SU-326P30-23

METHODS FOR MODIFYING MATRIX FACTORIZATIONS

BY

P. E. GILL

G. H. GOLUB

W. MURRAY

M. A. SAUNDERS

STAN-CS-72-322

NOVEMBER 1972

COMPUTER SCIENCE DEPARTMENT
School of Humanities and Sciences
STANFORD UNIVERSITY

[†] METHODS FOR MODIFYING MATRIX FACTORIZATIONS

by

P. E. Gill,[*] G. H. Golub,[**] W. Murray[*] and M. A. Saunders[**,+]

Abstract

In recent years several algorithms have appeared for modifying the factors of a matrix following a rank-one change. These methods have always been given in the context of specific applications and this has probably inhibited their use over a wider field. In this report several methods are described for modifying Cholesky factors. Some of these have been published previously while others appear for the first time. In addition, a new algorithm is presented for modifying the complete orthogonal factorization of a general matrix, from which the conventional QR factors are obtained as a special case. A uniform notation has been used and emphasis has been placed on illustrating the similarity between different methods.

[†] Also National Physical Laboratory Report NAC 29.

[*] Division of Numerical Analysis and Computing, National Physical Laboratory, Tedding-ton, England.

[**] Computer Science Department, Stanford University, Stanford, California, U. S. A.

[+] Applied Mathematics Division, Division of Scientific and Industrial Research, Wellington, New Zealand.

The work of G. H. Golub and M. A. Saunders was in part supported by the AEC and the NSF.

Contents

1. Introduction

Consider the system of equations

$$Ax = b$$

where A is an n x n matrix and b is an n vector. It is well known that x should be computed by means of some factorization of A , rather than by direct computation of A^{-1} . The same is true when A is an mx n matrix and the minimal least squares solution is required; in this case it is usually not advisable (or necessary) to compute the pseudo-inverse of A explicitly (see Peters and Wilkinson, 1970).

Once x has been computed it is often necessary to solve a modified system

$$\bar{A}\,\bar{x} = \bar{b} \ .$$

Clearly, we should be able to modify the factorization of A to obtain factors for $\bar{A}$, from which $\bar{x}$ may be computed as before. In this paper we consider one particular type of modification, in which $\bar{A}$ has the form

$$\bar{A} = A + \alpha y z^T$$

where α is a scalar and y and z are vectors of the appropriate dimensions. The matrix $\alpha y z^T$ is a matrix of rank one, and the problem is usually described as that of _updating_ the factors of' A following a _rank-one modification_.

There are at least three matters for consideration in computing modified factors:

(a) The modification should be performed in as few operations as possible. This is especially true for large systems when there is a need for continual updating.

(b) The numerical procedure should be stable. Many of the procedures for modifying matrix inverses or pseudo-inverses that have been recommended in the literature are numerically unstable.

(c) If the original matrix is sparse it is desirable to preserve its sparsity as much as possible. The factors of a matrix are far more likely to be sparse than its inverse.

Modification methods have been used extensively in numerical optimization, statistics and control theory. In this paper, we describe some methods that have appeared recently, and we also propose some new methods. We are concerned mainly with algebraic details and shall not consider sparsity hereafter. The reader is referred to the references marked with an asterisk for details about particular applications.

1.1 Notation

The elements of a matrix A and a vector x will be denoted by a_{ij} and x_j respectively. We will use A^T to denote the transpose of A , and $\|x\|_2$ to represent the 2-norm of x , i.e. $\|x\|_2 = (x^T x)^{\frac{1}{2}}$. The symbols Q, R, L and D are reserved for matrices which are respectively orthogonal, upper triangular, unit lower triangular and diagonal. In particular we will write $D = \text{diag}\,(d_1, d_2, \ldots, d_n)$.

2. Preliminary results

Most of the methods given in this paper are based in some way upon the properties of orthogonal matrices. In the following we discuss some important properties of these matrices with the intention of using the material in later sections.

2.1 Givens and Householder matrices

The most common application of orthogonal matrices in numerical analysis is the reduction of a given n-vector z to a multiple of the first column of the identity matrix, i.e. find an $n \times n$ orthogonal matrix P such that

$$Pz = +\gamma e_1 \tag{1}$$

This can be done by using either a sequence of plane rotation (Givens) matrices or a single elementary hermitian (Householder) matrix. In order to simplify the notation we will define the former as

$$\left[S^{c} \quad 1^{s}_{-c} \right. \tag{2}$$

and call this a <u>Givens matrix</u> rather than a plane rotation since it corresponds to a rotation followed by a reflection about an axis.

