]  \hspace{1cm} (16)

3. The formation of \( A^{-1} = U^{-1}L^{-1} \)

The formation of \( A^{-1} = U^{-1}L^{-1} \) is the reverse process of the triangular decomposition of \( A \) into \( L \) and \( U \). Let \( b_{ij} \) denote the elements of \( A^{-1} \), \( v_{ij} \) the elements of \( U^{-1} \) and \( m_{ij} \) the elements of \( L^{-1} \). Then the basic steps in the formation of \( A^{-1} = U^{-1}L^{-1} \) are

\[
b_{ij} = v_{ij} + \sum_{k=1}^{i-1} v_{ik} m_{kj} , \quad i \leq j
\]  \hspace{1cm} (17)

\[
b_{ij} = v_{jj} m_{jj} + \sum_{k=1}^{j-1} v_{ik} m_{kj} , \quad i > j
\]  \hspace{1cm} (18)
Hence, it may be shown that the formation of $A^{-1} = U^{-1}L^{-1}$ requires
the number of multiplications given as

$$\sum_{j=1}^{n} \sum_{i=1}^{j} (i-1) + \sum_{i=1}^{n} \sum_{j=1}^{i-1} j = \frac{n^3}{3} - \frac{n}{3} \tag{19}$$

and the number of additions given as

$$\sum_{j=1}^{n} \sum_{i=1}^{j} (i-1) + \sum_{i=1}^{n} \sum_{j=1}^{i-1} (j-1) = \frac{n^3}{3} - \frac{n^2}{2} + \frac{n}{6} \tag{20}$$

4. Back Substitution

To complete the solution of $A = C$, the following steps are required:

( i ) $y$ is evaluated from $Ly = C$.

(ii) $b$ is evaluated from $Ub = y$.

(i) can be written in the form:

$$l_{11}y_1 = c_1$$

$$l_{21}y_1 + l_{22}y_2 = c_2$$

$$\ldots \ldots \ldots \ldots \ldots$$

$$l_{n1}y_1 + l_{n2}y_2 + \ldots + l_{nn}y_n = c_n$$

The variables $y_1, y_2, \ldots, y_n$ are determined in succession from
the first, second, \ldots, $n$th equations, respectively,

i.e. $y_1 = c_1 / l_{11}$

$$y_2 = (c_2 - l_{22}y_1) / l_{22}, \text{ etc.}$$

For a general step the evaluation of $y_k$ requires one reciprocal, $k$
multiplications and $k-1$ additions. Hence to solve the system of
equations $Ly = C$ requires $n(n+1)/2$ multiplications and $n(n-1)/2$ addi-
itions. The solution of $Ub = y$ requires the same amount of arithmetic.
However in our case all \( l_{ii} = 1 \) and so the number of multiplications required is reduced by \( n \). The amount of arithmetic required in the back substitution phase is

\[
\begin{align*}
&n^2 \quad \text{multiplications} \\
&n^2 - n \quad \text{additions.}
\end{align*}
\] (21)

As shown in the table 1, the total number of arithmetic operations to find the inverse and the solution is

\[
\begin{align*}
&n^3 + n^2 - n \quad \text{multiplications} \\
&n^3 - n \quad \text{additions}
\end{align*}
\] (22)

Table 1. Number of computations required for solution vector and inverse matrix using Gaussian elimination

<table>
<thead>
<tr>
<th>Step</th>
<th>Multiplications</th>
<th>Additions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Decompose A into L &amp; U</td>
<td>( (n^3 - n)/3 )</td>
<td>( (2n^3 - 3n^2 + n)/6 )</td>
</tr>
<tr>
<td>Invert U</td>
<td>( n(n-1)(n+4)/6 )</td>
<td>( n(n-1)(n-2)/6 )</td>
</tr>
<tr>
<td>Invert L</td>
<td>( n(n-1)(n-2)/6 )</td>
<td>( n(n-1)(n-2)/6 )</td>
</tr>
<tr>
<td>Formation of ( A^{-1} = U^{-1}L^{-1} )</td>
<td>( (n^3 - n)/3 )</td>
<td>( (2n^3 - 3n^2 + n)/6 )</td>
</tr>
<tr>
<td>Back Substitution</td>
<td>( n^2 )</td>
<td>( n^2 - n )</td>
</tr>
<tr>
<td>Total</td>
<td>( n^3 + n^2 - n )</td>
<td>( n^3 - n^2 )</td>
</tr>
</tbody>
</table>

III. Cholesky Method

The Cholesky method is used when the matrix \( A \) is symmetric and positive definite. It uses a decomposition which transforms the matrix \( A \) into the product of a lower triangular matrix, \( L \), and the transpose of itself, \( L^T \), i.e. \( A = LL^T \). The proof of \( A = LL^T \), due to Cholesky, is by induction and may be found in Fox(1954) or Wilkinson(1965).
A particularly useful property of Cholesky decomposition is that if the matrix is scaled originally so that all elements of A are bounded by unity then so, too, are all elements of L, and no pivoting is required. The following processes are given for finding the inverse, $A^{-1}$, and the solution, X, in $Ab=C$.

1. $A$ is decomposed into $L$ and $L^T$.
2. $A^{-1}$ is obtained from $L^T A^{-1} = L^{-1}$.
3. $b$ is obtained from $Ly=C$ and $L^T b=y$.

1. Cholesky Decomposition

A symmetric and positive definite matrix $A$ is transformed as $A=LL^T$. (1)

The computation of $L$ is illustrated by the following example.

Consider the case of a $4 \times 4$ matrix. We have to find the elements $l_{ij}$ corresponding to known elements $a_{ij}$ for which the following matrix equation holds.

\[
\begin{bmatrix}
  l_{11} & 0 & 0 & 0 \\
  l_{21} & l_{22} & 0 & 0 \\
  l_{31} & l_{32} & l_{33} & 0 \\
  l_{41} & l_{42} & l_{43} & l_{44}
\end{bmatrix}
\begin{bmatrix}
  a_{11} & a_{12} & a_{13} & a_{14} \\
  a_{12} & a_{22} & a_{23} & a_{24} \\
  a_{13} & a_{23} & a_{33} & a_{34} \\
  a_{14} & a_{24} & a_{34} & a_{44}
\end{bmatrix}
= \begin{bmatrix}
  l_{11} & l_{21} & l_{31} & l_{41} \\
  0 & l_{22} & l_{32} & l_{42} \\
  0 & 0 & l_{33} & l_{43} \\
  0 & 0 & 0 & l_{44}
\end{bmatrix}
\]

From (2), we have 10 independent equations.

\[
\begin{align*}
  l_{11} &= a_{11}, \\
  l_{11}l_{21} &= l_{21}l_{11} = a_{12}, \\
  l_{11}l_{31} &= l_{31}l_{11} = a_{13}, \\
  l_{11}l_{41} &= l_{41}l_{11} = a_{14},
\end{align*}
\]

(3)
giving successively $l_{11}$, $l_{12}$, $l_{13}$, and $l_{14}$:

\[
\begin{align*}
  l_{11}^2 &= a_{11}, \\
  l_{21}^2 + l_{22}^2 &= a_{22} \\
  l_{21}l_{31} + l_{22}l_{32} &= l_{31}l_{21} + l_{32}l_{22} = a_{23},
\end{align*}
\]
\[ l_{21}l_{41} + l_{22}l_{42} = l_{41}l_{21} + l_{42}l_{22} = a_{24}, \]  
\[ (4) \]
giving successively \( l_{22}, l_{32}, \) and \( l_{42}; \)
\[ l_{31}^2 + l_{32}^2 + l_{33}^2 = a_{33}, \]
\[ l_{31}l_{41} + l_{32}l_{42} + l_{33}l_{43} = l_{41}l_{31} + l_{42}l_{32} + l_{43}l_{33} = a_{34}, \]  
\[ (5) \]
giving successively \( l_{33} \) and \( l_{43}; \)
\[ l_{41}^2 + l_{42}^2 + l_{43}^2 + l_{44}^2 = a_{44}, \]  
\[ (6) \]
giving \( l_{44}. \)
The basic steps in the Cholesky decomposition are
\[ l_{ii} = \left( a_{ii} - \sum_{j=1}^{i-1} l_{ij}^2 \right)^{1/2}, \quad i = 1, \ldots, n \]  
\[ (7) \]
\[ l_{ij} = \frac{a_{ij} - \sum_{k=1}^{j-1} l_{ik}^j l_{jk}}{l_{jj}}, \quad i = 1, \ldots, n \]
\[ \quad j = 1, \ldots, i-1 \]  
\[ (8) \]
The division in (8) is usually replaced by computing instead the reciprocal of the square root in (7) and then multiplying the numerator in (8) by the corresponding reciprocal. In fact, the computed reciprocals are stored in the diagonal positions of \( L \) and used directly at later steps of computation. From (7), and (8) \((i-1)\) multiplications and \((i-1)\) additions are needed to compute \( l_{ii} \) and \( j \) multiplications and \((j-1)\) additions to compute \( l_{ij}. \) In the case of \( 4 \times 4 \) matrix,
\[ L = \begin{pmatrix}
0 & 0 & 0 & 0 \\
1 & 1 & 0 & 0 \\
1 & 2 & 2 & 0 \\
1 & 2 & 3 & 3
\end{pmatrix} \]
Where each element $\overline{1}_{ij}$ denotes the number of multiplications required to compute its own. From (7) and (8) the number of multiplications required to decompose $A$ into $L$ and $L^T$ is given by:

$$
\sum_{i=1}^{n} \left[ (i-1) + \sum_{j=1}^{i-1} j \right] = \frac{n^3}{6} + \frac{n^2}{2} - \frac{2n}{3} \quad (9)
$$

Similarly the number of additions is given by:

$$
\sum_{i=1}^{n} \left[ (i-1) + \sum_{j=1}^{i-1} (j-1) \right] = \frac{n^3}{6} - \frac{n}{6} \quad (10)
$$

In fact, the upper triangular matrix, $L^T$ by Cholesky decomposition can also be obtained from the upper triangular matrix, $U$ formed by the triangular decomposition. This can be done by dividing each row of $U$ by the square root to the diagonal elements of $U$.

2. Matrix Inversion

For finding the inverse of a symmetric and positive definite matrix $A$ we use the different method from that in the previous section. Here we take advantage of the way the matrix is decomposed.

$$
A = LL^T \\
A^{-1} = (L^T)^{-1}L^{-1} \\
L^T_A^{-1} = L^{-1} \quad (11)
$$

The (11) does not need the explicit inversion of $L$. Consider the case of a $3 \times 3$ matrix. We denote $l_{ij}$ as the elements of $L^T$, $m_{ij}$ as the elements of $L^{-1}$, and $b_{ij}$ as the elements of $A^{-1}$. 
Then (11) can be expressed as

\[
\begin{bmatrix}
1_{11} & 1_{21} & 1_{31} \\
0 & 1_{22} & 1_{32} \\
0 & 0 & 1_{33}
\end{bmatrix}
\begin{bmatrix}
b_{11} & b_{12} & b_{13} \\
b_{12} & b_{22} & b_{23} \\
b_{13} & b_{23} & b_{33}
\end{bmatrix}
= 
\begin{bmatrix}
m_{11} & 0 & 0 \\
m_{21} & m_{22} & 0 \\
m_{31} & m_{32} & m_{33}
\end{bmatrix}
\]

Thus \(b_{33} = m_{33}/1_{33}\)

\[b_{23} = -1_{32}b_{33}/1_{22}\]

\[b_{22} = (m_{22} - 1_{32}b_{23})/1_{22}\]

\[b_{13} = -(1_{21}b_{13} + 1_{31}b_{33})/1_{11}\]

\[b_{12} = -(1_{21}b_{22} + 1_{31}b_{23})/1_{11}\]

\[b_{11} = (m_{11} - 1_{21}b_{12} - 1_{31}b_{13})/1_{11}\] (12)

From (12) it can be seen that only the diagonal elements \(m_{11}, m_{22},\) and \(m_{33}\) of \(L^{-1}\) are needed to compute \(A^{-1}\). But these are just the reciprocals of \(1_{ii}\), the diagonal elements of \(L\).

The elements of \(A^{-1}\) are given by

\[
b_{ii} = \frac{1}{1_{ii}} \left[ \frac{1}{1_{ii}} - \sum_{k=i+1}^{n} l_{ki}b_{ik} \right]
\] (13)

which requires \((n-i+1)\) multiplications and \((n-i)\) additions, and

\[
b_{ij} = \frac{1}{1_{ii}} \sum_{k=i+1}^{n} l_{ki}b_{kj}, \quad i < j,
\] (14)

which requires \((n-i+1)\) multiplications and \((n-i-1)\) additions.

From (13) and (14) the number of multiplications is

\[
\sum_{i=1}^{n} \left( n - i + 1 \right) + \sum_{j=1}^{n} \sum_{i=1}^{j-1} \left( n - i + 1 \right) = \frac{n^3}{3} + \frac{n^2}{2} + \frac{n}{6}
\] (15)
and the number of addition is
\[ \sum_{i=1}^{n} (n-i) + \sum_{j=1}^{n} \sum_{i=1}^{j-1} (n-i-1) = \frac{n^3}{3} - \frac{n^2}{2} + \frac{n}{6} \quad (16) \]

3. Back Substitution

To complete the solution of \( AX = C \), the following steps are required:

( i ) \( y \) is obtained from \( Ly = C \).

( ii ) \( b \) is obtained from \( L^Tb = y \).

