NBER WORKING PAPER SERIES

THE SINGULAR VALUE ANALYSIS IN MATRIX COMPUTATION

Richard Becker*
Neil Kaden*
Virginia Klema*

Working Paper No. 46

COMPUTER RESEARCH CENTER FOR ECONOMICS AND MANAGEMENT SCIENCE
National Bureau of Economic Research, Inc.
575 Technology Square
Cambridge, Massachusetts 02139

July 1974

Preliminary: not for quotation

*NBER Computer Research Center. Research supported in part by National Science Foundation Grant GJ-1154X3 to the National Bureau of Economic Research, Inc.

Abstract

This paper discusses the robustness and the computational stability of the singular value decomposition algorithm used at the NBER Computer Research Cneter. The effect of perturbations on input data is explored. Suggestions are made for using the algorithm to get information about the rank of a real square or rectangular matrix. The algorithm can also be used to compute the best approximate solution of linear systems of equations in the least squares sense, to solve linear systems of equations with equality constraints, and to determine dependencies or near dependencies among the rows or columns of a matrix.

A copy of the subroutine that is used and some examples on which it has been tested are included in the appendixes.

Contents

Appendixes

The singular value decomposition of a matrix is one of the most elegant algorithms in numerical algebra for exposing quantitative information about the structure of a system of linear equations. It can be used to get information about the rank of a square or rectangular matrix, to compute the best approximate solution of a linear system of equations in the least squares sense, to solve systems of linear equations with equality constraints, and to determine dependencies or near-dependencies among the rows or columns of a matrix. Occasionally the singular value decomposition is used in the iterations of linear systems that tend toward the solution of nonlinear systems of equations. The condition number of a matrix with respect to the solution of a linear system of equations is a by-product of the singular value decomposition as is the production of the pseudo-inverse and the solution of homogeneous systems of equations.

The condition number of a matrix with respect to the solution of a linear system of equations shows how well the vector x is defined by the transformation $Ax=b$. The condition number $\kappa(A)$ of the nonsingular matrix A is the ratio $\frac{\sigma_{max}}{\sigma_{min}}$ where σ_{max} and σ_{min} are, respectively, the maximum and minimum singular values of A (i.e., the non-negative square roots of the eigenvalues of A^TA where A^T denotes the transpose of A). For example, if $\kappa(A)=10^6$, a perturbation of 2^{-20} in the elements of A can change the computed solution $\hat{x}$ by a factor of $2^{-20} \cdot 10^6$, that is to say, even the leading digit may be changed. For a more rigorously detailed explanation, see [9].*

*Numerals in square brackets refer to entries in the Reference section, p. 15.

In the discussion that follows, we seek to compute directly the best approximate solution to the possibly over-determined or under-determined system of equations

$$Ax = b.$$

The singular value decomposition is used to obtain this solution. Frequently a user, or a problem originator, poses a problem from which he wants to obtain a solution vector x in the sense of least squares from the system of equations

$$A^TAx=A^Tb.$$

Possibly he thinks the information he needs comes from the solution

$$x=(A^TA)^IA^Tb.$$

Classically, (1) if the data matrix A and the vector b are exact (that is to say, there is no uncertainty in the data A and b), (2) if the precision of the arithmetic of the machine is such that A^TA can be formed and stored exactly, and (3) if A^TA is of full rank, the solution x could be obtained from $(A^TA)^IA^Tb$. However, given that these three conditions are seldom attainable in practice, the solution should not be computed in this way because of the extra precision that is required. Furthermore, unless there is a priori <u>exact</u> information known about the rank of A, the solution x cannot be obtained from the pseudo-inverse of A with any more

authenticity than from $(A^TA)^I$. That is to say the rank should be determined during the course of computing the singular value decomposition. Reliable information about rank deficiency cannot be obtained from triangular factorization.

Sylvester wrote an article on the singular value decomposition of real nxn matrices in 1889 [10]. Eckert and Young extended the work to general matrices in 1936 [1]. The definitive paper on calculating the singular value decomposition was written by Golub and Kahan [2]. Though the paper was published in 1965, it is fair to say that its use as a robust tool of mathematical software is recent and, as of now, is not very widespread (see [4] and [5]).

The singular values of the matrix A and the non-negative square roots of the eigenvalues of the symmetric matrix A^TA are mathematically equal, but may be different computationally. Singular values correct to working accuracy for the matrix A can often be computed when certain small eigenvalues cannot be computed for A^TA. This fact is not startling. It is caused by the perturbation of an exact A^TA introduced in the multiplication of A^T by A. There are many examples of such matrices, one of which is illustrated in [9], assuming a 4-decimal-place machine, as

$$A = \begin{bmatrix} 1.005 & 0.995 \\ .995 & 1.005 \end{bmatrix}$$

having singular values 2.0 and .01. The matrix A^TA in 4-decimal arithmetic is

$$A^TA = \begin{bmatrix} 2.000 & 2.000 \\ 2.000 & 2.000 \end{bmatrix}$$

with eigenvalues 4.0 and 0.0. Attrition in forming A^TA has obscured all information about the smaller singular value.

The subroutine MINFIT, using the notation in [2], reduces the system of equations

$$Ax = b$$

where A has m rows and n columns (m can be less than, equal to, or greater than n) to the form

$$U \Sigma V^T x = b$$

giving

$$\Sigma V^T x = U^T b.$$

The columns of V are the orthonormal eigenvectors of A^TA. The transformation U^Tb is formed directly -- U is not computed explicitly. The columns of U are the orthonormal eigenvectors of AA^T. If one needs the explicit columns of U he should append the identity matrix I_m to the right-hand side b. There is no restriction, at the subroutine level, on the number of columns of b; it can be zero.

The diagonal matrix, Σ, contains the singular values of A. The transformations used to obtain the decomposition preserve unitarily invariant norms, thereby assuring that the norm of Σ is that of A. The diagonal elements of Σ, when ordered, are $\sigma_1 \geq \sigma_2 \geq \sigma_3 \ldots \geq \sigma_n \geq 0$. MINFIT does not order the singular values. Given information about the certainty of the data A and b, one can choose the best approximating matrix A_r of full rank that is nearest, in the norm sense, to the matrix A. From A_r the best candidate solution x for Ax=b can be computed. If σ_r is chosen such that

$\sigma_1 \geq \sigma_2 \cdots \geq \sigma_r > 0$, $\sigma_{r+1} \geq \sigma_{r+2} \cdots \geq \sigma_n$, whereby $\sigma_{r+1}, \cdots, \sigma_n$ are effectively considered to be zero, the condition number of A is the ratio, $\frac{\sigma_1}{\sigma_r}$. If the matrix A is equilibrated, i.e., scaled, so that $\sigma_1=1$, σ_r should be not less than the square root of the machine precision, or a constant representing the uncertainty in the data, whichever is larger. To be arbitrary about the choice of σ_r relative to σ_1 is difficult. At the NBER Computer Research Center we have chosen a rank tolerance equal to the floating point representation of 2^{-26}, the square root of the machine precision, 2^{-52}. There is an obvious danger that this range rolerance may be inadequate for some problesm. For example suppose that $A=U\ \Sigma\ V^T$ such that

$$\Sigma = \begin{bmatrix} 1 & & & & & & \\ & 2^{-2} & & & & & \\ & & 2^{-3} & & & & \\ & & & 2^{-26} + \varepsilon & & & \\ & & & & 2^{-26} & & \\ & & & & & 2^{-27} & \\ & & & & & & \ddots \end{bmatrix}$$

where $2^{-52} \leq \varepsilon < 2^{-26}$, say.

The arbitrary rank tolerance would leave σ_4 unchanged but set σ_5 to zero. Thus A_4 would be deemed to have full rank whereas a more judicious choice of rank is 3. This example, though artificial, is given to encourage all users to display the diagonal matrix, Σ, to see his particular problem's distribution of the σ_i.

Given an appropriate choice of σ_r

$$||A|| - ||A_r|| \leq \left(\sum_{i=r+1}^{n} (\sigma_i)^2\right)^{1/2}$$

where $||\cdot||$ indicates the Frobenius norm, i.e. $||A|| = \left(\sum_{\substack{i=1,m \\ j=1,n}} (a_{ij})^2\right)^{1/2}$.