This matrix has the same favorable numerical properties as the usual plane rotation matrix (see Wilkinson, 1965, pp. 131-152), but is now symmetric. The choice of c and s to perform the reduction

$$\begin{bmatrix} c & s \\ s & -c \end{bmatrix} \begin{bmatrix} z_1 \\ z_2 \end{bmatrix} = \begin{bmatrix} +\gamma \\ 0 \end{bmatrix}$$

is given by

$$\gamma^2 = z_1^2 + z_2^2$$

$$\gamma = \text{sign}\,(z_1)\sqrt{\gamma^2} \qquad (3)$$

and $$c = z_1/\gamma\ ,\ s = z_2/\gamma\ .$$

Note that $0 \le c \le 1$. In order to perform the reduction (1) we must embed the matrix (2) in the n-dimensional identity matrix. We shall use P_j^i to denote the matrix which, when applied to the vector $[z_1,z_2,\ldots,z_n]^T$, reduces z_j to zero by forming a linear combination of this element with z_i , i.e.

$$P_j^i z = \begin{array}{c} \begin{array}{cccccccccccccc} & & & & i & & & & & j & & & & \end{array} \\ \left[\begin{array}{cccccccccccccc}
1 \\
& \cdot \\
& & \cdot \\
& & & 1 \\
& & & & c & & & & & s \\
& & & & & 1 \\
& & & & & & \cdot \\
& & & & & & & \cdot \\
& & & & & & & & 1 \\
& & & & & & & & & -c \\
& & & & & & & & & & 1 \\
& & & & & & & & & & & \cdot \\
& & & & & & & & & & & & \cdot \\
& & & & & & & & & & & & & 1
\end{array}\right] \end{array}
\begin{bmatrix} z_1 \\ \cdot \\ \cdot \\ \cdot \\ z_i \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ z_j \\ \cdot \\ \cdot \\ \cdot \\ z_n \end{bmatrix}
= \begin{bmatrix} z_1 \\ \cdot \\ \cdot \\ \cdot \\ \bar{z}_i \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ 0 \\ \cdot \\ \cdot \\ \cdot \\ z_n \end{bmatrix}$$

There are several sequences of Givens matrices which will perform the reduction (1); for example

$$P_2^1 P_3^2 \ .\ .\ .\ .\ P_{n-1}^{n-2} P_n^{n-1}\ z\ ,$$

or

$$P_2^1 P_3^1 \ .\ .\ .\ {}_{n-1}^{-1} P_n^{-1}\ z\ .$$

To perform the same reduction in one step using a single Householder matrix, we have

$$P = I + \frac{1}{\tau} uu^T ,$$

where $$u = z + \gamma e_1 ,$$

$$\tau = -\gamma u_1 \tag{4}$$

and $$\gamma = \text{sign}(z_1) \|z\|_2 .$$

This time P is such that

$$Pz = -\gamma e_1 .$$

In the 2-dimensional case, we can show that

$$P = \begin{bmatrix} [illegible] & s \\ [illegible] & c \end{bmatrix} = -\begin{bmatrix} c & s \\ s & -c \end{bmatrix}$$

where c , s are the quantities defined earlier for the Givens matrix. Hence the 2×2 Householder and 2×2 Givens transformations are analytically the same, apart from a change of sign.

There are several applications where 2-dimensional transformations are used. The amount of computation needed to multiply a $2 \times n$ matrix A by a 2×2 Householder matrix computed using equations (4) is $4n + O(1)$ multiplications and $3n + O(1)$ additions. If this computation is arranged as suggested by Martin, Peters and Wilkinson (1971) and the relevant matrix is written as

$$I + \begin{bmatrix} -u_1/\gamma \\ -u_2/\gamma \end{bmatrix} [1 \quad u_2/u_1]$$

then the multiplication can be performed in 3n + O(1) multiplications and 3n + O(1) additions. Straightforward multiplication of A by a 2 x2 Givens requires 4n + O(1) multiplications and 2n + O(1) additions. Again the work can be reduced to 3n + O(1) multiplications and 3n + O(1) additions, as follows.

