A computational method for evaluating generalized inverses

By R. P. Tewarson*

A method for computing the generalized inverse of a matrix is described, which makes use of the well-known Gauss–Jordan elimination method and the orthogonal triangularizations.

1. Introduction

Let A be an $m \times n$ matrix of rank r with real elements $(r \leqslant m \leqslant n)$. There exist elementary permutation matrices P and K such that

$$KAP = G = \begin{bmatrix} N & B \\ C & CN^{-1}B \end{bmatrix}, \qquad (1.1)$$

where N is an $r \times r$ non-singular matrix (Penrose, 1956). Let $m - r = c$ and $n - r = d$, then C is $c \times r$, B is $r \times d$ and $CN^{-1}B$ is $c \times d$. For the permutation matrices K and P we have $K^+ = K^{-1} = K^T$ and $P^+ = P^{-1} = P^T$. By direct substitution in the four defining relations for the generalized inverse given by Penrose (1955), it can be easily verified that

$$A^+ = KG^+P. \qquad (1.2)$$

From (1.1) we have

$$\begin{aligned} G &= \begin{bmatrix} N & B \\ C & CN^{-1}B \end{bmatrix} \\ &= \begin{bmatrix} N \\ C \end{bmatrix} [I, \Delta], \text{ where } \Delta = N^{-1}B. \\ &= \begin{bmatrix} I \\ CN^{-1} \end{bmatrix} N[I, \Delta]. \end{aligned}$$

It can be easily shown (Greville, 1960) that

$$G^+ = [I, \Delta]^+ N^{-1} \begin{bmatrix} I \\ CN^{-1} \end{bmatrix}^+. \qquad (1.3)$$

We shall use the Gauss–Jordan elimination method (Ralston, 1965; p. 339) to evaluate Δ, N^{-1}, CN^{-1}, P and K. This will be followed by orthogonal triangularizations. (Householder, 1964; p. 133 and Wilkinson, 1965; p. 152, 223) on $\begin{bmatrix} I \\ \Delta^T \end{bmatrix}$ and $\begin{bmatrix} I \\ CN^{-1} \end{bmatrix}$ to evaluate $[I, \Delta]^+$ and $\begin{bmatrix} I \\ CN^{-1} \end{bmatrix}^+$.

2. Use of Gauss–Jordan elimination

Suppose that the Gauss–Jordan elimination is performed on the matrix G such that N is reduced to the identity matrix. This is equivalent to

$$\begin{bmatrix} N^{-1} & 0 \\ -CN^{-1} & I \end{bmatrix} \begin{bmatrix} N & B \\ C & CN^{-1}B \end{bmatrix} = \begin{bmatrix} I & \Delta \\ 0 & 0 \end{bmatrix}. \qquad (2.1)$$

The computation can be arranged such that at the completion of the elimination process G gets transformed as follows:

$$G \xrightarrow{G-J} \begin{bmatrix} N^{-1} & \Delta \\ -CN^{-1} & 0 \end{bmatrix}. \qquad (2.2)$$

Since in practice the elimination is performed on A instead of G, the position of all the pivots when performing elimination on A therefore determines the permutation matrices P and K (Tewarson, 1966; §3).

3. Orthogonal triangularizations

For $k = 1, 2, \ldots, r$ let us define the following:

$$V^{(k+1)} = \theta_k V^{(k)}, \qquad (3.1)$$

where $\theta_k = I - \beta_k u^{(k)} u^{(k)^T}$ and $V^{(1)} = \begin{bmatrix} I \\ \Delta^T \end{bmatrix}$. (3.2)

If the element in the ith row and the jth column of $V^{(k)}$ is denoted by $v_{ij}^{(k)}$, then the scalar β_k is given by

$$\beta_k = [\sigma_k(\sigma_k + |v_{kk}^{(k)}|)]^{-1} \qquad (3.3)$$

where

$$\sigma_k = \left[\sum_{i=k}^{n} (v_{ik}^{(k)})^2 \right]^{1/2}. \qquad (3.4)$$

The n element column vector $u^{(k)}$ is given by

$$u_i^{(k)} = 0, \; i < k, \qquad (3.5)$$

$$u_k^{(k)} = sgn\,(v_{kk}^{(k)})(\sigma_k + |v_{kk}^{(k)}|), \qquad (3.6)$$

$$u_i^{(k)} = v_{ik}^{(k)}, \; i > k. \qquad (3.7)$$

Thus θ_k is a symmetric orthogonal matrix of order n, which in (3.1), transforms all the elements $v_{ik}^{(k)}$, $i > k$ to zero. (Householder, 1964; p. 133 and Wilkinson, 1965; p. 152, 223.) Since the columns of $\begin{bmatrix} I \\ \Delta^T \end{bmatrix}$ are linearly independent, we have