As shown in the back substitution of Gaussian elimination, to solve the systems of equations \( Ly = C \) requires \( n(n+1)/2 \) multiplications and \( n(n-1)/2 \) additions. The solution of \( L^Tb = y \) requires the same amount of arithmetic. The amount of arithmetic required in the back substitution phase is

\[ n^2 + n \quad \text{multiplications and} \]
\[ n^2 - n \quad \text{additions} \quad (17) \]

As shown in the table 2, the total number of arithmetic operations to find the inverse and the solution is

\[ \frac{n^3}{2} + 2n^2 + \frac{n}{2} \quad \text{multiplications and} \]
\[ \frac{n^3}{2} + \frac{n}{2} - n \quad \text{additions} \quad (18) \]

| Decompose \( A \) into \( L \& L^T \) | \( n^3 + 3n^2 - 4n \)/6 | \( n^3 - n \)/6 |
| A\(^{-1}\) from \( L^T A^{-1} = L^{-1} \) | \( 2n^3 + 3n^2 + n \)/6 | \( 2n^3 - 3n^2 + n \)/6 |
| Back substitution | \( n^2 + n \) | \( n^2 - n \) |
| Total | \( n^3 + 4n^2 + n \)/2 | \( n^3 + n^2 - 2n \)/2 |

Table 2. Number of computations required for solution vector and inverse matrix using Cholesky's method.
IV. Sweep Operator

The method first developed by Beaton in 1964 is a modification of the Adjust operator that reduces the amount of core needed to compute the \(b\) values, ESS, and \((X'X)^{-1}\). Whereas the Adjust operator performs the mapping,

\[
\begin{bmatrix}
X'X & X'Y & I \\
Y'X & Y'Y & 0
\end{bmatrix} \xrightarrow{\text{Adjust}} \begin{bmatrix}
I & b & (X'X)^{-1} \\
0 & ESS & -b'
\end{bmatrix}, \quad (1)
\]

the Sweep operator performs the in-place mapping,

\[
\begin{bmatrix}
X'X & X'Y \\
Y'X & Y'Y
\end{bmatrix} \xrightarrow{\text{Sweep}} \begin{bmatrix}
(X'X)^{-1} & b \\
-b & ESS
\end{bmatrix}. \quad (2)
\]

It is illustrated by the following example. Suppose we have data

\[
\begin{array}{cccc}
X_0 & X_1 & X_2 & Y \\
1 & 1 & 1 & 1 \\
1 & 2 & 1 & 3 \\
1 & 3 & 1 & 3 \\
1 & 1 & -1 & 2 \\
1 & 2 & -1 & 2 \\
1 & 3 & -1 & 1 \\
\end{array}
\]

Adjust operator

\[
\begin{bmatrix}
X'X & X'Y & I \\
Y'X & Y'Y & 0
\end{bmatrix}
\]

Sweep operator

\[
\begin{bmatrix}
X'X & X'Y \\
Y'X & Y'Y
\end{bmatrix}
\]
These tableaus represent the following models with respect to \( Y \).

<table>
<thead>
<tr>
<th>Step</th>
<th>Model</th>
<th>( b ) values</th>
<th>ESS</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>( Y = 0 )</td>
<td>( . )</td>
<td>28</td>
</tr>
<tr>
<td>1</td>
<td>( Y = b_0 )</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>2</td>
<td>( Y = b_0 + b_1 X_1 )</td>
<td>( 3/2, 1/4 )</td>
<td>15/4</td>
</tr>
<tr>
<td>3</td>
<td>( Y = b_0 + b_1 X_1 + b_2 X_2 )</td>
<td>( 3/2, 1/4, 1/3 )</td>
<td>37/12</td>
</tr>
</tbody>
</table>
By observing tableaus in Adjust operator, note that whenever a variable is in the model, its associated \( X'X \) column is an identity column; whenever a variable is not in the model, its associated column in the original identity matrix remains unchanged. The Sweep operator takes advantages of these features: After adjusting the tableau for a particular variable, the Sweep operator replaces the identity column by the associated column in the original identity matrix. From these tableaus it can be seen that if an \( (n+1) \times (n+1) \) matrix \( A \) is swept on \( k \) th pivotal element, then it is transformed into a matrix \( A^* \) such that

\[
\begin{align*}
    a_{kk}^* &= \frac{1}{a_{kk}}, \\
    a_{ik}^* &= -a_{ik} a_{kk}^* & i \neq k, \\
    a_{kj}^* &= a_{kj} a_{kk}^* & j \neq k, \\
    a_{ij}^* &= a_{ij} - a_{ik} a_{kj}^* & i,j \neq k.
\end{align*}
\]

To compute \( a_{kk}^* \) requires one reciprocal. To compute \( a_{ik}^* \) and \( a_{kj}^* \) requires one multiplication. To compute \( a_{ij}^* \) requires two multiplications and one addition. For example, when \( k = 1 \) we have

\[
A^* = \begin{bmatrix}
1 & a_{12} & \cdots & a_{1n+1} \\
\frac{a_{21}}{a_{11}} & a_{11} & \cdots & a_{1n+1} \\
\vdots & \ddots & \ddots & \ddots \\
\frac{a_{n+1,1}}{a_{11}} & \cdots & \frac{a_{n+1,n+1}}{a_{11}} & a_{11} \\
\end{bmatrix}
\begin{bmatrix}
0 & 1 & 1 & \cdots & 1 \\
1 & 2 & 2 & \cdots & 2 \\
1 & 2 & 2 & \cdots & 2 \\
\vdots & \ddots & \ddots & \ddots & \ddots \\
1 & 2 & 2 & \cdots & 2 \\
\end{bmatrix}
\]

\( n \times n \)
The elements of the right matrix represent the number of multiplications required to compute its own. However, because the matrix $A$ is symmetric, it is really necessary to work with only the upper triangle. The symmetry of the matrix $A$ can be preserved by changing the sign of the pivot. In this case the number of multiplications required to obtain $A^*$ is as follows:

\[
\begin{pmatrix}
0 & 1 & 1 & \ldots & 1 \\
0 & 2 & 2 & \ldots & 2 \\
0 & 0 & 2 & \ldots & 2 \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
0 & 0 & 0 & \ldots & 2 \\
\end{pmatrix}_{n} \\
\sum_{i=1}^{n} 2i = n^2 + 2n
\]

And the number of additions is as follows:

\[
\begin{pmatrix}
0 & 0 & 0 & \ldots & 0 \\
0 & 1 & 1 & \ldots & 1 \\
0 & 0 & 1 & \ldots & 1 \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
0 & 0 & 0 & \ldots & 1 \\
\end{pmatrix}_{n} \\
\sum_{i=1}^{n} i = \frac{n^2}{2} + \frac{n}{2}
\]

Since we need $n$ sweepings on $n$ pivotal elements to invert the matrix $A$, the number of multiplications required is
\[ n(n^2 + 2n) = n^3 + 2n^2 \]

and the number of additions required is

\[ n(\frac{n^2}{2} + \frac{n}{2}) = \frac{n^3}{2} + \frac{n^2}{2} \]
V. Comparison based on the number of arithmetic operations

Table 3 summarizes the number of multiplications and additions required for computing \( b \), ESS, and the inverse of \( X'X \) from the normal equations: \( X'Xb = X'y \). In the case of both Gaussian elimination and Cholesky method, an extra \( 2n \) multiplications and one addition are required to compute ESS.

Table 3. Number of computations required for \( b \), ESS, and \((X'X)^{-1}\) using three algorithms

<table>
<thead>
<tr>
<th></th>
<th>Multiplications</th>
<th>Additions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gaussian elimination</td>
<td>( n^3 + n^2 + n )</td>
<td>( n^3 - n + 1 )</td>
</tr>
<tr>
<td>Cholesky method</td>
<td>((n^3+4n^2+5n)/2)</td>
<td>((n^3+ n^2-2n+2)/2)</td>
</tr>
<tr>
<td>Sweep operator</td>
<td>( n^3+2n^2 )</td>
<td>((n^3+ n^2)/2)</td>
</tr>
</tbody>
</table>

Gaussian elimination and the Sweep operator require about twice the number of multiplications as the Cholesky method. And Gaussian elimination requires twice the number of additions as the Cholesky method and the Sweep operator. By IBM SYSTEM/370 MODEL 158, 5 microseconds is taken to perform one addition, and 19 microseconds to perform one multiplication. Using this information we can find the execution time to obtain the \( b \) values, ESS, and \((X'X)^{-1}\) for three methods as follows:

- Gaussian : \((24n^3+19n^2+14n+5)\) microsec.
- Cholesky : \((12n^3+40.5n^2+42.5n+5)\) microsec.
- Sweep : \((21.5n^3+40.5n^2)\) microsec.

Table 4 shows the execution time of algorithms for different \( n \) values.

Table 4. Comparison of execution time for three algorithms (in seconds)

<table>
<thead>
<tr>
<th>( n )</th>
<th>5</th>
<th>10</th>
<th>20</th>
<th>30</th>
<th>50</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gaussian elimination</td>
<td>.004</td>
<td>.026</td>
<td>.200</td>
<td>.666</td>
<td>3.048</td>
</tr>
<tr>
<td>Cholesky method</td>
<td>.003</td>
<td>.017</td>
<td>.113</td>
<td>.362</td>
<td>1.603</td>
</tr>
<tr>
<td>Sweep operator</td>
<td>.004</td>
<td>.026</td>
<td>.188</td>
<td>.617</td>
<td>2.789</td>
</tr>
</tbody>
</table>

It can be seen that Cholesky method is the fastest and the Sweep operator is the next.
VI. Results and Discussion

We generated several \(X'X\) matrices, computed the inverse using all three methods and then compared their accuracies. The generated \(X'X\) matrices ranged from ill-conditioned to well-conditioned.

1. Condition number of \(X'X\)

A system of normal equations \(X'Xb=X'Y\) is said to be ill-conditioned if small variations in the elements of \(X'X\) and \(X'Y\) have a large effect on the exact solution, \(b\). For example, the difference, \(\delta b\), between the solution of \(X'Xb=X'Y\) and that of \((X'X+\delta X'X)(b+\delta b)=X'Y+\delta X'Y\) can be expressed as

\[
\delta b = (X'X + \delta X'X)^{-1}(\delta X'Y - \delta X'Xb),
\]

and its value depends critically on the inverse matrix. If \(X'X\) is nearly singular, that is, small changes in its elements can cause singularity, then \(\delta b\) could be very large. For the comparison of the accuracy of the computed solution vector, \(b\), we concentrated on the accuracy of the computed inverse of \(X'X\) which would mainly affect the accuracy of the computed \(b\). To obtain many \(X'X\) matrices, from well-conditioned to ill-conditioned, we started with an arbitrary correlation matrix, and produced correlation matrices with condition numbers 10, 10^3, 10^5, and 10^8. For this we used the subprogram CECORR(Milliken, 1981) the input of which is any correlation matrix and a desired condition number, and the output of which is a new correlation matrix with the desired condition number. With each new correlation matrix, \(R\), we generated data and constructed 20 uncorrected \(X'X\) matrices of dimension, \((n+1)\times(n+1)\). The sizes of the correlation matrices were 10 and 20 resulting in 11x11 and 21x21 uncorrected \(X'X\) matrices which were generated using
40 observations for each variable. Table 5 shows the range of the condition numbers of the $X'X$ matrices generated from correlation matrices with the condition numbers $10, 10^3, 10^5,$ and $10^8$.

<table>
<thead>
<tr>
<th>Cond. # of correlation matrix</th>
<th>Cond. # of $X'X$ matrix</th>
</tr>
</thead>
<tbody>
<tr>
<td>A $10$</td>
<td>$10 - 10^2$</td>
</tr>
<tr>
<td>B $10^3$</td>
<td>$10^3 - 10^4$</td>
</tr>
<tr>
<td>C $10^5$</td>
<td>$10^5 - 10^6$</td>
</tr>
<tr>
<td>D $10^8$</td>
<td>$10^8 - 10^9$</td>
</tr>
</tbody>
</table>

In practice it is common for one variable to be highly correlated with a linear combination of other variables so that the columns of $X$ may be close to being linearly dependent. This means that $X'X$ may be near singularity, the smallest eigenvalue will be small, and the condition number will be large. Examples of ill-conditioned matrices include Longley's (1967) test data with condition number of $4.8 \times 10^9$ and Wampler's (1970) polynomial models which had a value of the order of $10^6$.

In this report the $X'X$ matrices in D were considered ill-conditioned. Those in A were considered well-conditioned while those in B and C were in between.

2. Generating $X'X$ matrix

First we generated data with distributions for each variable $X_i \sim N(\mu_i, \sigma_i^2)$ and specified correlations between the variables. In fact, it was assumed that each variable followed the Standard Normal distribution such that $\mu_i=0$, and $\sigma_i^2=1$ for all $i$. Correlations between variables were obtained from the correlation matrix, $R$, which depended on the condition number and the matrix size desired for $X'X$ (See Appendix B).

To generate data for $n$ variables, $X_1$ through $X_n$, we repeated the following formula for $j=2$ to $n$. 

\[ x_1 = z_1 \cdot \sigma_1 + \mu_1 = z_1 \]
\[ x_j = z_j \cdot \sigma_j^* + \mu_j^* \]

\( z_j \) was obtained from RNOR(0) of Marsaglia's (1976) Super-Duper random number generator as a Standard Normal deviate.

\( \sigma_j^{*2} \) is the conditional variance of \( x_j \) given \( x_1=x_1, \ldots, x_{j-1}=x_{j-1} \).

\[
\sigma_j^{*2} = \sigma_j^2 - \left( \begin{array}{c} \sigma_{j,1} \\
\vdots \\
\sigma_{j,j-1}
\end{array} \right) \sum_{j=1}^{j-1} \left( \begin{array}{c} \sigma_{1,j} \\
\vdots \\
\sigma_{j-1,j}
\end{array} \right) \sigma_j^2 = 1
\]

where \( \sigma_{jk} \) is the covariance between \( x_j \) and \( x_k \), and \( \sum_{j=1} \) is the covariance matrix for \( x_1, \ldots, x_{j-1} \) which can be obtained from the correlation matrix \( R \) (\( R_{j-1} \) is the submatrix of \( R \), with first \( j-1 \) rows and columns) by

\[
\sum_{j-1} = D' R_{j-1} D = R_{j-1} \text{ since the diagonal matrix, } D, \text{ with } i \text{ th diagonal element equal to } \sigma_i \text{ is an identity matrix.}
\]

\( \mu_j^* \) is the conditional mean of \( x_j \) given \( x_1=x_1, \ldots, x_{j-1}=x_{j-1} \).

\[
\mu_j^* = \mu_j + \left( \begin{array}{c} \sigma_{j,1} \\
\vdots \\
\sigma_{j,j-1}
\end{array} \right) \sum_{j=1}^{j-1} \left( \begin{array}{c} x_1 - \mu_1 \\
\vdots \\
x_{j-1} - \mu_{j-1}
\end{array} \right), \text{ all } \mu_i = 0.
\]

Repeating this procedure for the number of observations desired, we obtained a data matrix, \( X \), and an \( X'X \) matrix of dimension \((n+1)x(n+1)\) for the first order linear model, \( Y = \beta_0 + \beta_1 X_1 + \cdots + \beta_n X_n \).

3. Accuracy

To compare the accuracies of the inverse computed using the three algorithms, we computed the amount of error in the inverse as the sum of the absolute differences between the product of the original \( X'X \) and its' computed \( X'X \) inverse and the identity matrix (It is denoted by

\[
\sum_{i=1}^{n+1} \sum_{j=1}^{n+1} \left| (X'X)(X'X)^{-1} - I \right| .
\]

Tables 6 and 7 show the average
amount of error in the computed inverses for different condition numbers. Matrices of size 11 x 11 are in Table 6 and size 21 x 21 are in Table 7. In all cases 20 matrices of each size were generated as described above.