Let the Givens matrix be defined as in (3). Define the quantity

$$\mu = \frac{z_2}{z_1 + \gamma} , \; |\mu| \leq 1 .$$

Since $s = z_2/\gamma$ we can redefine s as

$$s = \mu(c+1) .$$

Similarly, we have

$$c = 1 - \mu s .$$

A typical product is now of the form

$$\begin{bmatrix} c & s \\ s & -c \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} c & s \\ \mu(c+1) & \mu s-1 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} \tag{5}$$

$$= \begin{bmatrix} y_1 c + y_2 s \\ y_1 \mu(c+1) + y_2(\mu s-1) \end{bmatrix}$$

which will be defined as

$$= \begin{bmatrix} \bar{y}_1 \\ \bar{y}_2 \end{bmatrix}$$

Consequently, in order to perform the multiplication (5) we form

$$\bar{y}_1 = cy_1 + sy_2$$

and $$\bar{y}_2 = \mu(y_1 + \bar{y}_1) - y_2 \ .$$

Note that this scheme is preferable only if the time taken to compute a multiplication is more than the time taken to compute an addition. Also it may be advisable with both algorithms to modify the computation of γ to avoid underflow difficulties.

In the following work we will consider only 2×2 Givens matrices, although the results apply equally well to 2×2 Householder matrices since as noted earlier, the two are essentially the same.

2.2 Products of Givens matrices

The following lemma will help define some new notation and present properties of certain products of orthogonal matrices.

Lemma I.

Let P^j_{j+1} be a Givens matrix defined as in (3). Then the product

$$P^{n-1}_n P^{n-2}_{n-1} \cdots P^1_2$$

is of the form

$$H_L(p,\beta,\gamma) = \begin{bmatrix} p_1\beta_1 & \gamma_1 & & & & \\ p_2\beta_1 & p_2\beta_2 & \gamma_2 & & & \\ p_3\beta_1 & p_3\beta_2 & p_3\beta_3 & \ddots & & \\ \vdots & \vdots & \vdots & & \ddots & \\ & & & & \gamma_{n-2} & \\ p_{n-1}\beta_1 & p_{n-1}\beta_2 & p_{n-1}\beta_3 & \cdots & p_{n-1}\beta_{n-1} & \\ p_n\beta_1 & p_n\beta_2 & p_n\beta_3 & \cdots & p_n\beta_{n-1} & p_n\beta_n \end{bmatrix}$$

where the quantities p_j, β_j and γ_j are defined by either of the following recurrence relations:

<u>Forward recurrence</u>

1. $p_1 = c_1/\pi$, $\beta_1 = \pi$, $\eta_1 = s_1/\pi$, $\gamma_1 = s_1$, where π is an arbitrary non-zero scalar;

$$2.\ \left.\begin{array}{ll} p_j = c_j\eta_{j-1}, & \gamma_j = s_j \\ \beta_j = -c_{j-1}/\eta_{j-1}, & \eta_j = s_j\eta_{j-1} \end{array}\right\} \quad j = 2,3,\ldots,n-1;$$

3. $p_n = \eta_{n-1}$, $\beta_n = -c_{n-1}/p_n$.

<u>Backward recurrence</u>

1. $p_n = \pi$, $\beta_n = -c_{n-1}/\pi$, $\eta_{n-1} = s_{n-1}/\pi$, $\gamma_{n-1} = s_{n-1}$, where π is an arbitrary non-zero scalar;

$$2.\ \left.\begin{array}{ll} p_j = c_j/\eta_j, & \gamma_{j-1} = s_{j-1} \\ \beta_j = -c_{j-1}\eta_j, & \eta_{j-1} = s_{j-1}\eta_j \end{array}\right\} \quad j = n-1,\ldots,3,2;$$

3. $p_1 = c_1/\beta_1$, $\beta_1 = \eta_1$.

<u>Proof</u>

We will prove the lemma in the forward recurrence case; the remaining case can be proved in a similar way. Assume that the product $P^{k}_{k+1}P^{k-1}_{k} \ldots . P^{3}_{4} P^{2}_{3} P^{1}_{2}$ $(k < n-1)$ is given by

$$\begin{bmatrix} p_1\beta_1 & \gamma_1 & & & & & & \\ p_2\beta_1 & p_2\beta_2 & \cdot & & & & & \\ \vdots & \cdot & \cdot & \cdot & & & & \\ \vdots & \cdot & & \cdot & \cdot & & & \\ p_k\beta_1 & p_k\beta_2 & \cdots & p_k\beta_k & \gamma_k & & & \\ \eta_k\beta_1 & \eta_k\beta_2 & \cdots & \eta_k\beta_k & -c_k & & & \\ & & & & & 1 & & \\ & & & & & & 1 & \\ & & & & & & & 1 \end{bmatrix} \tag{6}$$