Noting that $U^TU = V^TV = VV^T = I_n$ and that the pseudo-inverse of Σ is the diagonal matrix

$$\Sigma^+ = \begin{bmatrix} \frac{1}{\sigma_1} & & & & & & \\ & \frac{1}{\sigma_2} & & & & & \\ & & \ddots & & & & \\ & & & \frac{1}{\sigma_r} & & & \\ & & & & 0 & & \\ & & & & & \ddots & \\ & & & & & & 0 \end{bmatrix}$$

the pseudo-inverse of A is

$$A^+ = V\Sigma^+ U^T.$$

There is seldom any reason to form a pseudo-inverse explicitly. MINFIT accumulates Householder transformations to produce a bidiagonal matrix having the same singular values as A, and continues, by a variant of the QR algorithm (see [3]), to diagonalize the bidiagonal form to give

$$\Sigma V^T x = U^T b = c$$

from which

$$x = V\Sigma^+ c.$$

Various candidate solutions x can be provided by different choices of a rank tolerance to fix σ_r. See [6], chapters 25 and 26.

For suitably chosen σ_r, consider those columns of V associated with $\sigma_{r+1}, \sigma_{r+2}, \ldots \sigma_n$ as V_ν, namely the columns of V that span the null space of A. Then

$$AV_\nu = 0.$$

When such columns V_ν exist, they constitute the non-trivial solutions of the homogenous system of equations

$$AX = 0.$$

The elements of the columns of V can be inspected to reveal dependencies or near dependencies among the columns, i.e., the variables of the coefficient matrix A. Analogously, the columns of U can reveal dependencies among the equations, i.e., the rows of A.

In using MINFIT, and providing it to other users, we are concerned with three distinct but related items, (1) the stability of the algorithm from the standpoint of numerical algebra, (2) the robustness of the mathematical software that implements the algorithm, and (3) the documentation that provides information on the use of the mathematical software.

The numerical stability of an algorithm usually means that the solution that is computed is the exact solution of a neighboring problem and that the neighboring problem can be defined in the sense of a backward error analysis. Such analysis for the singular value decomposition has been

published in [2], [11], [12], and [13]. The singular value decomposition is stable in the sense that the computation of eigensystems of Hermitian matrices is stable. In general, we expect

$$\frac{||A - U\Sigma V^T||}{||A||}$$

to be the order of machine precision, as is corroborated for the matrices in Appendix C. If this criterion should not be met for some matrix, A, the authors would like to know about it. For computational convenience we computed $\frac{||A|| - ||U\Sigma V^T||}{||A||}$ for the test matrices.

Robustness of this mathematical software is established to the extent of exposing test matrices on which the algorithm has performed correctly. Professor Gene Golub suggested two additional tests. These are

1) Decompose A to give $U\Sigma V^T$. Permute σ_i, reform $A=U\Sigma V^T$, and recompute the decomposition. This gives the effect of a perturbation on A in the sense that the resulting decomposition will show a permutation of the columns of U and V, yet give the same singular values of A. As additional tests we have taken orthonormal matrices U and V, particular σ_i, formed $U(\Sigma V^T)=A$ and computed $A=U\Sigma V^T$. Denote the maximum singular value by σ_{max} and the minimum singular value by σ_{min}. If $\frac{\sigma_{min}}{\sigma_{max}}$ is less than the relative machine precision, the computed σ_{min} may not be less than the relative precision of the machine on which it is computed, i.e. 2^{-52} for long

precision, 2^{-20} for short precision, on the IBM 360/370 machines.

2) Calculate the residuals r= Ax-b to observe the error between the true solution x and the computed solution $\hat{x}$. From Golub's formulation

$$\frac{||x - \hat{x}||}{||x||} \leq \varepsilon\kappa(A) + \varepsilon\kappa^2(A) \quad \frac{||r||}{||\hat{x}||}$$

in which the condition number $\kappa(A) = \frac{\sigma_{max}}{\sigma_{min}}$.

The second term on the right-hand side is dominant for least squares problems. In seeking the candidate solution $\hat{x}_k$ of least norm we compute

$$\mu_\kappa = \kappa^2(A_k) \frac{||r_k||}{||\hat{x}_k||}$$

for different choices of k. We could compute μ_k directly by forming $\hat{x}_k$ and r_k. However, taking advantage of $||U|| = ||V|| = 1$ it follows that

$$\mu_k = \left(\frac{\sigma_1}{\sigma_k}\right)^2 \frac{\left(\sum_{i=k+1}^{m} c_i^2\right)^{1/2}}{\left(\sum_{i=1}^{k} \frac{c_i^2}{\sigma_i^2}\right)^{1/2}}$$

where $c = U^T b$. This formulation permits the appropriate choice of the best approximating matrix $A_r = U\Sigma_r V^T$ from the minimum μ_r without explicitly computing the candidate solutions $\hat{x}_k$. The best approximate solution is obtained when μ_k is minimum.

Frequently the question is raised about using iterative methods for computing the singular value decomposition. There is an excellent discussion of such issues in [8] along with suggestions for constructing matrices with exact singular values.

Informally, we suggest certain guidelines for using MINFIT. Whenever possible one should avoid forming the product of a matrix by its transpose. Note that the eigenvalues Λ and eigenvectors X for the real symmetric matrix eigenproblem

$$AX = X\Lambda$$

are immediately available from MINFIT without ever forming A^TA. However, if the original problem is to obtain the eigensystem of a real symmetric positive definite, negative definite, or indefinite

matrix, SYMEIG (see [7]) should be used. One should, however, be warned that the appearance of zero or negative eigenvalues for a matrix believed to be positive definite signals the need to analyze the original data or the construction of the problem more carefully by obtaining the singular value decomposition of the original data matrix.

MINFIT can be used to obtain the solution of a linear system of equations. However, if the matrix of coefficients is known to have full rank, and, if the condition number of this matrix is small relative to the uncertainty in the data, one of the matrix factorization methods should be used. Such matrix factorization methods are 1) the Choleski factorization, 2) the LU decomposition with partial or complete pivoting where the elementary transformations have been stabilized by row and/or column interchanges, and 3) the orthogonal factorization with column pivoting. However, such factorizations cannot be guaranteed to give definitive information about the condition number of a matrix. Consider from [14] the bidiagonal matrix of order 100

$$\begin{bmatrix} .501 & -1 & & & & & \\ & .502 & -1 & & & & \\ & & \ddots & \ddots & & & \\ & & & \ddots & \ddots & & \\ & & & & .509 & -1 & \\ & & & & & .600 & \end{bmatrix}.$$

This matrix is extremely ill-conditioned with respect to the solution of a linear system of equations. Its smallest singular value is approximately 10^{-22} despite the fact that its smallest eigenvalue is .501. This matrix also shows that computation of the smallest eigenvalue is limited by the relative finite precision of the machine on which it is computed. That is to say, the small singular value, 10^{-22} will appear computationally to be no smaller than the order of machine precision. This result is not attributable to the construction of the algorithm, but rather to the finite precision of the machine's arithmetic.

We suggest everywhere the use of long precision on the IBM 360/370 machines to compute the solutions of linear systems of equations, eigensystems, and the singular value decomposition. Even so, we urge extreme caution wherever the number of rows, m, or the number of columns, n, of a matrix is of more than modest size, say 200, if the matrix is dense. The quantity $\frac{||A - U(\Sigma V^T)||}{||A|| \cdot \max(m,n)}$ should be the order of machine precision. However, the computational algorithms are, in general, $O(n^3)$ or $O(mn^2)$ processes. We advise a rigorous analysis of the structure of a matrix of high dimensions before any of the numerical algebra algorithms are used. See Appendix C for some timing results on random matrices.

The singular values of a matrix can be substantially altered by scaling the original data matrix as is shown by the examples in Appendix C. Deliberately, MINFIT does not include scaling of the rows or columns of the matrix A or right-hand sides b. For the best performance of the algorithm we suggest that columns of A be equilibriated such that the sums of their elements be as nearly equal as possible. Exact powers of

16 for the 360/370 machines should be used for scaling factors so that the data is not perturbed in trailing digits. Row scaling will have the effect of introducing weights on the data in a least squares problem and therefore should be done at a user's discretion. An excellent discussion of scaling is in [6].