Table 6. The average error amount of the computed inverse for three algorithms (11 x 11 X'X matrices)

<table>
<thead>
<tr>
<th>Cond. # of X'X</th>
<th>10-10²</th>
<th>10³-10⁴</th>
<th>10⁵-10⁶</th>
<th>10⁸-10⁹</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gaussian elimination</td>
<td>0.332D-13</td>
<td>0.048D-11</td>
<td>0.039D-09</td>
<td>0.040D-06</td>
</tr>
<tr>
<td>Cholesky method</td>
<td>0.328D-13</td>
<td>0.055D-11</td>
<td>0.041D-09</td>
<td>0.046D-06</td>
</tr>
<tr>
<td>Sweep operator</td>
<td>0.353D-13</td>
<td>0.130D-11</td>
<td>0.163D-09</td>
<td>0.164D-06</td>
</tr>
</tbody>
</table>

Table 7. The average error amount of the computed inverses for three algorithms (21 x 21 X'X matrices)

<table>
<thead>
<tr>
<th>Cond. # of X'X</th>
<th>10-10²</th>
<th>10³-10⁴</th>
<th>10⁵-10⁶</th>
<th>10⁸-10⁹</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gaussian elimination</td>
<td>0.203D-12</td>
<td>0.851D-11</td>
<td>0.807D-09</td>
<td>0.923D-06</td>
</tr>
<tr>
<td>Cholesky method</td>
<td>0.165D-12</td>
<td>0.559D-11</td>
<td>0.650D-09</td>
<td>0.508D-06</td>
</tr>
<tr>
<td>Sweep operator</td>
<td>0.223D-12</td>
<td>0.810D-11</td>
<td>0.869D-09</td>
<td>0.693D-06</td>
</tr>
</tbody>
</table>

It can be seen that Gaussian elimination and Cholesky method are more accurate than the Sweep operator for size 11 x 11 matrices, but Cholesky method is the most accurate for size 21 x 21 matrices.

4. Execution Time

Table 8 and 9 show the average execution time to find the b-values, ESS, and \((X'X)^{-1}\) for different condition numbers of \(X'X\) matrices.
Table 8. The average execution time to find the b values, ESS, and \((X'X)^{-1}\) for three algorithms (11x11 \(X'X\) matrices)

<table>
<thead>
<tr>
<th>Cond. # of (X'X)</th>
<th>10^{-2}</th>
<th>10^{3}-10^{4}</th>
<th>10^{5}-10^{6}</th>
<th>10^{8}-10^{9}</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gaussian elimination</td>
<td>.053</td>
<td>.061</td>
<td>.054</td>
<td>.060</td>
</tr>
<tr>
<td>Cholesky method</td>
<td>.021</td>
<td>.025</td>
<td>.024</td>
<td>.037</td>
</tr>
<tr>
<td>Sweep operator</td>
<td>.036</td>
<td>.044</td>
<td>.038</td>
<td>.040</td>
</tr>
</tbody>
</table>

Table 9. The average execution time to find the b values, ESS, and \((X'X)^{-1}\) for three algorithms (21x21 \(X'X\) matrices)

<table>
<thead>
<tr>
<th>Cond. # of (X'X)</th>
<th>10^{-2}</th>
<th>10^{3}-10^{4}</th>
<th>10^{5}-10^{6}</th>
<th>10^{8}-10^{9}</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gaussian elimination</td>
<td>.343</td>
<td>.340</td>
<td>.337</td>
<td>.331</td>
</tr>
<tr>
<td>Cholesky method</td>
<td>.133</td>
<td>.109</td>
<td>.111</td>
<td>.113</td>
</tr>
<tr>
<td>Sweep operator</td>
<td>.218</td>
<td>.221</td>
<td>.212</td>
<td>.234</td>
</tr>
</tbody>
</table>

As shown by the number of arithmetic operations Cholesky's method is the fastest and the Sweep operator is the next.

In conclusion Cholesky's method appeared to be the most accurate and fastest of the three algorithms and, therefore, the one we would recommend. The Sweep operator's accuracy seemed to be slightly less than that of Gaussian elimination but its execution time was consistently less.
References

Conte, S.D. and de Boor, Carl(1972), Elementary Numerical Analysis,

New York : John Wiley & Sons


IBM SYSTEM/370 MODEL 158 Functional Characteristics

Issaacson, E., and Keller, H.B.(1966), Analysis of Numerical Methods,
New York : John Wiley & Sons

Kronsjo, L.I.(1979), Algorithms—Their Complexity and Efficiency,
New York : John Wiley & Sons


Milliken, G.A.(1981), "Computer program for generating correlation matrix" "


APPENDICES
APPENDIX A

COMPUTER PROGRAM FOR GENERATING X'X MATRIX
AND COMPUTING ITS INVERSE USING THREE ALGORITHMS
THIS PROGRAM PERFORMS THE FOLLOWING PROCEDURES.

1. GENERATE DATA GIVEN DISTRIBUTIONS AND CORRELATION COEFFICIENTS BETWEEN VARIABLES.
2. CONSTRUCT UNCORRECTED $X'X$ MATRIX
3. COMPUTE THE CONDITION NUMBER OF $X'X$
   (WE GENERATE MANY $X'X$ MATRICES WITH DIFFERENT CONDITION NUMBERS)
4. COMPUTE THREE INVERSES OF $X'X$, SOLUTION VECTORS, AND ERROR SUM OF SQUARES USING THREE ALGORITHMS.
5. COMPARE THEIR ACCURACIES, AND EXECUTION TIME
IMPLICIT REAL*8 (A-H,O-Z)
REAL*8 TITLE(10), FMT(10)
REAL*8 MEAN(20), VAR(20), R(20,20), C(20,20)
REAL*8 COV(20,20), LTCOV(260), ICOV(20,20), PCIC(21)
REAL*8 DX(100,21), XPX(21,21), RSUM(21), INORM
REAL*8 ERR(3), DSUM(3), SSQS(3), EXTME(3), TSUM(3)
REAL*8 UPTR(400), EV(400), STOR(400), Y(100), XPY(21)
REAL*8 R(21,1), A(460), X(100,21), IXPX(21,21), TSQS(3)
REAL*8 DMIN(22), COV1(20,20), LTXPX(460)
INTEGER V(22)
READ(5,5) TITLE
5 FORMAT(10A8)
WRITE(6,6) TITLE
6 FORMAT(1X,10A8)
READ(5,7) IX, IY
7 FORMAT(1X,110,1X,1I0)
WRITE(6,7) IX, IY
10 READ(5,15, END=990) NOV, NOBS, NMT, CCND, IPRINT
15 FORMAT(12,13,13,1X,E16.5,14X,1I1)
WRITE(6,16) NOV, NOBS, NMT, CCND, IPRINT
16 FORMAT(1X,12,13,13,1X,E16.5,14X,1I1)
N=1
DJ=1.0
DJJ=2.0
NOV1 = NOV + 1
NOV2 = NOV1+1
IF(NOVI.GT.21. OR. NOBS.GT.100) GO TO 900

C
*** NOV ; TOTAL NO. OF VARIABLES
*** NOBS ; TOTAL NO. OF OBSERVATIONS
*** MEAN(I) ; MEAN OF ITH VAR.
*** VAR(I) ; VARIANCE OF ITH VAR.
*** N ; NO. OF X'X MATRIX GENERATED
*** NOV1 ; DIMENSION OF X'X MATRIX
*** FMT ; VARIABLE FORMAT FOR READING CORR. MATRIX

C
READ(5,20) (MEAN(I), I=1, NOV)
READ(5,20) (VAR(I), I=1, NOV)
WRITE(6,21) (MEAN(I), I=1, NOV)
WRITE(6,21) (VAR(I), I=1, NOV)
20 FORMAT(20F4.1)
21 FORMAT(1X,20F6.1)
READ(5,5) FMT
WRITE(6,6) FMT
DO 25 I=1, NOV
READ(5,FMT) (R(I,J), J=1, NOV)
WRITE(6,22) (R(I,J), J=1, NOV)
22 FORMAT(1X,11F12.4/1X,11F12.4)
25 CONTINUE

C
*** CALL THE SUBROUTINE GECORR FOR GENERATING CORR.
*** MATRIX GIVEN A DESIRED CONDITION NUMBER

C
CALL GECORR(R, NOV, CCND, IPRINT)

C
DO 30 I=1, NOV
DO 30 J=1, NOV
D(I,J) = 0.
30 CONTINUE
DO 32 I=1,NOV
32 D(I,I) = DSQRT(VAR(I))
DO 35 I=1,NOV
DC 35 J=1,NOV
SUM=0.
DO 36 K=1,NOV
36 SUM = SUM + D(I,K)*R(K,J)
COV(I,J) = SUM
35 CONTINUE
DO 40 I=1,NOV
DO 40 J=1,NOV
SUM=0.
DO 41 K=1,NOV
41 SUM = SUM + COV(I,K)*D(K,J)
40 COV(I,J) = SUM
DO 50 I=1,NOV
II = I*(I-1)/2
DO 50 J=1,I
50 LTOC(I+J) = COV(I,J)
C
C *** COV(I,J) ; COVARIANCE MATRIX
C *** LTOC(K) ; LOWER TRIANGULAR OF COVARIANCE MATRIX
C
CALL RSTART(Ix, IY)
C
C *** RSTART GIVES SEED FOR GENERATING RANDOM NUMBER
C *** RNOR(0) ; SUPER-DUPER RANDOM NUMBER GENERATOR AS
C A STANDARD NORMAL DEVIATE

ITNOV=(NOV*NOV+NOV)/2
53 IF(N.GT.NCRT) GO TO 800
WRITE(6,55) TITLE,N
55 FORMAT(10A8,'MATRIX # ',13,slash)
   IF(IPRINT.EQ.0) GO TO 77
   IF(IPRINT.EQ.1) GO TO 77
WRITE(6,60)
60 FORMAT(1X,'COVARIANCE MATRIX',slash)
DO 70 I=1,NOV
70 WRITE(6,71) (COV(I,J),J=1,NOV)
71 FORMAT(1X,1F12.5)
77 SQRTVR = DSQRT(VAR(1))
DO 290 IOBS=1,NOBS
STDN = RNOR(0)
X(IOBS,1) = STDN*SQRTVR + MEAN(1)
IVAR=1
IF(IPRINT.EQ.0) GO TO 82
IF(IPRINT.EQ.1) GO TO 82
WRITE(6,80) IVAR,STDN,X(IOBS,1)
80 FORMAT(5X,'X',9I2,5X,'RAN. #',F8.4,5X,'DATA',F10.4)
82 DC = COV(2,1)/COV(1,1)
CONM = MEAN(2) + DC*(X(IOBS,1)-MEAN(1))
CCNV = VAR(2) - DC*COV(1,2)
STDN = RNOR(0)
IVAR=2
X(IOBS,2) = STDN*DSQRT(CCNV) + CONM
IF(IPRINT.EQ.0) GO TO 85
IF(IPRINT.EQ.1) GO TO 85
WRITE(6,80) IVAR,STDN,X(IOBS,2)
C
C *** X(I,J) ; ITH OBSERVATION OF JTH VARIABLE
*** CCNM ; CONDITIONAL MEAN OF X2 GIVEN X1=C1
*** CCNV ; CONDITIONAL VARIANCE OF X2 GIVEN X1=C1

85 DO 200 IVAR=3,NOV
    IVARM1 = IVAR - 1
    ITVAR = IVARM1*(IVARM1+1)/2
    DO 90 I=1,ITVAR
    90 A(I) = LTCOV(I)

*** A ; LOWER TRIANGULAR OF COVARIANCE MATRIX
*** SUBROUTINE DMFSD AND DSINV IS TO CALCULATE THE
*** INVERSE OF COVARIANCE MATRIX

CALL DMFSD(A,IVARM1)
CALL DSINV(A,IVARM1)
DO 100 I=1,IVARM1
DO 100 J=1,I
    K = I*(I-1)/2 + J
    ICOV(I,J) = A(K)
    ICOV(J,I) = ICOV(I,J)
100 CONTINUE

*** ICOV ; FULL MATRIX OF COVARIANCE MATRIX INVERSE

IF(IOBS.GT.1) GO TO 115
IF(IPRINT.EQ.0) GO TO 115
IF(IPRINT.EQ.1) GO TO 115
WRITE(6,105)
105 FORMAT(//T10,'INVERSE OF COVARIANCE MATRIX UNDER',
         ' CONSIDERATION',//)
    DO 110 I=1,IVARM1
    110 WRITE(6,71) (ICOV(I,J),J=1,IVARM1)

115 DO 130 J=1,IVARM1
    PCIC(J) = 0.
    DO 120 I=1,IVARM1
    PCIC(J) = PCIC(J) + COV(IVAR,I)*ICOV(I,J)
120 CONTINUE
130 CONTINUE
    ADDM = 0.
    ADDV = 0.
    DO 140 J=1,IVARM1
    ADDM = ADDM + PCIC(J)*(X(IOBS,J)-MEAN(J))
    ADDV = ADDV + PCIC(J)*COV(J,IVAR)
140 CONTINUE
    CONM = MEAN(IVAR) + ADDM
    CONV = VAR(IVAR) - ADDV

*** CONM ; CONDITIONAL MEAN OF VARIABLE IVAR
*** CONV ; CONDITIONAL VARIANCE OF VARIABLE IVAR

STDN = RNOR(0)
X(IOBS,IVAR) = STDN*DSQRT(CONV) + CONM
IF(IPRINT.EQ.0) GO TO 200
IF(IPRINT.EQ.1) GO TO 200
WRITE(6,80) IVAR,STDN,X(IOBS,IVAR)
200 CONTINUE