This is true for $k=1$ by definition. The next product

$$P_{k+2}^{k+1} P_{k+1}^{k} P_{k}^{k+1} \cdots\cdots P_3^2 P_2^1$$

is given by

$$\begin{bmatrix}
p_1\beta_1 & \gamma_1 & & & & & \\
p_2\beta_1 & p_2\beta_2 & \ddots & & & & \\
\vdots & \vdots & \ddots & \ddots & & & \\
p_k\beta_1 & p_k\beta_2 & \cdots & p_k\beta_k & \gamma_k & & \\
c_{k+1}\eta_k\beta_1 & c_{k+1}\eta_k\beta_2 & \cdots & c_{k+1}\eta_k\beta_k & -c_{k+1}c_k & s_{k+1} & \\
s_{k+1}\eta_k\beta_1 & s_{k+1}\eta_k\beta_2 & \cdots & s_{k+1}\eta_k\beta_k & -s_{k+1}c_k & -c_{k+1} & \\
 & & & & & & 1 \\
 & & & & & & & \ddots
\end{bmatrix}$$

If we define $p_{k+1} = c_{k+1}\eta_k$, $\gamma_{k+1} = s_{k+1}$, $\beta_{k+1} = -c_k/\eta_k$, $\eta_{k+1} = s_{k+1}\eta_k$, then the product $P_{k+2}^{k+1} \cdots P_2^1$ is of a similar form to (6). Continuing in this way, and finally setting $p_n = \eta_{n\ 1}$ and $\beta_n = -c_{n-1}/p_n$, gives the required result.

For later convenience we shall use the notation

$$H_U(p,\beta,\gamma) = H_L(p,\beta,\gamma)^T .$$

The matrices $H_U(p,\beta,\gamma)$ and $H_L(p,\beta,\gamma)$ are defined as special upper and lower Hessenberg matrices respectively. In the same way we define a special upper triangular matrix $R(p,\beta,\gamma)$ as having the form

$$R(p,\beta,\gamma) = \begin{bmatrix} \gamma_1 & \beta_1 p_2 & \beta_1 p_3 & \cdots & \cdots & \beta_1 p_n \\ & \gamma_1 & \beta_2 p_3 & \cdots & \cdots & \beta_2 p_n \\ & & \gamma_3 & \cdots & \cdots & \beta_3 p_n \\ & & & \ddots & & \vdots \\ & & & & \ddots & \vdots \\ & & & & \gamma_{n-1} & \beta_{n-1} p_n \\ & & & & & \gamma_n \end{bmatrix}$$

The particular recurrence relation used to form $H_L(p,\beta,\gamma)$ will depend upon the order in which the Givens matrices are generated. For example, if P_n^{n-1} is formed first then the backward recurrence relation can be used.

Lemma II

Let $D = \mathrm{diag}(d_1, d_2, \ldots . d_n)$, $\Gamma_1 = \mathrm{diag}(\gamma_1, \gamma_2, \ldots \gamma_{n-1}, 1)$, $\Gamma_2 = \mathrm{diag}(1, \gamma_1, \gamma_2, \ldots . \gamma_{n-1})$ and $e = (1, 1, \ldots, 1, 1)^T$.

1. $DH_L(p,\beta,\gamma) = H_L(\bar{p},\bar{\beta},\bar{\gamma})D$

   where $\bar{\beta}_i = \beta_i/d_i$, $\bar{p}_i = d_i p_i$, $i=1,2,\ldots,n$, $\bar{\gamma}_i = d_i\gamma_i/d_{i+1}$, $i=1,2,\ldots,n-1$.

2. $R(p,\beta,\gamma)D = DR(\bar{p},\bar{\beta},\gamma)$

   where $\bar{\beta}_i = \beta_i/d_i$, $\bar{p}_i = d_i p_i$, $i=1,2,\ldots,n$.

3. $R(p,\beta,\gamma) = DR(p,\bar{\beta},e)$

   where $\bar{\beta}_i = \beta_i/\gamma_i$, $i=1,2,\ldots,n-1$, $d_i = \gamma_i$, $i=1,2,\ldots,n$.

4. $H_L(p,\beta,\gamma) = \Gamma_1 H_L(\bar{p},\beta,e)$

   $= H_L(p,\bar{\beta},e)\Gamma_2$.

where $\bar{p}_i = p_i/\gamma_i$ $(i < n)$, $\bar{p}_n = p_n$,

and $\bar{\beta}_i = \beta_i/\gamma_i$ $(i > 1)$, $\bar{\beta}_1 = \beta_1$.

