Annals of Economic and Social Measurement, 2/3, 1973

A NOTE ON MATRIX FACTORIZATION*

VIRIGINIA KLEMA

Over the last twenty-five years numerical algebraists have investigated the numerically stable solution of certain problems of digital computing. Such problems often had their origin in the physical sciences, but developments over the last five years in computing hardware and econometrics-oriented software systems now make it appropriate to relate particular algorithms in the literature of numerical algebra to the analysis of data for large econometric models. The cross-fertilization of disciplines at the NBER's Computer Research Center facilitates this application of numerical analysis research to the field of econometrics.

This note, expository in nature, is intended to draw attention to papers which describe—and programmed procedures which implement—stable algorithms for solving linear systems of equations, least squares problems, and pseudo-inverses. Implicit in the solution of nonlinear systems of equations is the need for the information provided by a matrix inverse or pseudo-inverse (though the inverse or pseudo-inverse need not be computed explicitly). The choice of matrix-factorization method is important because it governs numerical stability, computer time, and storage requirements.

Methods of factorization are correlated, in Table 1, with the properties of matrices to which they apply. Papers describing the algorithmic details are cited in the following discussion and are listed in the bibliography. Attention is focused on direct methods for computing x of $Ax = B$, where a direct method defines x in a finite number of arithmetic operations and, at times, square roots. If the structure of a matrix of coefficients guarantees convergence of iterative methods such as Jacobi, Gauss–Seidel, SOR, or ADI, these methods can be used. Such techniques are not included here though their viability in connection with partial differential equations is well known.

The classes of matrices considered are those whose dimensions and rank are as follows:

(1) $n \times n$, rank n;
(2) $m \times n$, rank min (m, n);
(3) $m \times n$, rank r, $r < \min(m, n)$.

Each is considered first for the case in which primary storage is sufficient for the factorization, and second for the case in which secondary storage must be used. The case in which the rank must be determined during computation is also considered.

Since the explicit inverse or pseudo-inverse is, in general, not required, the problem is to find a factorization of A that gives triangular matrices formed by

* This work was supported by National Science Foundation Grant GJ-1154x to the National Bureau of Economic Research, Inc. The author wishes to thank Brian Smith of the Argonne National Laboratory for valuable discussions on singular value analysis; Edwin Kuh for suggesting the inclusion of Table 1; and John Kirsch for editorial assistance.

TABLE 1

PROPERTIES OF MATRICES AND METHODS OF FACTORIZATION

Methods of Factorization \ Properties of matrices	Symmetric positive definite	Symmetric indefinite	Non-symmetric, full rank $n \times n$	$m \times n$, rank min (m, n)	$m \times n$ rank $k <$ min (m, n)	symmetric high order*	non-symmetric high order*	$m \times n$ high order* rank min (m, n)	$m \times n$ high order* rank $k <$ min (m, n)
Choleski $A = LL^T$ or $A = LDL^T$	X								
Bunch $A = LDL^T$, D block diagonal		X							
LU with partial pivoting			X						
Householder transformations QU, $Q^TQ = I$				X					
Givens matrices $QU \cdot Q^TQ = I$						X	X	X	
SVD, $A = U\Sigma V^T$ Householder transformations					X				
SVD, $A = U\Sigma V^T$ Givens matrices									X

* High order implies matrix dimensions are such that the array cannot be held in core.

elementary or unitary transformations on A. The result of the factorization can usually overwrite A.

Given Class (1), where A is real, non-symmetric,

$$Ax = b,$$

the simplest factorization is based on elementary transformations giving

$$Ax = LUx = b$$

from which

$$Ly = b$$

$$Ux = y$$

where L and U are lower and upper triangular matrices, respectively, with L being unit-triangular (i.e., L has 1's on the diagonal). The elementary transformations usually require partial pivoting such that appropriate row or column interchanges are made to ensure numerical stability in the sense that multipliers are small throughout the calculation. Efficient ALGOL, FORTRAN, and PL/1 implementations on this algorithm are given in [5], and the posteriori error analysis is given in [17], [25], and [26]. Necessary equilibration is discussed in [5]. The LU decomposition is used for the case in which A is real non-symmetric; whereas A,

symmetric positive definite, is decomposed by Choleski factorization

$$A = LL^T \quad \text{or} \quad A = LDL^T$$

where L is lower triangular and L^T is the transpose of L. The factorization LDL^T permits L to have unit diagonal; D is a diagonal matrix; and the computation of square roots is avoided. See [20] for an ALGOL implementation and [26] for the associated error analysis. Pivoting is not necessary for the positive-definite symmetric matrix.