IF(IPRINT.EQ.0) GO TO 290
IF(IPRINT.EQ.1) GO TO 290
WRITE(6,210) IOBS,(X(IOBS,J),J=1,NOV)
210 FORMAT(//1X,'#',12,10F12.4)
290 CONTINUE
   DO 295 I=1,NOBS
   DX(I,1) = 1.
   DO 295 J=1,NOV
   JJ=J+1
   DX(I, JJ) = X(I, J)
295 CONTINUE
   IF (IPRINT.EQ.0) GO TO 345
   WRITE(6,300)
300 FORMAT(/'DATA MATRIX/',/)
   DO 320 I=1,NOBS
   WRITE(6,310) (DX(I,J), J=1,NOV1)
310 FORMAT(1X,1LE12.4/1X,1LE12.4)
   CONTINUE
   WRITE(6,340)
340 FORMAT(/'X'X MATRIX/',/)
   DO 370 I=1,NOV1
   DO 360 J=1,NOV1
   XPX(I, J) = 0.
   DO 350 K=1,NOBS
   XPX(I, J) = XPX(I, J) + DX(K, I) * DX(K, J)
350 CONTINUE
   IF (IPRINT.EQ.0) GO TO 370
   WRITE(6,310) (XPX(I, JJ), JJ=1,NOV1)
370 CONTINUE
   DO 375 I=1,NOBS
375 Y(I) = RNOR(0)
   DO 390 I=1,NOV1
   SUM = 0.
   DO 380 J=1,NOBS
   SUM = SUM + DX(I, J) * Y(J)
380 SUM = SUM + DX(I, I) * Y(I)
   YPY = SUM
C
C *** SUBROUTINE TARRAY IS CALLED TO TRANSFORM X'X
C     INTO UPPER TRIANGULAR MATRIX
C *** SUBROUTINE EIGEN IS CALLED TO FIND THE EIGEN
C     VALUES OF X'X
C
   CALL TARRAY(NOV1, 21, UPTR, XPX)
   CALL EIGEN(UPTR, STOR, NOV1, 1)
   NC = 1
   DO 400 I=1,NOV1
   NC = NC + (I**2 - I) / 2
   EV(I) = UPTR(NC)
400 CONTINUE
   WRITE(6, 402)
402 FORMAT(/,'EIGEN VALUE FOR X'X MATRIX')
   WRITE(6, 310) (EV(I), I=1,NCV1)

C
C *** SUBROUTINE SORT FINDS THE MAX AND MIN OF EIGEN
C     VALUES FOR CONDITION NUMBER
C
   CALL SORT(EV, NOV1)
   CONDNO = EV(1) / EV(2)
   WRITE(6, 410) CONDNO
410 FORMAT(/,'CONDITION NUMBER OF X'X = ',E13.5)
DO 420 I=1,NOV1
  II = I*(I-1)/2
  DO 420 J=1,I
    420 LTXPX(II+J) = XPX(I,J)
C
C *** LTXPX ; LOW TRIANGULAR OF XPX MATRIX
C
IF(IPRINT.EQ.0) GO TO 425
WRITE(6,422)
  422 FORMAT(1X,'INVERSE BY CHOLESKY METHOD')//
C
C *** SUBROUTINE DMFSD FINDS CHOLESKY DECOMPO. OF X'X
C *** SUBROUTINE DSNV CALCULATES THE INVERSE OF X'X
C
425 CONTINUE
C
C *** LET LL'=A, THEN AX=B IS LL'X=B.
C FOR THE SOLUTION X, WE SOLVE LY=B AND L'X=Y.
C *** SUBROUTINE SOLTON IS TO SOLVE LY=B AND L'X=Y.
C
CALL INTIME(ITIME1)
CALL DMFSD(LTXPX,NOV1)
CALL SOLTON(NOV1,LTXPX,B,XPY,YPY,ESS)
CALL DSNV(LTXPX,NOV1)
CALL INTIME(ITIME2)
EXTIME(1) = (ITIME2 - ITIME1)/100.
DO 430 I=1,NOV1
  II = I*(I-1)/2
  DO 430 J=1,I
    IXPX(I,J) = LTXPX(II+J)
    IXPX(J,I) = IXPX(I,J)
430 CONTINUE
IF(IPRINT.EQ.0) GO TO 440
DO 435 I=1,NOV1
  WRITE(6,310) (IXPX(I,J),J=1,NOV1)
435 CONTINUE
440 DERR=0.
CALL ERRINV(XPX,IXPX,NOV1,DERR,IPRINT)
C
C *** SUBROUTINE ERRINV IS TO CALCULATE THE AMOUNT OF
C THE ERROR OF THE COMPUTED INVERSE OF X'X
C
ERR(I) = DERR
WRITE(6,480) N,ERR(I)
  480 FORMAT(1X,'MATRIX #',I3,T15,
        1 'CHOLESKY ERROR = ',E12.5)
WRITE(6,490) EXTIME(1)
  490 FORMAT(T15,'EX. TIME = ',F12.7)
WRITE(6,494) (B(I),I=1,NOV1)
  494 FORMAT(T15,'SOLUTION : ',8E13.5)
C
C *** COMPUTE THE INVERSE OF X'X USING SWEEP OPERATOR
C
DMIN(I)=0.
DO 500 I=1,NOV2
  V(I)=1
500 CONTINUE
DO 505 I=1,NOV1
  II = I*(I-1)/2
  DO 505 J=1,I
505  LTXPX(II+J) = XPX(I,J)
      IF(IPRINT.EQ.0) GO TO 507
      WRITE(6,506)

506  FORMAT('1',/1X,'INVERSE BY SWEEP OPERATOR' //)

C  *** SUBROUTINE LTG2SP IS CALLED FOR X'X INVERSE
C
507  NTOT = (NOV1*NOV1+NGV1)/2
      DO 508 I=1,NOV1
      LTXPX(NTOT+I) = XPY(I)
      LTXPX(NTOT+NOV2) = YPY
      CALL INTIME(ITIME1)
      CALL LTG2SP(I,NOV2,LTXPX,DMIN,V)
      DO 510 I=1,NOV2
      DMIN(I) = 1.0E-12*LTXPX(I*(I+1)/2)
      DO 520 K=2,NOV1
      CALL LTG2SP(K,NOV2,LTXPX,DMIN,V)

520  CONTINUE
      CALL INTIME(ITIME2)
      EXTIME(2) = (ITIME2 - ITIME1)/100.
      DO 530 I=1,NOV1
      II = I*(I-1)/2
      DO 530 J=1,I
      IXPX(I,J) = LTXPX(II+J)
      IXPX(J,I) = IXPX(I,J)

530  CONTINUE
      IF(IPRINT.EQ.0) GO TO 540
      DO 535 I=1,NOV1
      WRITE(6,310) (IXPX(I,J),J=1,NOV1)

535  CONTINUE

540  DERR=0.
      CALL ERRINV(XPX,IXPX,NOV1,DERR,IPRINT)
      ERR(2) = DERR
      WRITE(6,580) N,ERR(2)

580  FORMAT('1',/1X,'MATRIX #',I3,T15.,
         1   'Sweep Error = ',E12.5)
      WRITE(6,490) EXTIME(2)
      DO 590 I=1,NOV1
      LTXPX(NTOT+I) = -LTXPX(NTOT+I)
      WRITE(6,494) (LTXPX(NTOT+I),I=1,NGV1)
      DO 600 I=1,NOV1
      DO 600 J=1,NOV1
      IXPX(I,J) = XPX(I,J)

600  CONTINUE

C  *** COMPUTE X'X INVERSE USING GAUSSIAN ELIMINATION
C
      CALL INTIME(ITIME1)
      CALL GAUEL(NOVI,IXPX,B,XPY,YPY,ESS,IPRINT)
      CALL INTIME(ITIME2)
      EXTIME(3) = (ITIME2 - ITIME1)/100.
      IF(IPRINT.EQ.0) GO TO 640
      WRITE(6,610)

610  FORMAT('1',/1X,'INVERSE BY GAUSSIAN ELIMINATION' //)
      DO 635 I=1,NOV1
      WRITE(6,310) (IXPX(I,J),J=1,NOV1)

635  CONTINUE

640  DERR=0.
      CALL ERRINV(XPX,IXPX,NOV1,DERR,IPRINT)
      ERR(3) = DERR
WRITE(6,680) N,ERR(3)
680 FORMAT(//1X,'MATRIX #',I3,T15,
1 'GAUSSIAN ERROR = ',E12.5)
WRITE(6,490) EXTIME(I)
WRITE(6,494) (B(I),I=1,NCV1)
IF(N.GE.2) GO TO 710
DO 700 I=1,3
SSQS(I) = 0.0
TSQS(I) = 0.0
TSM(I) = EXTIME(I)
700 DSUM(I) = ERR(I)
N=N+1
GO TO 53
710 N=N+1
DJ=DJ+1.0
DO 720 I=1,3
DSUM(I) = DSUM(I) + ERR(I)
DXI = DJ*ERR(I)-DSUM(I)
SSQS(I) = SSQS(I) + DXI*DXI/DJJ
TSM(I) = TSM(I) + EXTIME(I)
DXT = DJ*EXTIME(I)-TSM(I)
TSQS(I) = TSQS(I) + DXT*DXT/DJJ
720 CONTINUE
DJJ = DJJ+DJ+DJ
GO TO 53
800 DJM1 = DJ-1
SQRTD=DSQRT(DJ)
WRITE(6,805)
805 FORMAT(111',/T38,'ERROR',T78,'EX. TIME',//)
WRITE(6,810)
810 FORMAT(T31,'AVERAGE',T43,'STD. ERROR',T71,
1 'AVERAGE',T83,'STD. ERROR')
DO 840 I=1,3
AVG = DSUM(I)/DJ
VARI= SSQS(I)/DJM1
STDEV =DSQRT(VARI)
STDEMR = STDEDEV/SQRTD
TAVG = TSM(I)/DJ
TVAR = TSQS(I)/DJM1
TDEV =DSQRT(TVAR)
TERR = TDEV/SQRTD
WRITE(6,820) I,AVG,STDEMR,TAVG,TERR
820 FORMAT(/T10,'METHOD ',I1,T25,E13.5,T40,E13.5,T65,
1 E13.5,T80,E13.5)
840 CONTINUE
CALL RSTOP(JX,JY)
WRITE(6,890) JX,JY
890 FORMAT(111',/1X,'FOR A NEW SEED ',2I12)
900 STOP
END
SUBROUTINE GECORR(A,NR,C,IPRINT)
IMPLICIT REAL*8(A-H,C-Z)

C GENERATE A P.D. CORRELATION MATRIX
DIMENSION A(20,20), W(20,20), AS(20,20), BS(400),
1     EV(20), FMT(10)

C NR---- NUMBER OF ROWS (AND COLS) OF PROPOSED MATRIX
C C---- DESIRED CONDITION NUMBER
C MNR = 20
C A---- THE PROPOSED CORRELATION MATRIX
C W---- THE WEIGHT MATRIX
DO 4 I=1,NR
   DO 4 J=1,NR
      W(I,J) = 1.0
      IF(I.EQ.J) W(I,J)=0.0
4 CONTINUE
WRITE (6,6) NR,C
6 FORMAT (1H1,10X,22HTHE NUMBER OF ROWS IS ,I3,3X,
130HTHE IDEAL CONDITION NUMBER IS ,E16.5)
   IF(IPRINT.EQ.0) GO TO 50
   WRITE (6,7)
7 FORMAT (10X,28H THE PROPOSED MATRIX IS—A—)
   DO 8 I=1,NR
      WRITE (6,11) (A(I,J),J=1,NR)
8 FORMAT (6,11) (A(I,J),J=1,NR)
11 FORMAT (/1X,10F13.4/1X,10F13.4)
C CALL THE SUBROUTINE FOR COMPUTING THE P.D.
C CORRELATION MATRIX
C 50 CALL PDCORR(A,W,AS,NR,C,AK,CODE,CT,BS,MNR,EV,
1     IPRINT)
C OUTPUT RESULTS
   IF(IPRINT.EQ.0) GO TO 60
   WRITE (6,12)
12 FORMAT (10X,36H THE NEW CORRELATION MATRIX IS—AS—)
   DO 13 I=1,NR
      WRITE (6,11) (AS(I,J),J=1,NR)
13 FORMAT (6,11) (AS(I,J),J=1,NR)
60 WRITE (6,16) CT
16 FORMAT (10X,31H THE CONDITION NUMBER OF AS IS ,E16.5)
   WRITE (6,17) (EV(I),I=1,NR)
17 FORMAT (10X,28H THE CHARACTERISTIC ROOTS ARE/,
1     10X,10E12.5)
   DO 20 I=1,NR
      DO 20 J=1,NR
      A(I,J) = AS(I,J)
20 RETURN
END
SUBROUTINE ARRAY (MODE, I, J, N, M, S, D)
IMPLICIT REAL*8(A-H,O-Z)
DIMENSION S(I), D(I)
NI = N-I
IF (MODE-1) 1,1,3
1 IJ = I*J+1
NM = N*J+1
DO 2 K=1,J
   NM = NM-NI
   DO 2 L=1,I
      IJ = IJ+1
      NM = NM+1
2 D(NM) = S(IJ)
RETURN
3 IJ = 0
   NM = 0
   DO 5 K=1,J
      DO 4 L=1,I
         IJ = IJ+1
      4 NM = NM+1
3 S(IJ) = D(NM)
5 NM = NM+NI
RETURN
END

SUBROUTINE ASSTAR(A,W,AS,NR,CT,AK,STOR,EV,EPL,BS,IPRINT)
C COMPUTE THE NEW MATRIX TO BE CONSIDERED FOR OPTIMUM
C CORR MATRIX
IMPLICIT REAL*8(A-H,O-Z)
DIMENSION A(20,20), W(20,20), AS(20,20), EV(20),
   1 STOR(400), BS(400)
NRM1 = NR-1
WRITE (6,1) AK
1 FORMAT (10X,4HAK= ,E15.8)
DO 2 I=1,NRM1
   K = I+1
2 J = K, NR
   AS(I,J) = A(I,J)*(1.0-1.0/(AK*W(I,J)))
DO 3 I=1, NR
   AS(I,J) = AS(J,I)
3 AS(I,J) = 1.0
   CALL TARRAY (NR,20,BS,AS)
   CALL EIGEN (BS,STOR,NR,1)
   N = 1
   DO 4 I=1,NR
      N = I+(I*I-I)/2
   4 EV(I) = BS(N)
   CONTINUE
C COMPUTE THE CHARACTERISTIC ROOTS OF AS
C SORT CHAR ROOTS TO GET MAX AND MIN
   CALL SORT (EV,NR)
C COMPUTE CONDITION NUMBER
   CT = 2.0E20
   IF (DABS(EV(2)).GT.0.000000001) CT = EV(1)/EV(2)
   IF (IPRINT.EQ.0) GO TO 20
   WRITE (6,5) CT,EV(1),EV(2)
5 FORMAT (10X,3E16.8)
20 RETURN
END
SUBROUTINE CHECK (CT,C,CHECKY,CS,SK,AK1)
IMPLICIT REAL*8(A-H,O-Z)
C DETERMINE IF THE CONDITION NUMBER IS SMALL ENOUGH----
C IF SO CHECKY=1
CHECKY = 0.0
C20P = C*0.2
DEL = (DABS(CT-C))*10000.0/C
IF (DEL .LE. C20P) CHECKY=1.0
IF (CHECKY .EQ. 1.0) RETURN
IF (CT.GT.C) CALL DECREA (SK,CS,AK1)
IF (CT.LE.C) CALL INCREA (SK,CS,AK1)
RETURN
END