Lawson further points out in [6] that it is important to take advantage of information about the certainty of data. For example, if data is known to have uncertainty in the third decimal place, that digit and all that follow are arbitrary. The matrix

$$\begin{bmatrix} 1.02 & 1.09 \\ 1.05 & 1.01 \end{bmatrix}$$

if uncertain in the third figure could lead to

$$\begin{bmatrix} 1.00 & 1.00 \\ 1.00 & 1.00 \end{bmatrix} .$$

The eigenvectors of a symmetric matrix, and therefore, the singular vectors U and V from MINFIT are known only to within a constant multiplier of modulus 1. If anyone should attempt to recompute the results in Appendix C on a machine whose arithmetic is different from that of the IBM 360/67 he may observe a change in sign on the columns of U or V.

The Fortran IV subroutine MINFIT, imbedded in TROLL (see [7]), that forms the singular value decomposition and obtains a best approximate solution vector x is an adaptation of ANLF233S from the Argonne National Laboratory. ANLF233S written by Burton Garbow, ANL, is a Fortran IV translation, with certain modifications, of the Algol 60 procedure MINFIT [3]. We have augmented

ANLF233S by adding comments and producing the numerically best approximate solution x based on a particular rank tolerance chosen for the IBM 360/370 long precision arithmetic. The machine epsilon, that is, the smallest number, $\varepsilon > 0$, for which $1 + \varepsilon > 1$ is the floating point representation of $16^{-13} = 2^{-52}$ for the IBM 360/370 machines. The comments and the Fortran IV listing of the subroutine used at the Center is given in Appendix A. The description of the parameters for the TROLL interface is given in Appendix B. Appendix C contains selected matrices, computed solutions, and residual norm checks obtained from driver programs that use the singular value decomposition. These results were computed on the IBM 360/67. Comments, questions, or criticims of this subroutine should be brought to the attention of the authors of this working paper.

References

1. Eckert, C., and Young, G. (1936) "The Approximation of One Matrix by Another of Lower Rank," Psychometrika 1, 211-218.

2. Golub, G. and Kahan, W., "Calculating the Singular Values and Pseudo-inverse of a Matrix," in J. SIAM. Numer. Anal. SER B 2, 205-224 (1965).

3. Golub, G. H. and Reinsch, C., "Singular Value Decomposition and Least Squares Solutions," in J. H. Wilkinson and C. Reinsch (eds.) Handbook for Automatic Computation, Volume II: Linear Algebra, Springer Verlag, 134-151 (1971); prepublished in Numer. Math. 14, 403-420 (1970).

4. Hanson, R. and Lawson, C. L., "Extensions and Applications of the Householder Algorithm for Solving Linear Least Squares Problems," Mathematics of Computation, vol. 23, no. 108, 787-812 (1969).

5. Lawson, C. L. "Applications of Singular Value Analysis" in John Rice (ed.), Mathematical Software, Academic Press, Chapter 25 (1971).

6. Lawson, C. L., and Hanson, R. J., Solving Least Squares Problems, Prentice-Hall, (1974).

7. National Bureau of Economic Research, TROLL Experimental Programs: Numerical Methods, (1974).

8. Soderstrom, Torsten and Stewart, G. W.,"On the Numerical Properties of an Iterative Method for Computing the Moore-Penrose Generalized Inverse, SIAM J. Numer. Anal Vol. 11, No. 1 (1974), 61-74.