$$Q \begin{bmatrix} I \\ \Delta^T \end{bmatrix} = \begin{bmatrix} R \\ O \end{bmatrix}, \qquad (3.8)$$

where $Q = \theta_r \ldots \theta_2 \theta_1$ and R is an upper triangular non-singular matrix. In view of the fact that Q is an orthogonal matrix, from (3.8) it follows that

$$\begin{aligned} \begin{bmatrix} I \\ \Delta^T \end{bmatrix} &= Q^T \begin{bmatrix} R \\ O \end{bmatrix} \\ &= Q_r^T R, \end{aligned} \qquad (3.9)$$

* *Department of Applied Analysis, State University of New York, Stony Brook, New York,* This research was supported in part by the National Aeronautics and Space Administration, Washington, D.C., Grant No. NGR-33-015-013.

where Q_r^T denotes the first r columns of Q^T. If $Q_r^T = \begin{bmatrix}\alpha^T \\ \beta\end{bmatrix}$, where α^T is $r \times r$, then from (3.9) we have

$$\begin{bmatrix} I \\ \Delta^T \end{bmatrix} = \begin{bmatrix}\alpha^T \\ \beta\end{bmatrix} R$$

which gives

$$I = \alpha^T R$$

or

$$R^{T^{-1}} = \alpha, \tag{3.10}$$

where α is the matrix consisting of the elements in the first r rows and the first r columns of Q. From (3.9) we have $[I, \Delta] = R^T Q_r$ and because $[I, \Delta]$ has linearly independent rows, it follows (Greville, 1960) that

$$\begin{aligned}[I, \Delta]^+ &= (R^T Q_r)^+ \\ &= Q_r^T R(R^T Q_r Q_r^T R)^{-1} \\ &= Q_r^T R(R^T R)^{-1} \\ &= Q_r^T R^{T^{-1}},\end{aligned}$$

which in view of (3.10) gives

$$[I, \Delta]^+ = Q_r^T \alpha. \tag{3.11}$$

Now if we construct $\hat{Q}$ in a similar manner as Q, such that

$$\hat{Q}\begin{bmatrix} I \\ CN^{-1}\end{bmatrix} = \begin{bmatrix}\hat{R} \\ O\end{bmatrix}$$

where $Q = \hat{\theta}_r \dots \hat{\theta}_2 \hat{\theta}_1$ and $\hat{R}$ is an upper triangular non-singular matrix, then we have

$$\begin{aligned}\begin{bmatrix} I \\ CN^{-1}\end{bmatrix} &= \hat{Q}^T \begin{bmatrix}\hat{R} \\ O\end{bmatrix} \\ &= \hat{Q}_r^T \hat{R}, \end{aligned} \tag{3.12}$$

where $\hat{Q}_r^T$ denotes the first r columns of $\hat{Q}^T$. Let $\hat{Q}_r^T = \begin{bmatrix}\hat{\alpha} \\ \hat{\beta}\end{bmatrix}$, where $\hat{\alpha}$ is $r \times r$; then from (3.12) we have

$$\begin{bmatrix} I \\ CN^{-1}\end{bmatrix} = \begin{bmatrix}\hat{\alpha} \\ \hat{\beta}\end{bmatrix}\hat{R},$$

which gives

$$I = \hat{\alpha}\hat{R},$$

or

$$\hat{R}^{-1} = \hat{\alpha}, \tag{3.13}$$

where $\hat{\alpha}$ is the matrix consisting of the elements belonging to the first r rows and the first r columns of $\hat{Q}^T$. In view of the fact that $\begin{bmatrix} I \\ CN^{-1}\end{bmatrix}$ has linearly independent columns, it follows (Greville, 1960) from (3.12) that