SUBROUTINE DECREA (SK,CS,AK1)
IMPLICIT REAL*8(A-H,O-Z)
C DECREASE VALUE OF AK, IF JUST INCREASED THEN REDUCE
C STEP SIZE--CS
IF (SK .EQ. 2.0) CS = CS/2.0
AK1 = CS+(1.0-CS)*AK1
SK = 1.0
RETURN
END
SUBROUTINE EIGEN (A, R, N, MV)
IMPLICIT REAL*8(A-H,O-Z)
DIMENSION A(1), R(1)
C
GENERATE IDENTITY MATRIX
RANGE = 1.0E-6
IF (MV-1) 1,4,1
1 IQ = -N
DO 3 J=1,N
IQ = IQ+N
DO 3 I=1,N
I = IC+1
R(IJ) = 0.0
IF (I-J) 3,2,3
2 R(IJ) = 1.0
3 CONTINUE
C
COMPUTE INITIAL AND FINAL NORMS (ANORM & ANORMX)
4 ANORM = 0.0
DO 6 I=1,N
DO 6 J=I,N
IF (I-J) 5,6,5
5 IA = I+J*J-J)/2
ANORM = ANORM+A(IA)*A(IA)
6 CONTINUE
IF (ANORM) 32,32,7
7 ANORM = 1.414*DSQRT(ANORM)
ANORMX = ANORM*RANGE/FLOAT(N)
C
INITIALIZE INDICATORS AND COMPUTE THRESHOLD, THR
IND = 0
THR = ANORM
8 THR = THR/ FLOAT(N)
9 L = 1
10 M = L+1
C
COMPUTE SIN AND COS
11 MQ = (M*M-M)/2
LQ = (LL-L-L)/2
LM = L+MQ
IF (DABS(A(LM))<THR) 25,12,12
12 IND = 1
LL = L+LQ
MM = M+MQ
X = 0.5*(A(LL)-A(MM))
Y = -A(LM)/DSQRT(A(LM)*A(LM)+X*X)
IF (X) 13,14,14
13 Y = -Y
14 SINX = Y/DSQRT(2.0+(DSQRT(1.0-Y*Y)))
SINX2 = SINX*SINX
COSX = DSQRT(1.0-SINX2)
COSX2 = COSX*COSX
SINCS = SINX*COSX
C
ROTATE L AND M COLUMNS
ILQ = N*(L-1)
IMQ = N*(M-1)
DO 24 I=1,N
IQ = (I*I-I-1)/2
IF (I-L) 15,22,15
15 IF (I-M) 16,22,17
16 IM = I+MQ
GO TO 18
17 IM = M+IQ
18 IF (I-L) 19,20,20
19 IL = I*LQ
20 GO TO 21
21 X = A(IL)*COSX-A(IM)*SINX
   A(IM) = A(IL)*SINX+A(IM)*COSX
   A(IL) = X
22 IF (MV-1) 23,24,23
23 ILR = ILQ+I
   IMR = IMQ+I
   X = R(ILR)*COSX-R(IMR)*SINX
   R(IMR) = R(ILR)*SINX+R(IMR)*COSX
   R(ILR) = X
24 CONTINUE
   X = 2.0*A(LM)*SINCS
   Y = A(LL)*COSX2+A(MM)*SINX2-X
   X = A(LL)*SINX2+A(MM)*COSX2+X
   A(LM) = (A(LL)-A(MM))*SINCS+A(LM)*(COSX2-SINX2)
   A(LL) = Y
   A(MM) = X
C TESTS FOR COMPLETION
C TEST FOR M = LAST COLUMN
25 IF (M-N) 26,27,26
26 M = M+1
   GO TO 11
C TEST FOR L = SECOND FROM LAST COLUMN
27 IF (L-(N-1)) 28,29,28
28 L = L+1
   GO TO 10
29 IF (IND-1) 31,30,31
30 IND = 0
   GO TO 9
C COMPARE THRESHOLD WITH FINAL NORM
31 IF (THR-ANRMX) 32,32,8
C SQRT EIGENVALUES AND EIGENVECTORS
32 IQ = -N
   N2 = N*N
   DO 36 I=1,N
   IQ = IQ+N
   LL = I+(I*I-I)/2
   JQ = N*(I-2)
   DO 36 J=I,N
   JQ = JQ+N
   MM = J+(J*J-J)/2
   IF (A(LL)-A(MM)) 33,36,36
33 X = A(LL)
   A(LL) = A(MM)
   A(MM) = X
   IF (MV-1) 34,36,34
34 DO 35 K=1,N
   ILR = IQ+K
   IMR = JQ+K
   X = R(ILR)
   R(ILR) = R(IMR)
35 R(IMR) = X
36 CONTINUE
RETURN
END
SUBROUTINE PDCORR (A,W,AS,NR,C,AK,CODE,CT,BS,MNR,
  EV,IPRINT)
  C
  THIS ROUTINE DETERMINES SHRIKUN CORRELATION MATRIX
  C
  BY USING WEIGHT MATRIX AND SATISFYING A PARTICULAR
  C
  CONDITION NUMBER
  IMPLICIT REAL*8(A-H,O-Z)
  DIMENSION A(20,20), W(20,20), AS(20,20), EV(20),
  1     STOR(400), BS(400)
  C
  EPL = 2.0E-20
  C
  FIRST DETERMINE IF SELECTED MATRIX SATISFIES
  C
  THE CONDITIONS
  CALL TARRAY (NR,20,BS,A)
  CALL EIGEN (BS,STOR,NR,1)
  NR2 = NR*NR
  N = 1
  DO 1 I=1,NR
  N = I+(I+I-1)/2
  EV(I) = BS(N)
  1 CONTINUE
  IF(IPRINT.EQ.0) GO TO 30
  WRITE (6,2) (EV(I),I=1,NR)
  2 FORMAT (10X,20F6.2)
  30 CONTINUE
  CALL SORT (EV,NR)
  IF(IPRINT.EQ.0) GO TO 40
  WRITE (6,2) (EV(I),I=1,NR)
  40 CONTINUE
  C
  DETERMINE MAX AND MIN CHAR ROOTS--COMPUTE COND. #
  CT = EV(1)/EV(2)
  AK = 1.0
  C
  DETERMINE IF THE WEIGHT MATRIX WILL YIELD A P.D.
  C
  MATRIX FOR AK=1
  CALL ASTAR (A,W,AS,NR,CT,AK,STOR,EV,EPL,BS,IPRINT)
  IF (EV(2).GE.0.0.AND.CT.LE.C) GO TO 5
  C
  IF ABOVE CONDITION IS NOT TRUE, THEN
  C
  SHRINK THE ELEMENTS OF W
  DO 4 I=1,NR
  DO 4 J=1,NR
  4 W(I,J) = DSQRT(W(I,J))
  GO TO 3
  C
  THE WEIGHT MATRIX IS OK, NOW FIND OPTIMUM AK---
  C
  START SEARCH
  5 CS = .5
  AK = 2.0E20
  CALL ASTAR (A,W,AS,NR,CT,AK,STOR,EV,EPL,BS,IPRINT)
  C
  COMPUTE STARTING VALUE FOR AK
  AK1 = 1.0+(C-1.0)/(EV(1)-C*EV(2))
  CHECKY = 0.0
  SK = 0.0
  C
  DO ITERATIVE SEARCH
  6 AK = AK1
  CALL ASTAR (A,W,AS,NR,CT,AK,STOR,EV,EPL,BS,IPRINT)
  IF (EV(2).LE.0) CALL DECREA (SK,CS,AK1)
  IF (EV(2).GT.0) CALL CHECK (CT,C,CHECKY,CS,SK,AK1)
  C
  IF OPTIMUM MATRIX IS FCUNC CHECKY=1
  IF (CHECKY.EQ.1.0) RETURN
  GO TO 6
END
SUBROUTINE INCREASE(SK,CS,AK1)
IMPLICIT REAL*8(A-H,O-Z)
C INCREASE VALUE OF AK. IF LAST DECREASED THEN REDUCE
C STEP SIZE--CS
AK1 = AK1+(AK1-1.0)*CS/(2.0*(1.0-CS))
IF (SK.EQ.1.0) CS = CS/1.5
SK = 2.0
RETURN
END

SUBROUTINE SORT(EV,NR)
IMPLICIT REAL*8(A-H,O-Z)
DIMENSION EV(1)
C SORT ROUTINE DETERMINES MAX AND MIN-----
C STORES IN EV(1) AND EV(2)
C FIND MAX
EVX = -2.0E20
DO 1 I=1,NR
IF (EV(I).GT.EVX) EVX = EV(I)
1 CONTINUE
C FIND MIN
EVN = 2.0E20
DO 2 I=1,NR
IF (EV(I).LT.EVN) EVN = EV(I)
2 CONTINUE
C STORE MAX IN EV(1) AND MIN IN EV(2)
EV(1) = EVX
EV(2) = EVN
RETURN
END

SUBROUTINE TARRAY(NR,MR,T,S)
IMPLICIT REAL*8(A-H,O-Z)
C CONVERTS SYMMETRIC MATRIX S INTO UPPER TRIANGULAR
C STORAGE MATRIX T USED FOR EIGEN
DIMENSION T(1), S(1)
M = 1
DO 1 J=1,NR
N = (J-1)*MR
DO 1 I=1,J
NN = N+I
T(M) = S(NN)
1 M = M+1
RETURN
END
SUBROUTINE GAUEL(N,A,X,B,Y,P,E,S,I)  
IMPLICIT REAL*8 (A-H,O-Z)  
REAL*8 A(1), AMULT(450),AUPP(450),ALCW(450)  
REAL*8 X(1),B(1),XPY(21)  
DO 5 I=1,N  
5 XPY(I) = B(I)  
NR=21  
NN=NR+NR  
DO 10 I=1,NN  
AMULT(I) = 0.  
AUPP(I) = 0.  
10 ALOW(I) = 0.  
NM1=N-1  
KK=-NR  
DO 200 K=1,NM1  
KK=KK+NR  
IF(IPRINT.EQ.0) GO TO 40  
WRITE(6,33) K,A(KK+K)  
33 FORMAT(/' K=',13,E16.5)  
40 IF(A(KK+K).LT.10.E-10) GO TO 990  
J=K  
L=K+1  
P = 1.0/A(KK+K)  
DO 170 I = L,N  
R = A(KK+I)*P  
AMULT(KK+I) = R  
A(KK+I) = 0.  
JJ=NR*(L-1)-NR  
DO 160 J=L,N  
JJ=JJ+NR  
A(JJ+I) = A(JJ+I) - R*A(JJ+K)  
160 CONTINUE  
B(I) = B(I) - R*B(K)  
170 CONTINUE  
200 CONTINUE  
II=-NR  
DO 205 I=1,N  
II=II+NR  
205 AMULT(II+I) = 1.  
DET = 1.  
II=-NR  
DO 220 I=1,N  
II=II+NR  
DET = DET*A(II+I)  
220 CONTINUE  
IF(DET.LT.10.E-10) GO TO 990  
NIN = NR*(N-1)  
X(N) = B(N)/A(NIN+N)  
K=N-1  
KK=NIN-NR  
305 SUM=0.  
J=N  
JJ=NIN  
310 SUM = SUM + A(JJ+K)*X(J)  
J=J-1  
JJ=JJ-NR  
IF(J<K) 999,330,310  
330 X(K) = (B(K)-SUM)/A(KK+K)  
K=K-1  
KK=KK-NR
IF(K-1) 340, 305, 305
340 IF(IPRINT.EQ.0) GO TO 400
   WRITE(6,341)
341 FORMAT(1X,'GAUSSIAN ELIMINATION',//,1X,
   1 'UPPER TRIANGULAR MATRIX',//)
   WRITE(6,360) (A(I), I=1,NN)
360 FORMAT(1X,10E13.4)
   WRITE(6,370)
370 FORMAT(//, 1X,'LOWER TRIANGULAR MATRIX',//)
   WRITE(6,360) (AMULT(I), I=1,NN)

*** COMPUTE THE INVERSE OF UPPERTRIANGULAR OF A

   II=-NR
   DO 470 I=1,N
   II=II+NR
470 AUPP(II+I) = 1.0/A(II+I)
   DO 500 JJ=1,NM1
   J=N-JJ+1
   JX=NR*(J-1)
   JM1=J-1
   DO 490 II=1,JM1
   I=J-II
   IX=NR*(I-1)
   DSUM=0.
   II=II+1
   KK=NR*I-NN
   DO 480 K=II,J
   KK=KK+NR
480 DSUM = DSUM + A(KK+I)*AUPP(JX+K)
   AUPP(JX+I) = -AUPP(IX+I)*DSUM
490 CONTINUE
500 CONTINUE
   IF(IPRINT.EQ.0) GO TO 535
   WRITE(6,510)
510 FORMAT(//,1X,'INVERSE OF UPPER TRIANGULAR MATRIX',//)
   WRITE(6,360) (AUPP(I), I=1,NN)