The LU decomposition requires $1/3\ n^3$ multiplicative operations; Choleski factorization, $1/6\ n^3$; and a solution x for each right-hand-side b, n^2. Note that the computation of A^I, where A^I is the inverse of A, requires n^3 multiplications; and computation of $A^I b$ would require n^2 additional multiplications and would be less precise than the solution obtained from the triangular decomposition.

Symmetric indefinite systems are treated in [1].

Class (2) represents the linear least squares problem, i.e., the solution of the normal equations

$$A^T Ax = A^T b$$

where A^T is the transpose of A.

One seeks x, the solution of an overdetermined system of equations, such that

$$\|b - Ax\| = \min$$

where $\|b - Ax\|$ is the euclidean norm. The euclidean norm is unitarily invariant, and we choose a factorization

$$A = QU$$

where A is $m \times n$, $m > n$; $Q^T Q = I$; and U is upper-triangular of dimension $n \times n$, rank n. The linear least squares solution is

$$x = U^I Q^T b.$$

The condition of A is represented in U; and working with $A^T A$, whose condition number is the square of that of A, is avoided. The upper triangular matrix can be inverted easily whereas $A^T A$ cannot.

Given computation with exact arithmetic, the orthogonal factorization can be obtained by the classical Gram-Schmidt, the modified Gram-Schmidt, the Householder, or the Givens transformations. However, by the classical Gram-Schmidt algorithm, using finite precision arithmetic, the columns of Q can depart from orthogonality so severely that reorthogonalization is not effective (see [25] and [26]). Therefore, the three other methods are better. Published ALGOL procedures using Householder transformations are given in [3], and FORTRAN subroutines are in the subroutine library of the Applied Mathematics Division, Argonne National Laboratory. Lucid expositions of this problem are published in [10] and [22], and certain FORTRAN subroutines are compared in [24]. A PL/1 implementation of [3] was written at the NBER Computer Research Center by Harry Bochner.

The unitary factorization requires twice the number of operations needed for the factorization by elementary transformations. However, the extra work is

compensated by the property that $Q^TQ = I$, and by the fact that norms are preserved. In many cases, the addition or deletion of rows or columns of A (i.e., the factorization of A) is important. Updating methods are described in [4], [7], and [13].

In [22] Wilkinson wrote, "It is well known that the problem of determining the rank of a matrix is far from trivial when rounding errors are involved, as invariably they are on a digital computer." The least squares problem has a unique solution when the $m \times n$ matrix A has rank n. However, if the rank r is less than n, a family of solutions provides a unique solution vector x of minimal euclidean norm. To determine this solution, it is essential that r be defined during computation. Further, r must be defined explicitly for computations involving the pseudo-inverse. The most effective way to determine r is by singular value analysis. Singular value decomposition has not been widely used, though this decomposition is needed to determine the condition number of the matrix.

Class (3) is dealt with in [11] and [12] by forming

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

where A is $m \times n$, $m \geq n$; $U^TU = V^TV = VV^T = I$; and Σ is diagonal in which $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_n \geq 0$. U contains n eigenvectors associated with the n largest eigenvalues of AA^T; and V contains the eigenvectors of A^TA. The $\sigma_i (i = 1, 2, \ldots, n)$ are the nonnegative square roots of the eigenvalues of A^TA and are called the singular values of A. If the rank of A is r, each element $\sigma_{r+1} \ldots \sigma_n = 0$ to within working accuracy of the machine. The condition number of a matrix A with respect to the solution of the linear system $Ax = b$ is defined to be the ratio

$$\frac{\sigma_{\max}}{\sigma_{\min}} = K(A)$$

where $\sigma_{\max}$ is the largest singular value of A, and $\sigma_{\min}$ is the smallest singular value. The condition number of the symmetric positive definite matrix is given by the ratio of the largest to the smallest eigenvalues of the matrix.