9. Stewart, G. W., Introduction to Matrix Computations, Academic Press, 380-387 (1973).

10. Sylvester, J. J., Messenger of Math 19, 42 (1889).

11. Wedin, Per-Ake, "Perturbation Bounds in Connection with Singular Value Decomposition," BIT 12 (1972), 99-111.

12. Wedin, Per-Ake, "Perturbation Theory for Pseudo-Inverses," BIT 13 (1973), 217-232.

13. Wedin, Per-Ake, "On the Almost Rank Deficient Case of the Least Squares Problem," BIT (1973) 344-354.

14. Wilkinson, J. H., The Algebraic Eigenvalue Problem, Clarendon Press (1965), 195.

APPENDIX A: Listing of the Fortran IV Program MINFIT

```
      SUBROUTINE MINFIT(NM,M,N,A,W,IP,B,IERR,RV1,RETX)
C
      INTEGER I,J,K,L,M,N,II,IP,I1,KK,K1,LL,L1,M1,NM,ITS,IERR
      REAL*8 A(NM,N),W(N),B(NM,IP),RV1(N)
      REAL*8 C,F,G,H,S,X,Y,Z,EPS,SCALE,MACHEP,RKTOL
      REAL*8 DSQRT,DMAX1,DABS,DSIGN
      LOGICAL RETX
C
C     THIS SUBROUTINE DETERMINES, TOWARDS THE SOLUTION OF THE LINEAR
C                                                          T
C     SYSTEM AX=B, THE SINGULAR VALUE DECOMPOSITION A=USV  OF A REAL
C                                                    T
C     M BY N RECTANGULAR MATRIX, FORMING U B REATHER THAN U. HOUSEHOLDER
C     BIDIAGONALIZATION AND A VARIANT OF THE QR ALGORITHM ARE USED.
C     THIS SUBROUTINE COMPUTES A CANDIDATE SOLUTION X WHEN THE
C     LOGICAL INPUT PARAMETER RETX IS SET .TRUE.  THIS CANDIDATE
C     SOLUTION IS BASED ON THE RANK TOLERANCE SET TO
C     2.0D0**(-26), THE SQUARE ROOT OF THE MACHINE PRECISION
C     2.0D0**(-52).
C
C     ON INPUT:
C
C        NM MUST BE SET TO THE ROW DIMENSION OF THE TWO-DIMENSIONAL
C          ARRAY PARAMETERS AS DECLARED IN THE CALLING PROGRAM
C          DIMENSION STATEMENT.  NOTE THAT NM MUST BE AT LEAST
C          AS LARGE AS THE MAXIMUM OF M AND N;
C
C        M IS THE NUMBER OF ROWS OF A AND B;
C
C        N IS THE NUMBER OF COLUMNS OF A AND THE ORDER OF V;
C
C        A CONTAINS THE RECTANGULAR COEFFICIENT MATRIX OF THE SYSTEM;
C
C        IP IS THE NUMBER OF COLUMNS OF B. IP CAN BE ZERO;
C
C        B CONTAINS THE CONSTANT COLUMN MATRIX OF THE SYSTEM
C          IF IP IS NOT ZERO. OTHERWISE B IS NOT REFERENCED.
C
C        RETX MUST BE SET .TRUE. IF THE CANDIDATE SOLUTION X IS TO
C          BE COMPUTED.  IF ONLY THE SINGULAR VALUE DECOMPOSITION IS
C          DESIRED, SET RETX .FALSE.
C
C
C     ON OUTPUT:
C
C        A HAS BEEN OVERWRITTEN BY THE MATRIX V (ORTHOGONAL) OF THE
C          DECOMPOSITION IN ITS FIRST N ROWS AND COLUMNS. IF AN
C          ERROR EXIT IS MADE, THE COLUMNS OF V CORRESPONDING TO
C          INDICES OF CORRECT SINGULAR VALUES SHOULD BE CORRECT;
C
C        W CONTAINS THE N (NON-NEGATIVE) SINGULAR VALUES OF A (THE
C          DIAGONAL ELEMENTS OF S). THEY ARE UNORDERED. IF AN
C          ERROR EXIT IS MADE, THE SINGULAR VALUES SHOULD BE CORRECT
C          FOR INDICES IERR+1,IERR+2,...,N;
C
```

```
C                                        T
C        B HAS BEEN OVERWRITTEN BY U B. IF AN ERROR EXIT IS MADE,
C                      T
C          THE ROWS OF U B CORRESPONDING TO INDICES OF CORRECT
C          SINGULAR VALUES SHOULD BE CORRECT;
C
C     IF RETX IS TRUE, W WILL CONTAIN THE DIAGONAL OF THE PSEUDOINVERSE
C       OF THE DIAGONAL MATRIX S.  ANY SINGULAR VALUES THAT
C       ARE LESS THAN RKTOL TIMES THE LARGEST SINGLUAR VALUE ARE
C       SET TO ZERO IN THE PSEUDOINVERSE.
C                                                              T
C       ALSO, THE SOLUTION X IS RETURNED IN B, REPLACING U B.
C
C        IERR IS SET TO
C          ZERO     FOR NORMAL RETURN,
C          K        IF THE K-TH SINGULAR VALUE HAS NOT BEEN
C                   DETERMINED AFTER 30 ITERATIONS,
C          -1       IF THE MAXIMUM SINGULAR VALUE IS ZERO (INDICATING
C                   A ZERO A MATRIX ON INPUT).  ONLY SET IF
C                   RETX IS .TRUE..
C
C        RV1 IS A TEMPORARY STORAGE ARRAY.
C
C     ------------------------------------------------------------------
C
C     ::::::::::   MACHEP IS A MACHINE DEPENDENT PARAMETER SPECIFYING
C              THE RELATIVE PRECISION OF FLOATING POINT ARITHMETIC
C              MACHEP = 16.0D0**(-13) FOR LONG FORM ARITHMETIC
C              ON S360 ::::::::::::::
      DATA MACHEP/Z3410000000000000/
C     ::::::::::   RKTOL, FOR THESE APPLICATIONS, IS THE SQUARE
C              ROOT OF MACHEP ::::::::::::::
      DATA RKTOL/Z3A40000000000000/
C     :::::::::: HOUSEHOLDER REDUCTION TO BIDIAGONAL FORM ::::::::::
      IERR = 0
      G = 0.0D0
      SCALE = 0.0D0
      X = 0.0D0
C
      DO 300 I = 1, N
         L = I + 1
         RV1(I) = SCALE * G
         G = 0.0D0
         S = 0.0D0
         SCALE = 0.0D0
         IF (I .GT. M) GO TO 210
C
         DO 120 K = I, M
  120    SCALE = SCALE + DABS(A(K,I))
C
         IF (SCALE .EQ. 0.0D0) GO TO 210
C
         DO 130 K = I, M
            A(K,I) = A(K,I) / SCALE
            S = S + A(K,I)**2
```

```
  130     CONTINUE
C
          F = A(I,I)
          G = -DSIGN(DSQRT(S),F)
          H = F * G - S
          A(I,I) = F - G
          IF (I .EQ. N) GO TO 160
C
          DO 150 J = L, N
             S = 0.0D0
C
             DO 140 K = I, M
  140        S = S + A(K,I) * A(K,J)
C
             F = S / H
C
             DO 150 K = I, M
                A(K,J) = A(K,J) + F * A(K,I)
  150     CONTINUE
C
  160 IF (IP .EQ. 0) GO TO 190
C
      DO 180 J = 1,IP
         S = 0.0D0
C
         DO 170 K = I, M
  170    S = S + A(K,I) * B(K,J)
C
         F = S / H
C
         DO 180 K = I, M
            B(K,J) = B(K,J) + F * A(K,I)
  180 CONTINUE
C
  190    DO 200 K = I, M
  200    A(K,I) = SCALE * A(K,I)
C
  210    W(I) = SCALE * G
         G = 0.0D0
         S = 0.0D0
         SCALE = 0.0D0
         IF (I .GT. M .OR. I .EQ. N) GO TO 290
C
         DO 220 K = L, N
  220    SCALE = SCALE + DABS(A(I,K))
C
         IF (SCALE .EQ. 0.0D0) GO TO 290
C
         DO 230 K = L, N
            A(I,K) = A(I,K) / SCALE
            S = S + A(I,K)**2
  230    CONTINUE
C
         F = A(I,L)
         G = -DSIGN(DSQRT(S),F)
```

```
            H = F * G - S
            A(I,L) = F - G
C
            DO 240 K = L, N
  240       RV1(K) = A(I,K) / H
C
         IF (I .EQ. M) GO TO 270
C
            DO 260 J = L, M
               S = 0.0D0
C
               DO 250 K = L, N
  250          S = S + A(J,K) * A(I,K)
C
               DO 260 K = L, N
                  A(J,K) = A(J,K) + S * RV1(K)
  260       CONTINUE
C
  270       DO 280 K = L, N
  280       A(I,K) = SCALE * A(I,K)
C
  290       X = DMAX1(X,DABS(W(I))+DABS(RV1(I)))
  300    CONTINUE
C        :::::::::: ACCUMULATION OF RIGHT-HAND TRANSFORMATIONS ::::::::::
C        :::::::::: FOR I=N STEP -1 UNTIL 1 DO -- ::::::::::
         DO 400 II = 1, N
            I = N + 1 - II
            IF (I .EQ. N) GO TO 390
            IF (G .EQ. 0.0D0) GO TO 360
            H = A(I,L) * G
C
            DO 320 J = L, N
  320       A(J,I) = A(I,J) / H
C
            DO 350 J = L, N
               S = 0.0D0
C
               DO 340 K = L, N
  340          S = S + A(I,K) * A(K,J)
C
               DO 350 K = L, N
                  A(K,J) = A(K,J) + S * A(K,I)
  350       CONTINUE
C
  360       DO 380 J = L, N
               A(I,J) = 0.0D0
               A(J,I) = 0.0D0
  380       CONTINUE
C
  390       A(I,I) = 1.0D0
            G = RV1(I)
            L = I
  400    CONTINUE
C
         IF (M .GE. N .OR. IP .EQ. 0) GO TO 510
```

```
      M1 = M + 1
C
      DO 500 I = M1, N
C
         DO 500 J = 1, IP
         B(I,J) = 0.0D0
  500 CONTINUE
C     :::::::::: DIAGONALIZATION OF THE BIDIAGONAL FORM ::::::::::
  510 EPS = MACHEP * X
C     :::::::::: FOR K=N STEP -1 UNTIL 1 DO -- ::::::::::
      DO 700 KK = 1, N
         K1 = N - KK
         K = K1 + 1
      ITS = 0
C     :::::::::: TEST FOR SPLITTING.
C                FOR L=K STEP -1 UNTIL 1 DO -- ::::::::::
  520    DO 530 LL = 1, K
            L1 = K - LL
            L = L1 + 1
            IF (DABS(RV1(L)) .LE. EPS) GO TO 565
C     :::::::::: RV1(1) IS ALWAYS ZERO, SO THERE IS NO EXIT
C                THROUGH THE BOTTOM OF THE LOOP ::::::::::
            IF (DABS(W(L1)) .LE. EPS) GO TO 540
  530    CONTINUE
C     :::::::::: CANCELLATION OF RV1(L) IF L GREATER THAN 1 ::::::::::
  540    C = 0.0D0
         S = 1.0D0
C
         DO 560 I = L, K
            F = S * RV1(I)
            RV1(I) = C * RV1(I)
            IF (DABS(F) .LE. EPS) GO TO 565
            G = W(I)
            H = DSQRT(F*F+G*G)
            W(I) = H
            C = G / H
            S = -F / H
            IF (IP .EQ. 0) GO TO 560
C
            DO 550 J = 1, IP
               Y = B(L1,J)
               Z = B(I,J)
               B(L1,J) = Y * C + Z * S
               B(I,J) = -Y * S + Z * C
  550       CONTINUE
C
  560    CONTINUE
C     :::::::::: TEST FOR CONVERGENCE ::::::::::
  565    Z = W(K)
         IF (L .EQ. K) GO TO 650
C     :::::::::: SHIFT FROM BOTTOM 2 BY 2 MINOR ::::::::::
      IF (ITS .EQ. 30) GO TO 1000
      ITS = ITS + 1
         X = W(L)
         Y = W(K1)
```

```
            G = RV1(K1)
            H = RV1(K)
            F = ((Y - Z) * (Y + Z) + (G - H) * (G + H)) / (2.0D0 * H * Y)
            G = DSQRT(F*F+1.0D0)
            F = ((X - Z) * (X + Z) + H * (Y / (F + DSIGN(G,F)) - H)) / X
C       ::::::::::: NEXT QR TRANSFORMATION :::::::::::
            C = 1.0D0
            S = 1.0D0
C
            DO 600 I1 = L, K1
               I = I1 + 1
               G = RV1(I)
               Y = W(I)
               H = S * G
               G = C * G
               Z = DSQRT(F*F+H*H)
               RV1(I1) = Z
               C = F / Z
               S = H / Z
               F = X * C + G * S
               G = -X * S + G * C
               H = Y * S
               Y = Y * C
C
               DO 570 J = 1, N
                  X = A(J,I1)
                  Z = A(J,I)
                  A(J,I1) = X * C + Z * S
                  A(J,I) = -X * S + Z * C
  570          CONTINUE
C
               Z = DSQRT(F*F+H*H)
               W(I1) = Z
C       ::::::::::: ROTATION CAN BE ARBITRARY IF Z IS ZERO :::::::::::
               IF (Z .EQ. 0.0D0) GO TO 580
               C = F / Z
               S = H / Z
  580          F = C * G + S * Y
               X = -S * G + C * Y
               IF (IP .EQ. 0) GO TO 600
C
               DO 590 J = 1, IP
                  Y = B(I1,J)
                  Z = B(I,J)
                  B(I1,J) = Y * C + Z * S
                  B(I,J) = -Y * S + Z * C
  590          CONTINUE
C
  600       CONTINUE
C
            RV1(L) = 0.0D0
            RV1(K) = F
            W(K) = X
            GO TO 520
C       ::::::::::: CONVERGENCE :::::::::::
```

```
  650     IF (Z .GE. 0.0D0) GO TO 700
C      :::::::::: W(K) IS MADE NON-NEGATIVE ::::::::::
         W(K) = -Z
C
         DO 690 J = 1, N
  690    A(J,K) = -A(J,K)
C
  700 CONTINUE
      IF (.NOT. RETX) GO TO 1001
C     :::::::::: FIND MAXIMUM ELEMENT OF W ::::::::::
      Z = 0.0D0
      DO 750 J = 1, N
         X = W(J)
         IF (X .LE. Z) GO TO 750
         Z = X
  750 CONTINUE
      IF (Z .EQ. 0) GO TO 999
C     :::::::::: FORM PSEUDO INVERSE OF DIAG(W) ::::::::::
      DO 800 J = 1, N
         X = W(J) / Z
         IF (X .LE. RKTOL) GO TO 790
         W(J) = 1.0D0 / W(J)
         GO TO 800
  790    W(J) = 0.0D0
  800 CONTINUE
C     :::::::::: FORM X (RETURNED IN B) ::::::::::
      DO 900 J = 1, IP
C
         DO 810 I = 1, N
            RV1(I) = W(I) * B(I,J)
  810    CONTINUE
C
         DO 890 I = 1, N
C
            X = 0.0D0
            DO 850 I1 = 1, N
               X = X + A(I,I1) * RV1(I1)
  850       CONTINUE
C
            B(I,J) = X
C
  890    CONTINUE
C
  900 CONTINUE
C
      GO TO 1001
C     :::::::::: ERROR IF MAX SINGULAR VALUE = 0 ::::::::::
  999 K=-1
C     :::::::::: SET ERROR -- NO CONVERGENCE TO A
C                SINGULAR VALUE AFTER 30 ITERATIONS ::::::::::::
 1000 IERR = K
 1001 RETURN
C     :::::::::: LAST CARD OF MINFIT ::::::::::
      END
```