*** COMPUTE THE INVERSE OF LOWERTRIANGULAR OF A

   II=-NR
   DO 540 I=1,N
   II=II+NR
540 ALOW(II+I) = 1.0
   NM1 = N-1
   JJ=-NR
   DO 600 J=1,NM1
   JJ=JJ+NR
   J1=J+1
   DO 590 I=J1,N
   DSUM=0.0
   IM1=I-1
   KK=NR*(J-1)-NR
   DO 580 K=J,IM1
   KK=KK+NR
580 DSUM = DSUM + AMULT(KK+I)*ALOW(JJ+K)
   ALOW(JJ+I) = -DSUM
590 CONTINUE
600 CONTINUE
   IF(IPRINT.EQ.0) GO TO 650
WRITE(6,610)
610 FORMAT(/1x,'INVERSE OF LOWER TRIANGULAR MATRIX'/)
WRITE(6,360) (ALOW(I), I=1,NN)
C
C *** COMPUTE THE INVERSE OF A
C
650 DO 700 I=1,N
   JJ=-NR
   DO 680 J=1,N
      JJ=JJ+NR
      JJI=JJ+I
      A(JJI) = 0.0
      KK=-NR
      DO 680 K=1,N
         KK=KK+NR
         A(JJI) = A(JJI) + AUPP(KK+I)*ALOW(JJ+K)
    680 CONTINUE
    SUM=0.0
   700 CONTINUE
   DO 800 I=1,N
      SUM = SUM + X(I)*XPY(I)
      ESS = YPY - SUM
   800 GO TO 999
990 WRITE(6,991)
991 FORMAT(1x,'THE MATRIX X'*X IS SINGULAR. ')
999 RETURN
END
SUBROUTINE DSINV(A,N)
REAL*8 A(1),DIN,WORK
IPIV=N*(N+1)/2
IND=IPIV
DO 6 I=1,N
   DIN=1.0/A(IPIV)
   A(IPIV)=DIN
   MIN=N
   KEND=I-1
   LANF=N-KEND
   IF(KEND.LE.0)GOTO 5
   J=IND
   DO 4 K=1,KEND
      WORK=0.
      MIN=MIN-1
      LHOR=IPIV
      LVER=J
      DO 3 L=LANF,MIN
         LVER=LVER+1
         LHOR=LHOR+L
      3 WORK=WORK+A(LVER)*A(LHOR)
      A(J)=-WORK*DIN
   4 J=J-MIN
   5 IPIV=IPIV-MIN
   6 IND=IND-1
   DO 8 I=1,N
      IPIV=IPIV+1
      J=IPIV
      DO 8 K=1,N
         WORK=0.
         LHOR=J
      7 LHOR=LHOR+L
      WORK=WORK+A(LHOR)*A(LVER)
   8 J=J+K
   RETURN
END
SUBROUTINE SOLTON(N,A,X,B,Y,YPY,ESS)
IMPLICIT REAL*8 (A-H,O-Z)
REAL*8 A(1),X(1),B(1),YPY(1),Y(100)

*** SOLVE LY=B FOR Y.

Y(1) = B(1)/A(1)
DO 60 I=2,N
   ISTART = (I*I-I)/2 + 1
   LAST = ISTART + I - 1
   LASTM1 = LAST - 1
   DSUM = 0.
   II=0
   DO 50 J=ISTART,LASTM1
      II = II+1
      DSUM = DSUM + A(J)*Y(II)
   50 CONTINUE
   Y(I) = (B(I)-DSUM)/A(LAST)
60 CONTINUE

*** SOLVE L'X=Y FOR X.

NTT = (N*N+N)/2
X(N) = Y(N)/A(NTT)
I=N-1
70 DSUM=0.
   J=N
   II = (I*I+I)/2
80 IJ = (J*J-J)/2 + I
   DSUM = DSUM + A(IJ)*X(J)
   J=J-1
   IF(J-I) 100,90,80
90 X(I) = (Y(I)-DSUM)/A(I)
   I=I-1
   IF(I-I) 100,70,70
100 SUM=0.
   DO 200 I=1,N
200 SUM = SUM + X(I)*B(I)
   ESS = YPY - SUM
RETURN
END
SUBROUTINE DMFSD(A,N)
REAL*8 DOUB
REAL*8 A(1), DPIV, DSUM, DSQRT
KPIV=0
DO 11 K=1,N
KPIV=KPIV+K
IND=KPIV
LEND=K-1
DOUB = A(KPIV)
TOL=ABS(.01*SNGL(DOUB))
DO 11 I=K,N
DSUM=0.
IF(LEND.EQ.0)GOTO 4
DO 3 L=1,LEND
LANF=KPIV-L
LIND=IND-L
3 DSUM=DSUM+A(LANF)*A(LIND)
4 DSUM=A(IND)-DSUM
IF(I.NE.K)GO TO 10
DOUB = DSUM
IF(SNGL(DOUB)-TOL)6,6,9
6 IF(DSUM.LE.0.0)GOTO 12
KT=K-1
WRITE(6,100)KT
100 FORMAT(20X,'ROUNDING ERROR IN ROW ',I2)
9 DPIV=DSQRT(DSUM)
A(KPIV)=DPIV
DPIV=1.0/DPIV
GO TO 11
10 A(IND)=DSUM*DPIV
11 IND=IND+I
RETURN
12 WRITE(6,101)K
101 FORMAT(20X,'MATRIX IS SINGULAR AT ROW ',I2)
RETURN
END
SUBROUTINE LTG2SP(K,N,A,DMIN,V)
C LOWER TRIANGULAR G2 SWEET OPERATOR AS DESCRIBED
C BY GOODNIGHT - AMERICAN STATISTICIAN,
C AUGUST 1979 VOL 33 NO. 3
C WRITTEN BY K. E. KEMP FOR 285-725
REAL*8 A(I),DMIN(N),D,B,C
INTEGER V(N)
KDIAG=(K*K+K)/2
KK=KDIAG-K
CHECK FOR LINEAR DEPENDENCY
IF(V(K).EQ.1 .AND. A(KDIAG).LE.DMIN(K))GO TO 1
  D=1.0/A(KDIAG)
C SLEEP ROWS 1 THRU N
  DO 3 I=1,N
    IF(I.EQ.K)GO TO 3
    II=(I*I-I)/2
    IF(I.LT.K)GO TO 10
    B=A(II+K)*D
    GO TO 11
  10 B=A(KK+I)*D*V(K)*V(I)
C SLEEP COLUMNS UP TO DIAGONAL OF ROW I
  DO 11 J=1,I
    IF(J.EQ.K)GO TO 4
    IF(K.LT.J)GO TO 12
    C=A(KK+J)
    GO TO 13
  12 C=V(J)*V(K)*A((J+J-J)/2+K)
  13 A(J+J)=A(J+J)-B*C
  3 CONTINUE
C PIVOT ON KTH ROW AND COLUMN
  DO 8 J=1,K
    8 A(KK+J)=A(KK+J)*D
  DO 9 I=K,N
    IK=(I*I-I)/2+K
    9 A(IK)=-A(IK)*D
    A(KDIAG)=D
C SET INDICATOR VECTOR V
    V(K)=-V(K)
RETURN
1 WRITE(6,100)K
100 FORMAT('ROW',I3,'IS A LINEAR FUNCTION OF PREVIOUS ROWS IN MODEL')
C ZERO ROW K
  DO 2 J=1,K
    2 A(KK+J)=0.0
  IF(K.EQ.N)GO TO 6
C ZERO COLUMN K
  NM1=N-1
  IK=KDIAG
  DO 5 I=K,NM1
    IK=IK+1
    5 A(IK)=0.0
  RETURN
END
SUBROUTINE ERRINV(A,B,N,DERR,IPRINT)
IMPLICIT REAL*8 (D)
REAL*8 A(21,21),B(21,21),C(21,21)

C
C THE AMOUNT OF THE ERROR OF THE COMPUTED INVERSE
C --- THE ABSOLUTE SUM OF ELEMENTS CF
C A*B-D WHERE B IS THE COMPUTED INVERSE OF A
C AND D IS THE IDENTITY MATRIX
C
IF(IPRINT.EQ.0) GO TO 8
WRITE(6,5)
5 FORMAT(/1X,'PRODUCT OF XPX AND IPX',/)
8 DO 30 I=1,N
   DO 20 J=1,N
   C(I,J) = D(I,J)
   DO 10 K=1,N
      D(I,J) = D(I,J) + A(I,K)*B(K,J)
   10 CONTINUE
20 CONTINUE
   IF(IPRINT.EQ.0) GO TO 30
   WRITE(6,25) (D(I,J),JJ=1,N)
25 FORMAT(1X,10E13.4)
30 CONTINUE
   DO 40 I=1,N
   DO 40 J=1,N
      IF(I.EQ.J) D(I,J)=D(I,J)-1.0
      DERR = DERR + DABS(D(I,J))
   40 CONTINUE
RETURN
END
APPENDIX B1

10 X 10 CORRELATION MATRICES
WITH DIFFERENT CONDITION NUMBERS

52
<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1.0000</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>0.3832</td>
<td>1.0000</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>0.3056</td>
<td>0.9436</td>
<td>1.0000</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>0.4743</td>
<td>-0.1261</td>
<td>-0.1437</td>
<td>1.0000</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>0.1367</td>
<td>0.3821</td>
<td>0.2482</td>
<td>-0.3168</td>
<td>1.0000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>0.5361</td>
<td>0.6851</td>
<td>0.7645</td>
<td>0.2311</td>
<td>0.0201</td>
<td>1.0000</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>0.6407</td>
<td>-0.1911</td>
<td>-0.2264</td>
<td>0.5581</td>
<td>-0.2048</td>
<td>0.1169</td>
<td>1.0000</td>
</tr>
<tr>
<td>8</td>
<td>-0.8452</td>
<td>-0.0019</td>
<td>0.0677</td>
<td>-0.6163</td>
<td>0.0774</td>
<td>-0.2098</td>
<td>-0.8576</td>
</tr>
<tr>
<td>9</td>
<td>0.3945</td>
<td>-0.1314</td>
<td>-0.1342</td>
<td>0.9900</td>
<td>-0.3210</td>
<td>0.2125</td>
<td>0.4915</td>
</tr>
<tr>
<td>10</td>
<td>0.3821</td>
<td>0.6163</td>
<td>0.6013</td>
<td>0.0739</td>
<td>-0.0533</td>
<td>0.6006</td>
<td>0.1175</td>
</tr>
</tbody>
</table>

| 8 | 1.0000 |
| 9 | -0.5414 | 1.0000 |
| 10| -0.2370 | 0.0284 | 1.0000 |

```plaintext
8   9   10
```
<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1.0000</td>
<td>2</td>
<td>0.2682</td>
<td>1.0000</td>
<td>3</td>
<td>0.2139</td>
<td>0.6605</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>1.0000</td>
<td>9</td>
<td>-0.3790</td>
<td>1.0000</td>
<td>10</td>
<td>-0.1659</td>
</tr>
</tbody>
</table>

**CONDITION NUMBER = 0.10000D02**

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1.0000</td>
<td>2</td>
<td>0.3831</td>
<td>1.0000</td>
<td>3</td>
<td>0.3055</td>
<td>0.9434</td>
<td>1.0000</td>
<td>4</td>
<td>0.4742</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>1.0000</td>
<td>9</td>
<td>-0.5413</td>
<td>1.0000</td>
<td>10</td>
</tr>
</tbody>
</table>

**CONDITION NUMBER = 0.10000D04**
**CONDITION NUMBER = 0.10000D 06**

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1.0000</td>
<td>0.3846</td>
<td>1.0000</td>
<td>0.3066</td>
<td>0.9470</td>
<td>1.0000</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>0.4760</td>
<td>0.1265</td>
<td>0.1442</td>
<td>0.1372</td>
<td>0.3835</td>
<td>0.2491</td>
<td>0.3179</td>
</tr>
<tr>
<td>3</td>
<td>0.5380</td>
<td>0.6875</td>
<td>0.7672</td>
<td>0.2320</td>
<td>0.0202</td>
<td>1.0000</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>0.6429</td>
<td>0.1918</td>
<td>0.2272</td>
<td>0.5601</td>
<td>0.2055</td>
<td>0.1173</td>
<td>1.0000</td>
</tr>
<tr>
<td>5</td>
<td>0.8483</td>
<td>0.0019</td>
<td>0.0680</td>
<td>0.6185</td>
<td>0.0777</td>
<td>0.2105</td>
<td>0.8607</td>
</tr>
<tr>
<td>6</td>
<td>0.3960</td>
<td>0.1318</td>
<td>0.1347</td>
<td>0.9935</td>
<td>0.3221</td>
<td>0.2132</td>
<td>0.4933</td>
</tr>
<tr>
<td>7</td>
<td>0.3835</td>
<td>0.6185</td>
<td>0.6035</td>
<td>0.0742</td>
<td>0.0535</td>
<td>0.6027</td>
<td>0.1179</td>
</tr>
</tbody>
</table>

**CONDITION NUMBER = 0.10000D 09**

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1.0000</td>
<td>0.3846</td>
<td>1.0000</td>
<td>0.3067</td>
<td>0.9471</td>
<td>1.0000</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>0.4760</td>
<td>0.1265</td>
<td>0.1442</td>
<td>0.1372</td>
<td>0.3835</td>
<td>0.2491</td>
<td>0.3179</td>
</tr>
<tr>
<td>3</td>
<td>0.5381</td>
<td>0.6876</td>
<td>0.7672</td>
<td>0.2320</td>
<td>0.0202</td>
<td>1.0000</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>0.6430</td>
<td>0.1918</td>
<td>0.2272</td>
<td>0.5601</td>
<td>0.2055</td>
<td>0.1173</td>
<td>1.0000</td>
</tr>
<tr>
<td>5</td>
<td>0.8483</td>
<td>0.0019</td>
<td>0.0680</td>
<td>0.6186</td>
<td>0.0777</td>
<td>0.2105</td>
<td>0.8607</td>
</tr>
<tr>
<td>6</td>
<td>0.3960</td>
<td>0.1318</td>
<td>0.1347</td>
<td>0.9935</td>
<td>0.3222</td>
<td>0.2132</td>
<td>0.4933</td>
</tr>
<tr>
<td>7</td>
<td>0.3835</td>
<td>0.6185</td>
<td>0.6035</td>
<td>0.0742</td>
<td>0.0535</td>
<td>0.6028</td>
<td>0.1179</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>1.0000</td>
<td>0.5434</td>
<td>1.0000</td>
</tr>
<tr>
<td>9</td>
<td>0.2378</td>
<td>0.0285</td>
<td>1.0000</td>
</tr>
<tr>
<td>10</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
APPENDIX B2