Singular value decomposition frequently reveals singular values that are small relative to some norm of the matrix but are not exactly zero. The columns of U and V are orthonomalized, and the transformations to obtain U, Σ, and V are norm-preserving; it follows that

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

Therefore, the user—or more appropriately, the computer programmer—can determine the reliable tolerance by which r is fixed. The deflation of rank causes the data to be perturbed by σ_{r+1}. For example, if $\sigma_{r+1} = 10^{-6}$, the deflation changes the data in the sixth decimal place but not in the first five.

The authors of [10], [12], [14], and [19] analyze, and make good suggestions for, the use of singular value decomposition for solving linear and nonlinear systems of equations and linear least squares problems.

Singular value decomposition is a powerful tool of numerical analysis. Much information is obtained, but $2mn^2 + 4n^3$ operations are required. Updating this decomposition requires operations of the order n^3 and is described in [2].

The ALGOL procedures listed in [11]—and FORTRAN subroutines based on these procedures and written at the Applied Mathematics Division, Argonne National Laboratory—compute the singular value decomposition and the solution of minimal norm.

Large computing installations have, or can obtain, efficiently coded programs for computing these factorizations for matrices whose dimensions are sufficiently small for the matrix A, and the right-hand-side b's, to be held in core. On a paged machine modifications may be necessary; [21] suggests such a modification for LU decomposition.

To deal with high order matrices (on a non-paged machine) that require use of secondary storage, it is necessary to use factorizations that minimize the number of accesses to the secondary storage device. In particular, either rows or columns, but not both, are required for a given stage of the computation. This requirement is not satisfied by elementary transformations with partial pivoting or by the Householder transformations for unitary factorizations.

The Householder factorization represents a sequence of rotations in the plane and is typically formed in the following way. The matrix A is successively transformed by matrices $P^{(k)}$, $k = 1, 2, \ldots, n$ such that

$$A^{(k+1)} = P^{(k)}A^{(k)}$$

in which $a_{ik}^{(k)}$ is annihilated for $i = k + 1, \ldots, n$. The generation of the matrices $P^{(k)}$ is described in [3], and requires the square root of the sum of squares of the elements $a_{ik}^{(k)}$ for $i = k, \ldots, n$—that is, the elements in the kth column of $A^{(k)}$. The matrix $P^{(k)}$ is usually not computed explicitly, but its transformation demands access to the columns of A; processing by groups of rows or columns is awkward unless A can be contained completely in core. All remaining columns of A must be processed in stage k before stage $k + 1$ can begin. On Householder factorization, see further [3], [16], and [17].

An alternative unitary factorization is presented by Givens in [8] and [9]. This method creates the necessary zero elements but permits processing by one or more rows at a time. The initial implementation of the Givens rotations requires $4/3\, n^3$ operations, twice the number required by Householder transformations. A recent modification of the Givens matrices by Morvin Gentleman [6] shows that the Givens transformations can be obtained in only $2/3\, n^3$ operations. The Givens transformations show a marked increase in efficiency when A is of high order; in particular, they take advantage of any sparsity in A. An early implementation of the Givens rotations for solving linear systems is described in [8]. This should now incorporate Gentleman's modifications.

Whenever a matrix can be partitioned into submatrices, the problem is much more tractable. Such partitioning, related to particular structures, has been used in [18] for the Leontief model and in [15] for linear programming. Steward's algorithm [23] is used to expose a block recursive structure for economic modeling problems.

National Bureau of Economic Research
Computer Research Center for Economics and Management Science

References

[1] Bunch, J. R. and Parlett, B. N., "Direct Methods for Solving Symmetric Indefinite Systems of Linear Equations," J. SIAM Numer. Anal. vol. 8, no. 4, 639–655 (1971).