APPENDIX B: TROLL Implementation of MINFIT and Associated Output

The calling sequence for using the singular value decomposition within the TROLL environment is considerably different than that for the Fortran subroutine listed in Appendix A. This is a consequence of the basic design features of TROLL. However all computations are actually performed by the routine listed in Appendix A.

The TROLL version of the singular value decomposition is a function named MINFIT. Since it is a function, it returns a single data file as its result, and by TROLL convention it may not modify any of its arguments. The format of the TROLL call to MINFIT is

result = MINFIT (*A-matrix* <, *B-matrix* <, *code* >>)

where the <> indicate optional arguments.

Since we may desire several matrices as output from MINFIT, the data file returned as *result* may be made up of several matrices. The precise result returned by MINFIT is controlled by the *code* parameter as described in the following table for the linear system:

$$\underset{mxn}{A}\ \ \underset{nxp}{X} = B \qquad \text{where } A = U\Sigma V^T \text{ and } W = \text{diagonal of } \Sigma$$

Code	*B-matrix* omitted	*B-matrix* present
0	illegal	X (nxp) (default)
1	V (nxn) (default)	V (nxn)
2	W (nx1)	W (nx1)
3	illegal	U^TB (nxp)
4	$\begin{bmatrix} W \\ V \end{bmatrix}$ $\begin{pmatrix} 1xn \\ nxn \end{pmatrix}$	$\begin{bmatrix} W \\ V \\ (U^TB)^T \end{bmatrix}$ $\begin{pmatrix} 1xn \\ nxn \\ pxn \end{pmatrix}$

The correspondence between the TROLL parameters and the Fortran parameters is as follows:

Immediately prior to TROLL call to Fortran routine

TROLL	Fortran parameter
Max (number of rows of *A-matrix*, number of columns *A-matrix*)	NM
Number of rows of *A-matrix*	M
Number of columns of *A-matrix*	N
A-matrix	A
free storage	W
if *B-matrix* omitted then 0 else number of columns of *B-matrix*	IP
not set	IERR
free storage	RV1
if *code* = 0 or *code* omitted and *B-matrix* is present then .TRUE. else .FALSE.	RETX

After call of Fortran routine

If IERR is not zero then print appropriate error message, otherwise

Code	Fortran variable to be used as result
0	B (the solution X is formed in B)
1	A
2	W
3	B
4	$\begin{bmatrix} W \\ A \end{bmatrix}$ lxn / nxm or if *B-matrix* was specific $\begin{bmatrix} W \\ A \\ B \end{bmatrix}$ lxn / nxn / nxp

For more details on the use of the TROLL function, see [7].

The following output is the result of performing the TROLL version of MINFIT on the Longley data described in Appendix C. Row 1 of the matrix contains W, rows 2 through 8 contain the V matrix, and row 9 is $(U^T b)^T$.

```
MINFIT(LONGLEY_X,LONGLEY_Y,4)
```

ROW	COLUMN 1	COLUMN 2	COLUMN 3	COLUMN 4	COLUMN 5	COLUMN 6	COLUMN 7
1	1.6637E+06	8.3900E+04	3.4057E+03	1.5848E+03	41.654200	3.4323E-04	3.6504E+00
2	-2.3417E-06	1.0798E-05	-9.7316E-06	2.8160E-06	-5.1038E-04	-1.0000E+00	3.4181E-05
3	-2.4376E-04	6.1669E-04	-4.9654E-04	-1.4370E-03	-1.0442E-01	1.9307E-05	-9.9453E-01
4	-9.6034E-01	-2.7878E-01	-2.2179E-03	4.6324E-03	-1.3505E-03	-3.0688E-08	1.9871E-04
5	-7.7734E-03	1.0335E-02	7.8544E-01	-6.1856E-01	-1.7084E-02	-4.5778E-07	2.3034E-03
6	-6.2675E-03	1.3422E-02	-6.1866E-01	-7.8548E-01	8.0722E-03	-1.3094E-07	6.0617E-04
7	-2.7861E-01	9.5999E-01	-4.8046E-05	1.8883E-02	2.1381E-02	1.0430E-07	-1.6086E-03
8	-4.5794E-03	2.0892E-02	-1.8467E-02	4.8026E-03	-9.9412E-01	5.1138E-04	1.0439E-01
9	-2.5750E+05	4.6093E+04	-2.8832E+03	1.6049E+03	-1.7736E+03	1.1890E+03	210.980000

APPENDIX C: Selected Matrices, Computed Solutions, and Illustrative Examples

This appendix displays a representative sample of matrices on which the subroutine MINFIT has performed satisfactorily. The input matrices and the output computations have been retained on magnetic tape. The format of the printing was chosen for convenience and does not include the full fifteen decimal place output that was produced by the long precision computation on the machine. If anyone should attempt to reproduce these results on a machine whose arithmetic or relative precision is different from that of the IBM 360/67 he may get output that is different from that which we display. However, such results should be correct to the order of machine precision on which the computation is performed.

Though we include certain matrices of the Hilbert segments, we do not encourage their use as test matrices for software validation. The Hilbert segments are not representable exactly in a computing machine unless appropriate multipliers are used to preclude a perturbation on input of the data. We have used such multipliers.

Other matrices exhibited are a 3x3 matrix that is contrived to display information about near dependencies of rows or columns, a test matrix from [1]* and [2] and a matrix suggested by Ed Kuh. The matrix from [1] is exactly representable in the machine though it is ill conditioned with respect to the solution of linear systems of equations. The matrix in [3] shows the dependence of the solution vector x on the rank tolerance that is chosen.