20 X 20 CORRELATION MATRICES
WITH DIFFERENT CONDITION NUMBERS
<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1.0000</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>0.3832</td>
<td>1.0000</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>0.3056</td>
<td>0.9436</td>
<td>1.0000</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>0.4743</td>
<td>-0.1261</td>
<td>-1.437</td>
<td>1.0000</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>0.1367</td>
<td>0.3821</td>
<td>0.2482</td>
<td>-0.3168</td>
<td>1.0000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>0.5361</td>
<td>0.6851</td>
<td>0.7645</td>
<td>0.2311</td>
<td>0.0201</td>
<td>1.0000</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>0.6407</td>
<td>-0.1911</td>
<td>-0.2264</td>
<td>0.5581</td>
<td>-0.2048</td>
<td>0.1169</td>
<td>1.0000</td>
</tr>
<tr>
<td>8</td>
<td>-0.8452</td>
<td>-0.0019</td>
<td>0.0677</td>
<td>-0.6163</td>
<td>0.0774</td>
<td>-0.2058</td>
<td>-0.8576</td>
</tr>
<tr>
<td>9</td>
<td>0.3945</td>
<td>-0.1314</td>
<td>-0.1342</td>
<td>0.9900</td>
<td>-0.3210</td>
<td>0.2125</td>
<td>0.4915</td>
</tr>
<tr>
<td>10</td>
<td>0.3821</td>
<td>0.6163</td>
<td>0.6013</td>
<td>0.0739</td>
<td>-0.0533</td>
<td>0.6006</td>
<td>0.1175</td>
</tr>
<tr>
<td>11</td>
<td>0.3527</td>
<td>0.1252</td>
<td>0.4815</td>
<td>0.0914</td>
<td>0.2154</td>
<td>-0.1574</td>
<td>0.0184</td>
</tr>
<tr>
<td>12</td>
<td>-0.2418</td>
<td>0.0015</td>
<td>0.0924</td>
<td>-0.2718</td>
<td>0.0015</td>
<td>-0.4165</td>
<td>-0.0092</td>
</tr>
<tr>
<td>13</td>
<td>-0.0091</td>
<td>-0.0718</td>
<td>0.5291</td>
<td>-0.9356</td>
<td>-0.1584</td>
<td>-0.7147</td>
<td>0.0854</td>
</tr>
<tr>
<td>14</td>
<td>0.0854</td>
<td>-0.3258</td>
<td>0.1258</td>
<td>0.6154</td>
<td>0.0159</td>
<td>0.2475</td>
<td>-0.2451</td>
</tr>
<tr>
<td>15</td>
<td>0.2538</td>
<td>0.0017</td>
<td>-0.8192</td>
<td>-0.0018</td>
<td>0.0278</td>
<td>0.3124</td>
<td>-0.1159</td>
</tr>
<tr>
<td>16</td>
<td>-0.6352</td>
<td>0.0824</td>
<td>0.1456</td>
<td>0.0174</td>
<td>0.2471</td>
<td>-0.5124</td>
<td>-0.2417</td>
</tr>
<tr>
<td>17</td>
<td>0.0295</td>
<td>-0.6719</td>
<td>0.4124</td>
<td>0.0914</td>
<td>-0.5174</td>
<td>-0.0058</td>
<td>0.4840</td>
</tr>
<tr>
<td>18</td>
<td>0.9153</td>
<td>-0.2542</td>
<td>0.0180</td>
<td>-0.2415</td>
<td>-0.0170</td>
<td>0.0159</td>
<td>0.2154</td>
</tr>
<tr>
<td>19</td>
<td>-0.8124</td>
<td>0.1798</td>
<td>0.1892</td>
<td>0.4714</td>
<td>0.0090</td>
<td>0.0680</td>
<td>-0.1184</td>
</tr>
<tr>
<td>20</td>
<td>0.1736</td>
<td>0.3157</td>
<td>0.4718</td>
<td>0.2735</td>
<td>0.5481</td>
<td>0.2715</td>
<td>0.1514</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
<th>13</th>
<th>14</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>1.0000</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>-0.5414</td>
<td>1.0000</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>-0.2370</td>
<td>0.0284</td>
<td>1.0000</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>0.3696</td>
<td>0.2596</td>
<td>-0.0724</td>
<td>1.0000</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>-0.2848</td>
<td>0.1741</td>
<td>0.0058</td>
<td>0.2330</td>
<td>1.0000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>-0.0167</td>
<td>-0.5371</td>
<td>-0.9248</td>
<td>0.2884</td>
<td>0.1855</td>
<td>1.0000</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>0.0581</td>
<td>-0.4148</td>
<td>0.2746</td>
<td>-0.0673</td>
<td>-0.2505</td>
<td>0.3476</td>
<td>1.0000</td>
</tr>
<tr>
<td>15</td>
<td>-0.0074</td>
<td>0.0059</td>
<td>-0.6125</td>
<td>-0.9592</td>
<td>-0.1297</td>
<td>-0.2426</td>
<td>0.0270</td>
</tr>
<tr>
<td>16</td>
<td>-0.0185</td>
<td>0.0842</td>
<td>-0.0073</td>
<td>0.6402</td>
<td>0.5047</td>
<td>0.3021</td>
<td>-0.2306</td>
</tr>
<tr>
<td>17</td>
<td>0.6174</td>
<td>-0.2606</td>
<td>0.1852</td>
<td>-0.0831</td>
<td>-0.0849</td>
<td>0.0348</td>
<td>0.1184</td>
</tr>
<tr>
<td>18</td>
<td>0.4145</td>
<td>-0.1561</td>
<td>0.3524</td>
<td>-0.1912</td>
<td>-0.3600</td>
<td>0.1199</td>
<td>0.1074</td>
</tr>
<tr>
<td>19</td>
<td>0.6158</td>
<td>0.3475</td>
<td>0.0096</td>
<td>-0.0418</td>
<td>-0.0761</td>
<td>0.0156</td>
<td>0.0921</td>
</tr>
<tr>
<td>20</td>
<td>-0.0254</td>
<td>-0.2682</td>
<td>0.1142</td>
<td>-0.1955</td>
<td>0.0880</td>
<td>0.0443</td>
<td>0.0950</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>15</th>
<th>16</th>
<th>17</th>
<th>18</th>
<th>19</th>
<th>20</th>
</tr>
</thead>
<tbody>
<tr>
<td>15</td>
<td>1.0000</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>-0.4517</td>
<td>1.0000</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>0.1027</td>
<td>-0.0525</td>
<td>1.0000</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>0.1734</td>
<td>-0.1657</td>
<td>0.0836</td>
<td>1.0000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>0.0288</td>
<td>-0.0448</td>
<td>0.0269</td>
<td>-0.0021</td>
<td>1.0000</td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>0.1919</td>
<td>-0.0573</td>
<td>0.0124</td>
<td>-0.0529</td>
<td>-0.0059</td>
<td>1.0000</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
</tr>
<tr>
<td>---</td>
<td>-----</td>
<td>-----</td>
<td>-----</td>
<td>-----</td>
<td>-----</td>
<td>-----</td>
</tr>
<tr>
<td>1</td>
<td>1.0000</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>0.1357</td>
<td>1.0000</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>0.1082</td>
<td>0.3341</td>
<td>1.0000</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>0.1679</td>
<td>-0.0446</td>
<td>-0.509</td>
<td>1.0000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>0.0484</td>
<td>0.1353</td>
<td>0.0879</td>
<td>-0.1122</td>
<td>1.0000</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>0.1898</td>
<td>0.2426</td>
<td>0.2707</td>
<td>0.0818</td>
<td>0.0071</td>
<td>1.0000</td>
</tr>
<tr>
<td>7</td>
<td>0.2268</td>
<td>-0.0677</td>
<td>-0.0801</td>
<td>0.1976</td>
<td>-0.0725</td>
<td>0.0414</td>
</tr>
<tr>
<td>8</td>
<td>0.2993</td>
<td>-0.0007</td>
<td>0.0240</td>
<td>-0.2182</td>
<td>0.0274</td>
<td>-0.743</td>
</tr>
<tr>
<td>9</td>
<td>0.1397</td>
<td>-0.0465</td>
<td>-0.0475</td>
<td>0.3505</td>
<td>-1.136</td>
<td>0.0752</td>
</tr>
<tr>
<td>10</td>
<td>0.1353</td>
<td>0.2182</td>
<td>0.2129</td>
<td>0.0262</td>
<td>-0.0189</td>
<td>0.2126</td>
</tr>
<tr>
<td>11</td>
<td>0.1249</td>
<td>0.0443</td>
<td>0.1705</td>
<td>0.0324</td>
<td>0.0763</td>
<td>-0.557</td>
</tr>
<tr>
<td>12</td>
<td>-0.0856</td>
<td>0.0005</td>
<td>0.0327</td>
<td>-0.0962</td>
<td>0.0005</td>
<td>-0.1475</td>
</tr>
<tr>
<td>13</td>
<td>-0.0032</td>
<td>-0.0254</td>
<td>0.1909</td>
<td>-0.3312</td>
<td>-0.061</td>
<td>-0.2530</td>
</tr>
<tr>
<td>14</td>
<td>0.0302</td>
<td>-1.153</td>
<td>0.0445</td>
<td>0.2179</td>
<td>0.0556</td>
<td>0.0876</td>
</tr>
<tr>
<td>15</td>
<td>0.0899</td>
<td>0.0006</td>
<td>-2.900</td>
<td>-0.0006</td>
<td>0.0098</td>
<td>0.1106</td>
</tr>
<tr>
<td>16</td>
<td>-2.249</td>
<td>0.0292</td>
<td>0.0515</td>
<td>0.0062</td>
<td>0.0875</td>
<td>-1.814</td>
</tr>
<tr>
<td>17</td>
<td>-0.0104</td>
<td>-0.2379</td>
<td>0.1460</td>
<td>0.0324</td>
<td>-1.832</td>
<td>-0.0021</td>
</tr>
<tr>
<td>18</td>
<td>-0.3241</td>
<td>-0.0900</td>
<td>0.0064</td>
<td>-0.0855</td>
<td>0.0060</td>
<td>0.0056</td>
</tr>
<tr>
<td>19</td>
<td>-0.2876</td>
<td>0.0637</td>
<td>0.0670</td>
<td>0.1669</td>
<td>0.0032</td>
<td>0.0241</td>
</tr>
<tr>
<td>20</td>
<td>0.0615</td>
<td>0.1118</td>
<td>0.1670</td>
<td>0.0968</td>
<td>0.1941</td>
<td>0.0961</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
<th>13</th>
<th>14</th>
</tr>
</thead>
<tbody>
<tr>
<td>15</td>
<td>1.0000</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>-1.59</td>
<td>1.0000</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>0.0364</td>
<td>-0.0186</td>
<td>1.0000</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>0.0614</td>
<td>-0.0587</td>
<td>0.0296</td>
<td>1.0000</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>0.0102</td>
<td>-0.059</td>
<td>0.0095</td>
<td>-0.0007</td>
<td>1.0000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>0.0679</td>
<td>-0.0203</td>
<td>0.0044</td>
<td>-0.0187</td>
<td>-0.0021</td>
<td>1.0000</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
</tr>
<tr>
<td>---</td>
<td>------</td>
<td>------</td>
<td>------</td>
<td>------</td>
<td>------</td>
<td>------</td>
<td>------</td>
</tr>
<tr>
<td>1</td>
<td>1.0000</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>0.1737</td>
<td>1.0000</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>0.1385</td>
<td>0.4279</td>
<td>1.0000</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>0.2151</td>
<td>-0.0572</td>
<td>-0.0651</td>
<td>1.0000</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>-0.0620</td>
<td>0.1733</td>
<td>0.1126</td>
<td>-0.1436</td>
<td>1.0000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>0.2431</td>
<td>0.3106</td>
<td>0.3466</td>
<td>0.1048</td>
<td>0.0091</td>
<td>1.0000</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>0.2905</td>
<td>-0.0867</td>
<td>-1.026</td>
<td>0.2531</td>
<td>-0.0928</td>
<td>0.0530</td>
<td>1.0000</td>
</tr>
<tr>
<td>8</td>
<td>-0.3832</td>
<td>-0.0009</td>
<td>0.0307</td>
<td>-0.2795</td>
<td>0.0351</td>
<td>-0.0551</td>
<td>-0.3889</td>
</tr>
<tr>
<td>9</td>
<td>0.1789</td>
<td>-0.0596</td>
<td>-0.0608</td>
<td>0.4489</td>
<td>-1.1455</td>
<td>0.0963</td>
<td>0.2229</td>
</tr>
<tr>
<td>10</td>
<td>0.1733</td>
<td>0.2794</td>
<td>0.2726</td>
<td>0.0335</td>
<td>-0.0242</td>
<td>0.2723</td>
<td>0.0533</td>
</tr>
<tr>
<td>11</td>
<td>0.1599</td>
<td>0.0568</td>
<td>0.2183</td>
<td>0.0414</td>
<td>0.0977</td>
<td>-0.0714</td>
<td>0.0083</td>
</tr>
<tr>
<td>12</td>
<td>-1.096</td>
<td>0.0007</td>
<td>0.0419</td>
<td>-0.1232</td>
<td>0.0007</td>
<td>-1.888</td>
<td>-0.0042</td>
</tr>
<tr>
<td>13</td>
<td>-0.0041</td>
<td>-0.3262</td>
<td>0.2444</td>
<td>-0.4242</td>
<td>-0.0718</td>
<td>-3.241</td>
<td>0.0387</td>
</tr>
<tr>
<td>14</td>
<td>0.0387</td>
<td>-1.477</td>
<td>0.0570</td>
<td>0.2790</td>
<td>0.0072</td>
<td>0.1122</td>
<td>-1.111</td>
</tr>
<tr>
<td>15</td>
<td>0.1151</td>
<td>0.0008</td>
<td>-0.3714</td>
<td>-0.0008</td>
<td>0.0126</td>
<td>0.1416</td>
<td>0.0526</td>
</tr>
<tr>
<td>16</td>
<td>-0.2880</td>
<td>0.0374</td>
<td>0.0660</td>
<td>0.0079</td>
<td>0.1120</td>
<td>-2.323</td>
<td>-1.096</td>
</tr>
<tr>
<td>17</td>
<td>0.0134</td>
<td>-0.3047</td>
<td>0.1870</td>
<td>0.0414</td>
<td>-2.346</td>
<td>-0.026</td>
<td>0.2195</td>
</tr>
<tr>
<td>18</td>
<td>0.4150</td>
<td>-1.153</td>
<td>0.0082</td>
<td>-1.095</td>
<td>-0.077</td>
<td>0.0072</td>
<td>0.0977</td>
</tr>
<tr>
<td>19</td>
<td>-0.3684</td>
<td>0.0815</td>
<td>0.0858</td>
<td>0.2137</td>
<td>0.0041</td>
<td>0.0308</td>
<td>-0.0537</td>
</tr>
<tr>
<td>20</td>
<td>0.0787</td>
<td>0.1431</td>
<td>0.2139</td>
<td>0.1240</td>
<td>0.2485</td>
<td>0.1231</td>
<td>0.0686</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>1.0000</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>-0.2455</td>
<td>1.0000</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>-0.1074</td>
<td>0.0129</td>
<td>1.0000</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>0.1676</td>
<td>0.1177</td>
<td>-0.0328</td>
<td>1.0000</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>-0.1291</td>
<td>0.0789</td>
<td>0.0026</td>
<td>0.1056</td>
<td>1.0000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>0.0076</td>
<td>0.2435</td>
<td>-0.4193</td>
<td>0.1308</td>
<td>0.0841</td>
<td>1.0000</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>0.0263</td>
<td>-0.1881</td>
<td>0.1245</td>
<td>-0.0305</td>
<td>0.1136</td>
<td>0.1576</td>
<td>1.0000</td>
</tr>
<tr>
<td>15</td>
<td>0.0034</td>
<td>0.0027</td>
<td>-0.2777</td>
<td>0.0349</td>
<td>0.0588</td>
<td>-1.100</td>
<td>0.0122</td>
</tr>
<tr>
<td>16</td>
<td>-0.0084</td>
<td>0.0382</td>
<td>-0.0033</td>
<td>0.2903</td>
<td>0.2288</td>
<td>0.1370</td>
<td>-1.046</td>
</tr>
<tr>
<td>17</td>
<td>0.2799</td>
<td>-1.182</td>
<td>0.0840</td>
<td>-0.0377</td>
<td>0.3855</td>
<td>0.0158</td>
<td>0.0537</td>
</tr>
<tr>
<td>18</td>
<td>0.1879</td>
<td>-0.708</td>
<td>0.1598</td>
<td>-0.0867</td>
<td>-1.632</td>
<td>0.0544</td>
<td>0.0487</td>
</tr>
<tr>
<td>19</td>
<td>0.2792</td>
<td>0.1576</td>
<td>0.0044</td>
<td>-0.0190</td>
<td>0.0345</td>
<td>0.0071</td>
<td>0.0418</td>
</tr>
<tr>
<td>20</td>
<td>-0.0115</td>
<td>-0.1216</td>
<td>0.0518</td>
<td>-0.0886</td>
<td>0.0399</td>
<td>0.0201</td>
<td>0.0431</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>15</td>
<td>1.0000</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>-0.2048</td>
<td>1.0000</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>0.0466</td>
<td>-0.0238</td>
<td>1.0000</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>0.0786</td>
<td>-0.0751</td>
<td>0.0379</td>
<td>1.0000</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>0.0131</td>
<td>-0.0203</td>
<td>0.0122</td>
<td>-0.0010</td>
<td>1.0000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>0.0870</td>
<td>-0.0260</td>
<td>0.0056</td>
<td>-0.0240</td>
<td>-0.0027</td>
<td>1.0000</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
</tr>
<tr>
<td>1</td>
<td>1.0000</td>
<td>2</td>
<td>0.1742</td>
<td>1.0000</td>
<td>3</td>
<td>0.1389</td>
<td>0.4289</td>
</tr>
<tr>
<td>4</td>
<td>0.2156</td>
<td>-0.0573</td>
<td>-0.0653</td>
<td>1.0000</td>
<td>5</td>
<td>0.0622</td>
<td>0.1737</td>
</tr>
<tr>
<td>6</td>
<td>0.2437</td>
<td>0.3114</td>
<td>0.3475</td>
<td>0.1051</td>
<td>0.0091</td>
<td>1.0000</td>
<td>7</td>
</tr>
<tr>
<td>8</td>
<td>-0.3842</td>
<td>-0.0009</td>
<td>0.0308</td>
<td>-0.2802</td>
<td>0.0352</td>
<td>-0.0954</td>
<td>-0.3898</td>
</tr>
<tr>
<td>10</td>
<td>0.1737</td>
<td>0.2802</td>
<td>0.2733</td>
<td>0.0336</td>
<td>-0.0242</td>
<td>0.2730</td>
<td>0.0534</td>
</tr>
<tr>
<td>12</td>
<td>-0.1099</td>
<td>0.0007</td>
<td>0.0420</td>
<td>-0.1236</td>
<td>0.0007</td>
<td>-0.1893</td>
<td>-0.0042</td>
</tr>
<tr>
<td>14</td>
<td>0.0388</td>
<td>-0.1481</td>
<td>0.0572</td>
<td>0.2797</td>
<td>0.0072</td>
<td>0.1125</td>
<td>-0.1114</td>
</tr>
<tr>
<td>16</td>
<td>-0.2687</td>
<td>0.0375</td>
<td>0.0662</td>
<td>0.0079</td>
<td>0.1123</td>
<td>-0.2329</td>
<td>-0.1099</td>
</tr>
<tr>
<td>18</td>
<td>0.4161</td>
<td>-1.156</td>
<td>0.0082</td>
<td>-0.1098</td>
<td>-0.0077</td>
<td>0.0072</td>
<td>0.0979</td>
</tr>
<tr>
<td>20</td>
<td>0.0789</td>
<td>0.1435</td>
<td>0.2145</td>
<td>0.1243</td>
<td>0.2491</td>
<td>0.1234</td>
<td>0.0688</td>
</tr>
</tbody>
</table>