[2] Businger, P. A., "Updating a Singular Value Decomposition," BIT, vol. 10, no. 3, 376–385 (1970).

[3] Businger, P. and Golub, G. H., "Linear Least Squares Solutions by Householder Transformations," in J. H. Wilkinson and C. Reinsch (eds.) *Handbook for Automatic Computation, Volume II: Linear Algebra*, Springer Verlag, 111–118 (1971).

[4] Chambers, John M., "Regression Updating," JASA, vol. 66, no. 326, 744–748 (1971).

[5] Forsythe, G., and Moler, C., *Computer Solution of Linear Algebraic Equations*, 27–79, 44–45, 63, Prentice-Hall (1967).

[6] Gentleman, W. Morvin, "Least Squares Computations by Givens Transformations without Square Roots," CSRR-2062, Dept. of Applied Analysis and Computer Science, University of Waterloo (1972).

[7] Gill, P. E., Golub, G. H., Murray, W., Saunders, M. A., "Methods for Modifying Matrix Factorizations," STAN-CS-72-322, Computer Science Department, Stanford University (1972).

[8] Givens, W., "The Linear Equations Problem," Technical Report No. 3, Applied Mathematics and Statistics Laboratories, Stanford University (1959).

[9] Givens, W., "Numerical Computation of the Characteristic Values of a Real Symmetric Matrix," ORNL 1574, Oak Ridge National Laboratory (1954).

[10] Golub, G. H., "Matrix Decomposition and Statistical Calculations" in R. C. Milton and J. A. Nelder (eds.), *Statistical Computation*, Academic Press 365–397 (1969).

[11] 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).

[12] Golub, G. H., and Kahan, W., "Calculating the Singular Values and Pseudo-inverse of a Matrix," J. SIAM Numer. Anal. ser. B., vol. 2, no. 2, 205–224 (1965).

[13] Golub, G. H., and Styan, G., "Numerical Computations for Univariate Linear Models," STAN-CS-236-71, Computer Science Department, Stanford University (1971).

[14] 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).

[15] Hellerman, E., and Rarick, D., "The Partitioned Preassigned Pivot Procedure," in D. Rose and R. Willoughby (eds.), *Sparse Matrices and Their Applications*, Plenum Press, 67–76 (1972).

[16] Householder, A. S., "Unitary Triangularization of a Nonsymmetric Matrix," JACM, vol. 5, 339–342 (1958).

[17] Householder, A. S., The Theory of Matrices in Numerical Analysis, Blaisdell, 122–139 (1964).

[18] Kendrick, D., "On the Leontief Dynamic Inverse," *Q. Jour. of Economics*, 693–696 (1972).

[19] Lawson, C. L., "Applications of Singular Value Analysis" in John Rice (ed.), *Mathematical Software*, Academic Press, 347–356 (1971).

[20] Martin, R. S., Peters, G., and Wilkinson, J. H., "Symmetric Decomposition of a Positive Definite Matrix" in J. H. Wilkinson and C. Reinsch (eds.) *Handbook for Automatic Computation, Volume II: Linear Algebra*, Springer Verlag, 134–151 (1971).

[21] Moler, C., "Matrix Computations with Fortran and Paging," CACM, vol. 15, no. 4, 268–270 (1972).

[22] Peters, G., and Wilkinson, J. H., "The Least Squares Problem and Pseudo-inverses," *The Computer Journal*, vol. 13, no. 3, 309–316 (1970).

[23] Steward, D., "Partitioning and Tearing Systems of Equations," *J. SIAM Numer. Anal.*, ser. B., vol. 2, no. 2, 345–365 (1965).

[24] Wampler, R. H., "On the Accuracy of Least Squares Computer Programs," *JASA*, vol. 65, no. 330, 549–565 (1970).

[25] Wilkinson, J. H., Rounding Errors in Algebraic Processes, 94–97, 86–91, 138, 156 (1963).

[26] Wilkinson, J. H., *The Algebraic Eigenvalue Problem*, Oxford University Press, 209–212, 243–244, 345–412, 517, 607, 612.

received: April 13, 1973