$$\begin{aligned}\begin{bmatrix} I \\ CN^{-1}\end{bmatrix}^+ &= (\hat{Q}_r^T \hat{R})^+ \\ &= (\hat{R}^T \hat{Q}_r \hat{Q}_r^T \hat{R})^{-1} \hat{R}^T \hat{Q}_r \\ &= (\hat{R}^T \hat{R})^{-1} \hat{R}^T \hat{Q}_r \\ &= \hat{R}^{-1} \hat{Q}_r,\end{aligned}$$

which in view of (3.13) becomes

$$\begin{bmatrix} I \\ CN^{-1}\end{bmatrix}^+ = \hat{\alpha}\hat{Q}_r. \tag{3.14}$$

Finally, substituting (3.11) and (3.14) in (1.3) we have

$$G^+ = Q_r^T \hat{\alpha} N^{-1} \hat{\alpha} \hat{Q}_r,$$

which, in turn, when substituted in (1.2) yields

$$A^+ = K Q_r^T \hat{\alpha} N^{-1} \hat{\alpha} \hat{Q}_r P. \tag{3.15}$$

4. Final remarks

In floating-point computations, it is generally not easy to determine if some number is effectively zero or not. This fact leads to the following difficulty in the elimination process, namely, the problem of deciding whether a row of A has been transformed to zero or not. A technique, essentially due to Osborne (1965, p. 304) will now be described for the above problem. Let $L_i^{(k)}$ denote the ith row of A after $k - 1$ pivots have been chosen, where $i = 1, 2, \dots, m$ and $k = 1, 2, \dots, r$. Since $L_i^{(k)}$ consists of multiples of the rows of A added to $L_i^{(1)}$, it is therefore reasonable to compare the Euclidean norm of $L_i^{(k)}$ to that of $L_i^{(1)}$ and use $||L_i^{(k)}||/||L_i^{(1)}||$ as the criterion to decide whether a row is zero or not.

The Gauss–Jordan elimination is a reasonable choice for the evaluation of N^{-1}, Δ and $-CN^{-1}$, at least in a significantly large number of practical cases, in view of the following. In large scale linear programming computer codes (e.g., Honeywell, 1964; Share, 1964), the Gauss–Jordan elimination in the form of "the product form of inverse" (Tewarson 1966; Tewarson, 1967) is used in inverting the $r \times r$ submatrix of the $m \times m$ basis matrix. Usually r is less than m due to redundant constraints (artificial vectors at zero level) and the contradictory constraints (artificial vectors at non-zero level-problem infeasible). In both the cases those rows that turn out to be linear combinations of the others are determined as follows. If in a certain row, no pivot greater than a certain "pivot tolerance" can be found, then that row is considered to be a linear combination of the other rows in which pivots have already been chosen. (In connection with the design and writing of the linear programming compiler ALPS (Honeywell, 1964) the author solved about one hundred linear programming problems. These actual production problems were collected from diverse users of linear programming and ranged in size from 20 to 805 rows. A "pivot tolerance" of 10^{-5} appeared to be adequate for a 40-bit mantissa.)

We shall now point out some of the advantages of using the method described in this paper. If A is sparse (e.g. in linear programming (see Pyle, 1960)), then we can select the rows and columns that will constitute the matrix N, in such a manner that Δ as well as N^{-1} is kept sparse. Keeping Δ and N^{-1} sparse will not only save storage space but also keep the round-off errors low. The techniques described in Tewarson (1966) and

Tewarson (1967) can be utilized to keep Δ and N^{-1} sparse. For example, we count the number of non-zero elements in each row of A and call this the *row count vector.* Then the sum of those elements of the row count vector that correspond to the non-zero elements of a particular column j is called the *density measure* D_j of column j. Now the columns of A can be considered in ascending values of D_js as candidates for being chosen as the columns of N. Likewise the rows can be selected on the basis of ascending values of the elements of the row count vector; of course, some suitable updating (adjustment) of the elements of the two count vector would be needed as more and more columns of A are chosen to become the columns of N. Evidently the above-mentioned techniques can also be used in the other methods for computing the generalized inverses (Ben-Israel and Cohen, 1966). However, these methods require the formation of A^TA and in general A^TA tends to be denser than A; therefore, it seems that some of the advantages of keeping the various sub-matrices sparse may be offset.