On the output that is displayed, V has its usual meaning, W contains the unordered singular values from MINFIT, P is an integer vector that indicates the descending order of the singular values, MU contains μ_i for i=1,2,...,n for each right-hand side and C contains $U^T b$. X contains the candidate solution of Ax=b. IERR is the error indicator from MINFIT; it is non-zero if the computation of any singular value requires more than 30 iterations or if the maximum singular value is zero.

*Numerals in square brackets refer to entries in the Reference section, p. C14.

This 3x3 matrix shows output that indicates rank 2 if the smallest singular value is treated as zero. Given this interpretation, columns 1 and 2 are linearly dependent. This information is contained in column 2 of the V matrix.

```
 A=
( ROW    1 ):
 0.10101000 01  0.10098000 01  0.98000000 00
( ROW    2 ):
 0.10098000 01  0.10104000 01  0.98000000 00
( ROW    3 ):
 0.98000000 00  0.98000000 00  0.10100000 01


 R=
(COLUMN   1)
 0.10000000 01  0.0             0.0
(COLUMN   2)
 0.0            0.10000000 01  0.0
(COLUMN   3)
 0.0            0.0             0.10000000 01

IERR =    0

 V =
(COLUMN   1)
-0.57927490 00 -0.57933300 00 -0.57342300 00
(COLUMN   2)
-0.70801190 00  0.70619830 00  0.17604610-02
(COLUMN   3)
-0.40393050 00 -0.40701010 00  0.81925760 00


 W=
 0.29901010 01  0.44980760-03  0.39948830-01


 C=
(COLUMN   1)
-0.57927490 00 -0.70801190 00 -0.40393050 00
(COLUMN   2)
-0.57933300 00  0.70619830 00 -0.40701010 00
(COLUMN   3)
-0.57342300 00  0.17604610-02  0.81925760 00


 P =     1    3    2


 MU=
(COLUMN   1)
 0.46150140 02  0.88407600 00  0.56800110-02
(COLUMN   2)
 0.46145510 02  0.87738910 00  0.56945950-02
(COLUMN   3)
 0.46621110 02  0.43594980 00  0.42822180 00


USING MACHEP, X=
(COLUMN   1)
 0.11186300 04 -0.11073520 04 -0.10943610 02
(COLUMN   2)
-0.11073520 04  0.11129910 04 -0.54718030 01
(COLUMN   3)
-0.10943610 02 -0.54718030 01  0.16917920 02


 USING RKFUL, X=
(COLUMN   1)
 0.11186300 04 -0.11073520 04 -0.10943610 02
(COLUMN   2)
-0.11073520 04  0.11129910 04 -0.54718030 01
(COLUMN   3)
-0.10943610 02 -0.54718030 01  0.16917920 02
```

The matrix whose data is displayed on the following page was suggested by Ed Kuh. The matrix is 32x10 and has singular values, to 4 decimal places,

4921, 41.89, 30.33, 18.71, 8.573, 2.491,

4.763, 5.532, 6.162, 6.091.

The indicated rank determination is that the matrix is of rank 10 if the data is certain in all digits, of rank 1 if the third digit is doubtful.

The residual checks for the decomposition are

```
MAX-ROW-SUM RESIDUAL =     0.18182413270-14
EUCLIDIAN RESIDUAL =     0.24026975930-14
MAX-COL-SUM RESIDUAL =     0.13780222750-14
```

Truncation of the data to integers 234,231,...,311 gives singular values to 4 decimal places.

4911, 41.10, 30.07, 18.59, 8.356, 3.403,

6.299, 5.727, 4.963, 5.198

Data for A

346.6 — Row 32
342.1 — Row 31
337.9 — Row 30
337.9
331.2
326.7
321.8
314.5
312.2
311.7
311.6
307.4
303.8
300.8
294.6
290.7
286.4
283.2
278.9
272.6
266.2
262.4
257.3
254.7
255.3
254.0
253.8
253.4
249.2
245.8
240.9
234.4 — Row 1
231.7
231.2
227.9
226.0
220.8
214.7
209.0
201.5
202.2

Right-hand side B

214.6
216.7
225.0
228.4
230.1
231.0
230.3
232.3
234.6
237.3
241.8
247.7
252.7
256.8
260.4
262.0
264.4
267.5
272.8
277.2
279.3
283.8
285.4
284.5
287.4
292.2
296.2
304.0
309.8
314.8
316.3
321.1

The Hilbert matrix of order 7, generated in long precision, 7 digits of which are given for each element, is inexact in the machine.

```
 A=
( ROW     1 ):
0.1000000D 01  0.5000000D 00  0.3333333D 00  0.2500000D 00  0.2000000D 00  0.1666667D 00  0.1428571D 00
( ROW     2 ):
0.5000000D 00  0.3333333D 00  0.2500000D 00  0.2000000D 00  0.1666667D 00  0.1428571D 00  0.1250000D 00
( ROW     3 ):
0.3333333D 00  0.2500000D 00  0.2000000D 00  0.1666667D 00  0.1428571D 00  0.1250000D 00  0.1111111D 00
( ROW     4 ):
0.2500000D 00  0.2000000D 00  0.1666667D 00  0.1428571D 00  0.1250000D 00  0.1111111D 00  0.1000000D 00
( ROW     5 ):
0.2000000D 00  0.1666667D 00  0.1428571D 00  0.1250000D 00  0.1111111D 00  0.1000000D 00  0.9090909D-01
( ROW     6 ):
0.1666667D 00  0.1428571D 00  0.1250000D 00  0.1111111D 00  0.1000000D 00  0.9090909D-01  0.8333333D-01
( ROW     7 ):
0.1428571D 00  0.1250000D 00  0.1111111D 00  0.1000000D 00  0.9090909D-01  0.8333333D-01  0.7692308D-01
```

Its singular values are

```
W=
0.1660885D 01  0.2719202D 00  0.2128975D-01  0.1008588D-02  0.2938637D-04  0.4856763D-06  0.3493744D-08
```

Multiplication of the Hilbert matrix of order 7 by the constant 360360 allows a machine representation that is exact.

```
A=
( ROW     1 ):
0.3603600D 06  0.1801800D 06  0.1201200D 06  0.9009000D 05  0.7207200D 05  0.6006000D 05  0.5148000D 05
( ROW     2 ):
0.1801800D 06  0.1201200D 06  0.9009000D 05  0.7207200D 05  0.6006000D 05  0.5148000D 05  0.4504500D 05
( ROW     3 ):
0.1201200D 06  0.9009000D 05  0.7207200D 05  0.6006000D 05  0.5148000D 05  0.4504500D 05  0.4004000D 05
( ROW     4 ):
0.9009000D 05  0.7207200D 05  0.6006000D 05  0.5148000D 05  0.4504500D 05  0.4004000D 05  0.3603600D 05
( ROW     5 ):
0.7207200D 05  0.6006000D 05  0.5148000D 05  0.4504500D 05  0.4004000D 05  0.3603600D 05  0.3276000D 05
( ROW     6 ):
0.6006000D 05  0.5148000D 05  0.4504500D 05  0.4004000D 05  0.3603600D 05  0.3276000D 05  0.3003000D 05
( ROW     7 ):
0.5148000D 05  0.4504500D 05  0.4004000D 05  0.3603600D 05  0.3276000D 05  0.3003000D 05  0.2772000D 05
```

Its singular values are

```
W=
0.5985166D 06  0.9798916D 05  0.7671976D 04  0.3634546D 03  0.1058967D 02  0.1750183D 00  0.1259061D-02
```