<p>| | | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>1.0000</td>
<td>9</td>
<td>-0.2461</td>
<td>1.0000</td>
<td>10</td>
<td>-0.1077</td>
<td>0.0129</td>
</tr>
<tr>
<td>11</td>
<td>0.1680</td>
<td>0.1180</td>
<td>-0.0329</td>
<td>1.0000</td>
<td>12</td>
<td>-0.1295</td>
<td>0.0791</td>
</tr>
<tr>
<td>13</td>
<td>-0.0076</td>
<td>-0.2441</td>
<td>-0.4204</td>
<td>0.1311</td>
<td>0.0843</td>
<td>1.0000</td>
<td>14</td>
</tr>
<tr>
<td>15</td>
<td>-0.0034</td>
<td>0.0027</td>
<td>-0.2784</td>
<td>-0.4360</td>
<td>-0.0590</td>
<td>-0.1103</td>
<td>0.0123</td>
</tr>
<tr>
<td>17</td>
<td>0.2807</td>
<td>-1.185</td>
<td>0.0842</td>
<td>-0.0378</td>
<td>-0.0386</td>
<td>0.0158</td>
<td>0.0538</td>
</tr>
<tr>
<td>19</td>
<td>0.2799</td>
<td>0.1580</td>
<td>0.0044</td>
<td>-0.0190</td>
<td>-0.0346</td>
<td>0.0071</td>
<td>0.0419</td>
</tr>
</tbody>
</table>

<p>| | | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>15</td>
<td>1.0000</td>
<td>16</td>
<td>-0.2053</td>
<td>1.0000</td>
<td>17</td>
<td>0.0467</td>
<td>-0.0239</td>
</tr>
<tr>
<td>19</td>
<td>0.0131</td>
<td>-0.0204</td>
<td>0.0122</td>
<td>-0.0010</td>
<td>1.0000</td>
<td>20</td>
<td>0.0872</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td>1</td>
<td>1.0000</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>0.1742</td>
<td>1.0000</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>0.1389</td>
<td>0.4290</td>
<td>1.0000</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>0.2156</td>
<td>-0.0573</td>
<td>-0.0653</td>
<td>1.0000</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>0.0622</td>
<td>0.1737</td>
<td>0.1128</td>
<td>-1.1440</td>
<td>1.0000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>0.2437</td>
<td>0.3114</td>
<td>0.3475</td>
<td>0.1051</td>
<td>0.0091</td>
<td>1.0000</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>0.2912</td>
<td>-0.0869</td>
<td>-0.1029</td>
<td>0.2537</td>
<td>-0.0931</td>
<td>0.0531</td>
<td>1.0000</td>
</tr>
<tr>
<td>8</td>
<td>-0.3842</td>
<td>-0.0009</td>
<td>0.0308</td>
<td>-0.2802</td>
<td>0.0352</td>
<td>-0.0954</td>
<td>-0.3899</td>
</tr>
<tr>
<td>9</td>
<td>0.1793</td>
<td>-0.0597</td>
<td>-0.0610</td>
<td>0.4500</td>
<td>-0.1459</td>
<td>0.0966</td>
<td>0.2234</td>
</tr>
<tr>
<td>10</td>
<td>0.1737</td>
<td>0.2802</td>
<td>0.2733</td>
<td>0.0336</td>
<td>-0.0242</td>
<td>0.2730</td>
<td>0.0534</td>
</tr>
<tr>
<td>11</td>
<td>0.1603</td>
<td>0.0569</td>
<td>0.2189</td>
<td>0.0415</td>
<td>0.0979</td>
<td>-0.0716</td>
<td>0.0084</td>
</tr>
<tr>
<td>12</td>
<td>-1.099</td>
<td>0.0007</td>
<td>0.0420</td>
<td>-1.236</td>
<td>0.0007</td>
<td>-0.1893</td>
<td>-0.0042</td>
</tr>
<tr>
<td>13</td>
<td>-0.041</td>
<td>-0.0326</td>
<td>0.2451</td>
<td>-0.4253</td>
<td>-0.0720</td>
<td>-0.3249</td>
<td>0.0388</td>
</tr>
<tr>
<td>14</td>
<td>0.0388</td>
<td>-0.1481</td>
<td>0.0572</td>
<td>0.2797</td>
<td>0.0072</td>
<td>0.1125</td>
<td>-1.114</td>
</tr>
<tr>
<td>15</td>
<td>0.1154</td>
<td>0.0008</td>
<td>-0.3724</td>
<td>-0.0008</td>
<td>0.0126</td>
<td>0.1420</td>
<td>0.0527</td>
</tr>
<tr>
<td>16</td>
<td>-0.2887</td>
<td>0.0375</td>
<td>0.662</td>
<td>0.0079</td>
<td>0.1123</td>
<td>-0.2329</td>
<td>-1.099</td>
</tr>
<tr>
<td>17</td>
<td>0.0134</td>
<td>-0.3054</td>
<td>0.1875</td>
<td>0.0415</td>
<td>-0.2352</td>
<td>0.0026</td>
<td>0.2200</td>
</tr>
<tr>
<td>18</td>
<td>0.4161</td>
<td>-1.156</td>
<td>0.0082</td>
<td>-1.109</td>
<td>-0.0077</td>
<td>0.0072</td>
<td>0.0979</td>
</tr>
<tr>
<td>19</td>
<td>-3.693</td>
<td>0.0817</td>
<td>0.0860</td>
<td>0.2143</td>
<td>0.0041</td>
<td>0.0309</td>
<td>-0.0538</td>
</tr>
<tr>
<td>20</td>
<td>0.0789</td>
<td>0.1435</td>
<td>0.2145</td>
<td>0.1243</td>
<td>0.2492</td>
<td>0.1234</td>
<td>0.0688</td>
</tr>
</tbody>
</table>

<p>| | | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>1.0000</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>-0.2461</td>
<td>1.0000</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>-0.1077</td>
<td>0.0129</td>
<td>1.0000</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>0.1680</td>
<td>0.1180</td>
<td>-0.0329</td>
<td>1.0000</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>-0.1295</td>
<td>0.0791</td>
<td>0.0026</td>
<td>0.1059</td>
<td>1.0000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>-0.0076</td>
<td>-0.2442</td>
<td>-0.4204</td>
<td>0.1311</td>
<td>0.0843</td>
<td>1.0000</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>0.0264</td>
<td>-1.886</td>
<td>0.1248</td>
<td>-0.0306</td>
<td>-1.139</td>
<td>0.1580</td>
<td>1.0000</td>
</tr>
<tr>
<td>15</td>
<td>-0.0034</td>
<td>0.0027</td>
<td>-0.2784</td>
<td>-0.4360</td>
<td>-0.0590</td>
<td>-1.103</td>
<td>0.0123</td>
</tr>
<tr>
<td>16</td>
<td>-0.0084</td>
<td>0.0383</td>
<td>-0.0033</td>
<td>0.2910</td>
<td>0.2294</td>
<td>0.1373</td>
<td>-1.048</td>
</tr>
<tr>
<td>17</td>
<td>0.2807</td>
<td>-1.1185</td>
<td>0.0842</td>
<td>-0.0378</td>
<td>-0.0386</td>
<td>0.0158</td>
<td>0.0538</td>
</tr>
<tr>
<td>18</td>
<td>0.1884</td>
<td>-0.0710</td>
<td>0.1602</td>
<td>-0.0869</td>
<td>-0.1636</td>
<td>0.0545</td>
<td>0.0488</td>
</tr>
<tr>
<td>19</td>
<td>0.2799</td>
<td>0.1580</td>
<td>0.0044</td>
<td>-0.0190</td>
<td>-0.0346</td>
<td>0.0071</td>
<td>0.0419</td>
</tr>
<tr>
<td>20</td>
<td>-0.0115</td>
<td>-1.219</td>
<td>0.0519</td>
<td>-0.0889</td>
<td>0.0400</td>
<td>0.0201</td>
<td>0.0432</td>
</tr>
</tbody>
</table>

<p>| | | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>15</td>
<td>1.0000</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>-0.2053</td>
<td>1.0000</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>0.0467</td>
<td>-0.0239</td>
<td>1.0000</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>0.0788</td>
<td>-0.0753</td>
<td>0.0380</td>
<td>1.0000</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>0.0131</td>
<td>-0.0204</td>
<td>0.0122</td>
<td>-0.0010</td>
<td>1.0000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>0.0872</td>
<td>-0.0260</td>
<td>0.0056</td>
<td>-0.0240</td>
<td>-0.0027</td>
<td>1.0000</td>
<td></td>
</tr>
</tbody>
</table>
A COMPARISON OF ALGORITHMS FOR LEAST SQUARES ESTIMATES OF PARAMETERS IN THE LINEAR MODEL

by

CHUL H. AHN

B.S., Seoul National University, 1976

AN ABSTRACT OF A MASTER'S REPORT

submitted in partial fulfillment of the requirements for the degree

MASTER OF SCIENCE

Department of Statistics

KANSAS STATE UNIVERSITY

Manhattan, Kansas

1981
This report compared three algorithms for least squares estimates in the linear model: Gaussian elimination, Cholesky method, and the Sweep operator.

The comparison criteria were:

1. the number of arithmetic operations each algorithm requires
2. empirical data (accuracy and execution time).

As for the number of arithmetic operations Cholesky method requires much fewer operations than others. For accuracy comparison based on empirical data we concentrated on the accuracy of the computed inverse of $X'X$ which would mainly effect the accuracy of the solution vector estimates because we couldn't find the exact solution to the solution vector. We generated $X'X$ matrices from well-conditioned to ill-conditioned (condition number: $10 - 10^9$), and computed their inverse using the three algorithms.

To generate $X'X$ matrix we first generated data matrix given the distribution and the correlation matrix.

Results showed that Cholesky method was best for both accuracy and execution time.