It is easy to see that when triangularizing $\begin{bmatrix} I \\ \Delta^T \end{bmatrix}$ and $\begin{bmatrix} I \\ CN^{-1} \end{bmatrix}$ only the elements in the lower triangular part of the matrices Δ^T and CN^{-1} are to be transformed to zero. Due to this fact, the elementary Hermitian matrices of the type I—$2ww^T$ used in the triangularization will be sparse. This will save not only the computer storage but also minimize the round-off errors.

In (2.1), instead of the Gauss–Jordan elimination, the usual Gaussian elimination (Ralston 1965, p. 399) can be used as follows:

$$\begin{bmatrix} L^{-1} & 0 \\ -CN^{-1} & I \end{bmatrix}\begin{bmatrix} N & B \\ C & CN^{-1}B \end{bmatrix} = \begin{bmatrix} L^{-1}N & L^{-1}B \\ 0 & 0 \end{bmatrix}.$$

Where $N = LU$, L being a unit lower triangular and U an upper triangular matrix. The above is followed by the usual back substitution

$$\begin{aligned} U^{-1}[L^{-1}N, L^{-1}B] &= [I \quad N^{-1}B] \\ &= [I, \ \Delta]. \end{aligned}$$

In any case, it is recommended that complete pivoting be used in the elimination (Wilkinson, 1965; p. 212). When computing Q and $\hat{Q}$, it is possible to store them in factored form to replace $\begin{bmatrix} I \\ \Delta^T \end{bmatrix}$ and $\begin{bmatrix} I \\ CN^{-1} \end{bmatrix}$ respectively (Wilkinson, 1965; p. 235). Notice that R and $\hat{R}$ are not used in (3.15) and therefore need not be stored. A program based on the method given in the paper is available from Mr. T. Hasiotis, Applied Analysis Department, State University of New York, Stony Brook, N.Y., U.S.A. The program is written in FORTRAN IV for the IBM 7040/1401 system.

References

Ben-Israel, A., and Cohen, D. (1966). On iterative computation of generalized inverses and associated projections, *SIAM J. on Num. Anal.*, Vol. 3, p. 410.

Boot, J. C. G. (1963). The computation of the generalized inverse of singular or rectangular matrices, *Amer. Math. Mon.*, Vol. 70, pp. 302–303.

Greville, T. N. E. (1960). Some Applications of the Pseudoinverse of a Matrix, *SIAM Rev.*, Vol. 2, p. 15.

Householder, A. S. (1964). *The Theory of Matrices in Numerical Analysis*, New York: Blaisdell Publishing Co.

honeywell (1964). *ALPS*, Honeywell E.D.P., DSI-275, Wellesley Hills, Mass.

Osborne, E. E. (1965). Smallest least square solutions of linear equations, *J. SIAM*: Ser. B, Vol. 2, 1965, pp. 300–397.

Penrose, R. (1955). A Generalized Inverse for Matrices, *Proc. Cambridge Philos. Soc.*, Vol. 51, p. 406.

Penrose, R. (1956). On Best Approximate Solutions of Linear Matrix Equations, *Proc. Cambridge Philos. Soc.*, Vol. 52, p. 17.

Pyle, L. (1960). The generalized inverse in linear programming. *Doc. Th., Purdue U.*

Ralston, A. (1965). *A First Course in Numerical Analysis*, New York: McGraw Hill Book Co.

Share, (1962). *LP* 90 *Reference Manual*, SHARE Distr. Agency, No. 1377, White Plains, N.Y.

Tewarson, R. P. (1966). On the product form of inverse of sparse matrices, *SIAM Rev.*, Vol. 8, 3, pp. 336–342.

Tewarson, R. P. (1967). On the product form of inverses of sparse matrices and graph theory, *SIAM Rev.*, Vol. 9, 1, pp. 91–99.

Wilkinson, J. H. (1965). *The Algebraic Eigenvalue Problem*, London: Oxford University Press.

Dudley W. Hooper

Since this issue went to press the editors have heard, with deep regret, of the sudden death on 14 January 1968 of Mr. Dudley Hooper, Past President of The British Computer Society. Mr. Hooper was a founder of the Society and its predecessor, the London Computer Group, and he was the first editor of our companion publication, *The Computer Bulletin.* A full appreciation will be published in our next issue.