The Longley data matrix [3] with its associated output is

```
 A=
( ROW    1 ):
0.10000000D 01  0.83000000 02  0.2342890D 06  0.23560000 04  0.15900000 04  0.1076080D 06  0.19470000 04
( ROW    2 ):
0.10000000 01  0.88500000 02  0.2594260D 06  0.23250000 04  0.14560000 04  0.1086320D 06  0.19480000 04
( ROW    3 ):
0.10000000 01  0.88200000 02  0.2580540D 06  0.36820000 04  0.16160000 04  0.1097730D 06  0.19490000 04
( ROW    4 ):
0.10000000 01  0.89500000 02  0.2845990D 06  0.33510000 04  0.16500000 04  0.1109290D 06  0.19500000 04
( ROW    5 ):
0.10000000 01  0.96200000 02  0.3289750D 06  0.20990000 04  0.30990000 04  0.1120750D 06  0.19510000 04
( ROW    6 ):
0.10000000 01  0.98100000 02  0.3469990D 06  0.19320000 04  0.35940000 04  0.1132700D 06  0.19520000 04
( ROW    7 ):
0.10000000 01  0.99000000 02  0.3653850D 06  0.18700000 04  0.35470000 04  0.1150940D 06  0.19530000 04
( ROW    8 ):
0.10000000 01  0.10000000 03  0.3631120D 06  0.35780000 04  0.33500000 04  0.1162190D 06  0.19540000 04
( ROW    9 ):
0.10000000 01  0.10120000 03  0.3974690D 06  0.29040000 04  0.30480000 04  0.1173880D 06  0.19550000 04
( ROW   10 ):
0.10000000 01  0.10460000 03  0.4191800D 06  0.28220000 04  0.28570000 04  0.1187340D 06  0.19560000 04
( ROW   11 ):
0.10000000 01  0.10840000 03  0.4427690D 06  0.29360000 04  0.27980000 04  0.1204450D 06  0.19570000 04
( ROW   12 ):
0.10000000 01  0.11080000 03  0.4445460D 06  0.46810000 04  0.26370000 04  0.1219500D 06  0.19580000 04
( ROW   13 ):
0.10000000 01  0.11260000 03  0.4827040D 06  0.38130000 04  0.25520000 04  0.1233660D 06  0.19590000 04
( ROW   14 ):
0.10000000 01  0.11420000 03  0.5026010D 06  0.39130000 04  0.25140000 04  0.1253680D 06  0.19600000 04
( ROW   15 ):
0.10000000 01  0.11570000 03  0.5181730D 06  0.48060000 04  0.25720000 04  0.1278520D 06  0.19610000 04
( ROW   16 ):
0.10000000 01  0.11690000 03  0.5548940D 06  0.40070000 04  0.28270000 04  0.1300810D 06  0.19620000 04


 B=
(COLUMN   1)
 0.60323000 05  0.61122000 05  0.60171000 05  0.61187000 05  0.63221000 05  0.63639000 05  0.64989000 05
 0.63761000 05  0.66019000 05  0.67857000 05  0.68169000 05  0.66513000 05  0.68655000 05  0.69564000 05
 0.69331000 05  0.70551000 05

IERR =    0

 V =
(COLUMN   1)
-0.23417280-05 -0.24375680-03 -0.9603401D 00 -0.7773377D-02 -0.6267548D-02 -0.2786148D 00 -0.4579409D-02
(COLUMN   2)
 0.10797810-04  0.61668600-03 -0.2787807D 00  0.1033477D-01  0.1342164D-01  0.9599779D 00  0.2089197D-01
(COLUMN   3)
-0.97316110-05 -0.49653610-03 -0.2217931D-02  0.7854394D 00 -0.6186589D 00 -0.4804589D-04 -0.1846682D-01
(COLUMN   4)
 0.28160440-05 -0.1437022D-02  0.4632393D-02 -0.6185634D 00 -0.7854783D 00  0.1888282D-01  0.4802591D-02
(COLUMN   5)
-0.51038040-03 -0.1044185D 00 -0.1350474D-02 -0.1708369D-01  0.8072157D-02  0.2138102D-01 -0.9941230D 00
(COLUMN   6)
-0.99999990 00  0.1930712D-04 -0.3068798D-07 -0.4577772D-06 -0.1309433D-06  0.1042993D-06  0.5113788D-03
(COLUMN   7)
 0.3418098D-04 -0.9945321D 00  0.1987148D-03  0.2303608D-02  0.6061726D-03 -0.1608563D-02  0.1043920D 00


 W=
 0.16636680 07  0.83899620 05  0.3405674D 04  0.1584788D 04  0.4165420D 02  0.3432289D-03  0.3650380D 01


 C=
(COLUMN   1)
-0.15328510 00  0.45378580 00  0.89857260-01  0.2651757D 00  0.3055847D 00  0.1139390D-01  0.1530207D 00
-0.4846529D-01 -0.4020948D-02 -0.1943373D 00 -0.3430291D 00 -0.8976866D-01 -0.1198882D 00 -0.2986208D 00
-0.2974020D 00 -0.4662698D 00


 P =     1    2    3    4    5    7    6


 MU=
(COLUMN   1)
 0.19577650 20  0.3334570D 18  0.6697343D 17  0.1064331D 17  0.2458132D 15  0.4238646D 14  0.5433815D 11


 USING MACHEP, X=
 (COLUMN   1)
 -0.3464269D 07  0.1384952D 02 -0.3527839D-01 -0.2009419D 01 -0.1025133D 01 -0.5234782D-01  0.1819948D 04


 USING RKTOL, X=
 (COLUMN   1)
  0.23724110-01 -0.5303553D 02  0.7103303D-01 -0.4235556D 00 -0.5715101D 00 -0.4136687D 00  0.4839436D 02
```

The Bauer matrix with its associated output is

```
 A=
 ( ROW     1 ):
-0.74000000D 02  0.80000000D 02  0.18000000D 02 -0.11000000D 02 -0.40000000D 01 -0.80000000D 01
 ( ROW     2 ):
 0.14000000D 02 -0.69000000D 02  0.21000000D 02  0.28000000D 02  0.0             0.70000000D 01
 ( ROW     3 ):
 0.66000000D 02 -0.72000000D 02 -0.50000000D 01  0.70000000D 01  0.10000000D 01  0.40000000D 01
 ( ROW     4 ):
-0.12000000D 02  0.66000000D 02 -0.30000000D 02 -0.23000000D 02  0.30000000D 01 -0.30000000D 01
 ( ROW     5 ):
 0.30000000D 01  0.80000000D 01 -0.70000000D 01 -0.40000000D 01  0.10000000D 01  0.0
 ( ROW     6 ):
 0.40000000D 01 -0.12000000D 02  0.40000000D 01  0.40000000D 01  0.0             0.10000000D 01


 B=
(COLUMN    1)
 0.51000000D 02 -0.61000000D 02 -0.56000000D 02  0.69000000D 02  0.10000000D 02 -0.12000000D 02
(COLUMN    2)
-0.56000000D 02  0.52000000D 02  0.76400000D 03  0.40960000D 04 -0.13276000D 05  0.84210000D 04
(COLUMN    3)
-0.50000000D 01 -0.90000000D 01  0.70800000D 03  0.41650000D 04 -0.13266000D 05  0.84090000D 04


IERR =     0

 V =
(COLUMN    1)
 0.53159590D 00 -0.82429840D 00 -0.38242860-01  0.17949250 00  0.10576910-01  0.64390190-01
(COLUMN    2)
-0.62509500 00 -0.29815740 00  0.62845080 00  0.34816700 00 -0.63856900-01  0.10491370-01
(COLUMN    3)
 0.33696200 00  0.10421750 00  0.65658480 00 -0.52381900 00 -0.23501730 00 -0.33892800 00
(COLUMN    4)
-0.40824830 00 -0.40824830 00 -0.40824830 00 -0.40824830 00 -0.40824830 00 -0.40824830 00
(COLUMN    5)
 0.21539230 00  0.23251740 00 -0.70459190-01  0.60620830 00 -0.63896270 00 -0.34469610 00
(COLUMN    6)
 0.76299260-02 -0.64905330-02 -0.29192670-01  0.19498870 00  0.60467950 00 -0.77161490 00


 W=
 0.17383930 03  0.64861870 02  0.10667160 02  0.10000000 01  0.17524770 00  0.47441820-04


 C=
(COLUMN    1)
-0.11457290 03 -0.35669610 02 -0.79211710 01 -0.40824830 00  0.89820800 00 -0.85865440-04
(COLUMN    2)
 0.37040050-03  0.16174950-03 -0.31773030-01 -0.40824830 00 -0.19669080 01 -0.16264440 05
(COLUMN    3)
-0.11457260 03 -0.35669450 02 -0.79529440 01 -0.81649660 00 -0.10687000 01 -0.16264440 05


 P =     1     2     3     4     5     6


 MU=
(COLUMN    1)
 0.55143120 07  0.42339880 07  0.32020640 07  0.30130790 07  0.69023290 06  0.65274640 06
(COLUMN    2)
 0.23068100 15  0.14984950 15  0.16501570 12  0.12039250 10  0.43763960 08  0.14336980 01
(COLUMN    3)
 0.74578680 09  0.57262870 09  0.43232170 09  0.35115200 09  0.78558700 08  0.14337370 01


USING MACHEP, X=
(COLUMN    1)
 0.10000000 01  0.20000000 01 -0.10000000 01  0.30000000 01 -0.40000000 01 -0.12249380-09
(COLUMN    2)
-0.26157640 07  0.22251420 07  0.10008100 08 -0.66847850 08 -0.20730180 09  0.26453220 09
(COLUMN    3)
-0.26157630 07  0.22251440 07  0.10008100 08 -0.66847850 08 -0.20730180 09  0.26453220 09


 USING RKTOL, X=
(COLUMN    1)
 0.10000000 01  0.20000000 01 -0.10000000 01  0.30000000 01 -0.40000000 01 -0.12249380-09
(COLUMN    2)
-0.26157640 07  0.22251420 07  0.10008100 08 -0.66847850 08 -0.20730180 09  0.26453220 09
(COLUMN    3)
-0.26157630 07  0.22251440 07  0.10008100 08 -0.66847850 08 -0.20730180 09  0.26453220 09
```

The condition number of a nonsingular matrix may be improved by row or column scaling. The Bauer matrix, scaled as

```
 A=
( ROW     1 ):
-0.74000000D 02  0.80000000D 02  0.36000000D 02 -0.33000000D 02 -0.40000000D 02 -0.80000000D 02
( ROW     2 ):
 0.14000000D 02 -0.69000000D 02  0.42000000D 02  0.84000000D 02  0.0             0.70000000D 02
( ROW     3 ):
 0.66000000D 02 -0.72000000D 02 -0.10000000D 02  0.21000000D 02  0.10000000D 02  0.40000000D 02
( ROW     4 ):
-0.12000000D 02  0.66000000D 02 -0.60000000D 02 -0.69000000D 02  0.30000000D 02 -0.30000000D 02
( ROW     5 ):
 0.24000000D 02  0.64000000D 02 -0.11200000D 03 -0.96000000D 02  0.80000000D 02  0.0
( ROW     6 ):
 0.28000000D 02 -0.84000000D 02  0.56000000D 02  0.84000000D 02  0.0             0.70000000D 02
```

with singular values

```
W=
0.2959449D 03  0.1816570D 03  0.4893780D 02  0.1288217D 02  0.7095995D 00  0.1397107D-02
```

The singular value decomposition provides $U\Sigma V^T$ as the decomposition of a matrix A. Given the orthonormal columns U and V one can form another matrix $U\Sigma V^T$ for arbitrary Σ. Using U and V from the inexact Hilbert matrix of order 7, the reformed matrix

```
THE REFORMED A =
( ROW     1 ):
 0.20106490 02 -0.11191230 03  0.23250410 03 -0.24893840 03  0.14774330 03 -0.46373370 02  0.60402080 01
( ROW     2 ):
-0.11191230 03  0.13209940 04 -0.47280010 04  0.78498420 04 -0.67653190 04  0.29490710 04 -0.51555810 03
( ROW     3 ):
 0.23250410 03 -0.47280010 04  0.25364010 05 -0.58889480 05  0.67996620 05 -0.38558890 05  0.85824910 04
( ROW     4 ):
-0.24893840 03  0.78498420 04 -0.58889480 05  0.18005690 06 -0.26360290 06  0.18470440 06 -0.49868680 05
( ROW     5 ):
 0.14774330 03 -0.67653190 04  0.67996620 05 -0.26360290 06  0.47139410 06 -0.39302090 06  0.12386200 06
( ROW     6 ):
-0.46373370 02  0.29490710 04 -0.38558890 05  0.18470440 06 -0.39302090 06  0.37926500 06 -0.13550680 06
( ROW     7 ):
 0.60402080 01 -0.51555810 03  0.85824910 04 -0.49868680 05  0.12386200 06 -0.13550680 06  0.53689920 05
```

where the σ_i are 10^{-8}, 10^{-7}, 10^{-6}, 10^{-5}, 10^{-4}, 10^{-3} and 10^{-2}.

The computed σ_i from the reformed A are

```
W=
 0.10000000 07  0.10000000 06  0.10000000 05  0.10000000 04  0.10000000 03  0.10000000 01  0.10000000 02

MAX-ROW-SUM RESIDUAL =      0.12283894900-14
EUCLIDEAN RESIDUAL =      0.99904912580-15
MAX-COL-SUM RESIDUAL =      0.12283894900-14
```

However, choosing $\sigma_i = 10^{24}, 10^{20}, 10^{16}, 10^{12}, 10^{8}, 10^{4}, 10^{0}$ gives

```
W=
0.10000000D 25  0.10000000D 21  0.10000000D 17  0.26082340D 08  0.10000010D 13  0.21895020 08  0.35466650 07

MAX-ROW-SUM RESIDUAL =      0.60675626030-15
EUCLIDEAN RESIDUAL =      0.46976204570-15
MAX-COL-SUM RESIDUAL =      0.40450417350-15
```

The singular values smaller than 10^{12} are effected by the order of machine precision relative to σ_{max}.

Choosing $\sigma_i = 10^{0}, 10^{-4}, 10^{-8}, 10^{-12}, 10^{-16}, 10^{-20}, 10^{-24}$ gives

```
W=
0.10000000D 01  0.10000000D-03  0.10000000D-07  0.99999120D-12  0.11040820D-15  0.1852770D-16  0.25307140D-17

MAX-ROW-SUM RESIDUAL =      0.13160510700-14
EUCLIDEAN RESIDUAL =      0.88817841530-15
MAX-COL-SUM RESIDUAL =      0.26421021390-15
```

The order 100 matrix

$$\begin{bmatrix} .501 & -1 & & & \\ & .502 & -1 & & \\ & & \ddots & \ddots & \\ & & & \ddots & \\ & & & .600 & -1 \end{bmatrix}$$

has a maximum singular value ~1.587 and a minimum singular value $\sim 10^{-22}$. The minimum singular value computed on the IBM 360/67 is $.3329410 \times 10^{-15}$. Using long precision on the IBM 360/195 at Argonne National Laboratory, Jack Dongerra computed the same singular values as those from the 67 except for the minimum singular value which was $.3329272 \times 10^{-15}$. The arithmetic of the 195 is not the same as that of the 67. Multiplying this matrix by 10^3 (so that the input was internally representable as exact integers) gave the smallest singular value $.33294095 \times 10^{-12}$. Brian Smith suggested running this matrix on the 195 using short precision from which the smallest singular value was $.1287991 \times 10^{-5}$ and $.13423073 \times 10^{-2}$ for the matrix scaled by 10^3.

We have done some timing tests on the singular value decomposition. In general, accessing data is more costly than computing the singular value decomposition, so we would expect the use of Fortran H (opt=2) to reduce the computation times listed below by about 50%. From a Fortran IV G compilation on the 360/67 computer, the computation time for U, V, and Σ using SVD from [2] on random square matrices of dimension N is as follows:

N	Time in seconds
5	.074
10	.464
20	3.490
40	25.010
60	79.353
80	185.653

These times were obtained from the interval timer on the 67 which gives approximate microseconds at 13 microsecond intervals. These timings were obtained at the NBER Computer Research Center by Harry Bochner.

The time required by MINFIT is approximately that of SVD if U, V, and Σ are computed. However, in general, U is not needed. The time that is used to form V, Σ, and $U^T b$ is therefore reduced by almost 50% of the times listed here.

The time for computation of the singular value decomposition will be matrix dependent in that fewer iterations may be required when there are multiplicities or clusters of singular values.

References

1. Bauer, F. L., "Elimination with Weighted Row Combinations for Solving Linear Equations and Least Squares Problems," in J. H. Wilkinson and C. Reinsch (eds.) Handbook for Automatic Computation, Volume II: Linear Algebra, Springer Verlag, 7, 338-352 (1965).

2. Golub, G. H. and Reinsch, C., "Singular Value Decomposition and Least Squares Solutions," in J. H. Wilkinson and C. Reinsch (eds.) Handbook for Automatic Computation, Volume II: Linear Algebra, Springer Verlag, 134-151 (1971); prepublished in Numer. Math. 14, 403-420 (1970).

3. Longley, James W., "An Appraisal of Least Squares Programs for the Electronic Computer from the Point of View of the User," JASA 62, 819-841, 1967.