5. If $H_L(\bar{p},\bar{\beta},\gamma) = H_L(p,\beta,\gamma)$ then $\bar{\gamma}_i = \gamma_i$ and

$p_i/\bar{p}_i = \beta_i/\bar{\beta}_i$ = constant , for all i=1, 2, ... n .

The next three lemmas show how the product of special matrices with various general matrices may be computed efficiently.

Lemma III

Let B be an $m \times n$ matrix and $H_L(p,\beta,\gamma)$ an $n \times n$ special lower Hessenberg matrix. The product $\bar{B} = BH$ can be formed using either of the following recurrence relations:

Forward recurrence

1. $w^{(1)} = Bp$, $\bar{b}_{i1} = \beta_1 w_i^{(1)}$, $i = 1, 2, \ldots, m$;

2. $$\left.\begin{aligned} w_i^{(j)} &= w_i^{(j-1)} - p_{j-1} b_{i,j-1} \\ \bar{b}_{ij} &= \gamma_{j-1} b_{i,j-1} + \beta_j w_i^{(j)} \end{aligned}\right\} \begin{aligned} &i = 1, 2, \ldots, m , \\ &j = 2, 3, \ldots, n . \end{aligned}$$

Backward recurrence

1. $w_i^{(n)} = p_n b_{in}$, $i = 1, 2, \ldots, m$;

2. $$\left.\begin{aligned} \bar{b}_{ij} &= \gamma_{j-1} b_{i,j-1} + \beta_j w_i^{(j)} \\ w_i^{(j-1)} &= p_{j-1} b_{i,j-1} + w_i^{(j)} \end{aligned}\right\} \begin{aligned} &i = 1, 2, \ldots, m , \\ &j = n, n-1, \ldots, 2 ; \end{aligned}$$

3. $\bar{b}_{i1} = \beta_1 w_i^{(1)}$, $i = 1, 2, \ldots, m$.

Proof

We will give a proof for the forward recurrence case. The backward recurrence case can be shown in a similar way. The first column of $\bar{B}$ is given by

$$\bar{b}_{i1} = \beta_1 \sum_{j=1}^{n} b_{ij}p_j \ , \quad i=1,\ 2,\ \bullet\ ,\ m\ .$$

If we define $\qquad w^{(1)} = Bp$,

or

$$_i^{(1)} = \sum_{j=1} b_{ij}p_j \ , \quad i=1,\ 2,\ \ldots,\ m\ , \qquad (7)$$

then

$$\bar{b}_{i1} = \beta_1 w_i^{(1} \ , \quad i=1,\ 2,\ .\ .\ .\ .\ m\ .$$

Forming the-second column we have

$$\bar{b}_{i2} = \gamma_1 b_{i1} + \beta_2 \sum_{j=2}^{n} b_{ij}p_j \ , \quad i=1,\ 2,\ .\ .\ .\ .\ m\ . \qquad (8)$$

From equation (7) we have

$$w_i^{(1)} - b_{i1}p_1 = \sum_{j=2} b_{ij}p_j \ , \quad i=1,\ 2,\ \ldots,\ m\ ,$$

and if this vector is defined as $w^{(2)}$, then (8) becomes

$$\bar{b}_{i2} = \gamma_1 b_{i1} + \beta_2 w_i^{(2)} \ , \quad i=1,\ 2,\ .\ .\ .\ .\ m\ .$$

The other columns of $\bar{B}$ are formed in exactly the same way.

The backward recurrence is more efficient unless the product Bp is known a priori. It is also more convenient if $\bar{B}$ occupies the same storage as B.

The forward and backward recurrence relations require approximately 75% of′ the work necessary to f′orm the same product by successively multiplying B by each of the individual Givens matrices. Since $H_L(p,\beta,\gamma)$ is an orthogonal matrix there exists a vector v such that

$$H_L(p,\beta,\gamma)v = \alpha e_1$$

and we can regard $H_L(p,\beta,\gamma)$ as the matrix which reduces v to αe_1 . An equivalent reduction can be obtained by multiplying v by a single Householder matrix. If we have a product of the form

$$H_L(p_1,\beta_1,\gamma_1)\ldots\ldots H_L(p_r,\beta_r,\gamma_r)B$$

the computational effort involved applying lemma III is less than that using a similar product of the equivalent Householder matrices. This is because if D is a diagonal matrix, the product can be written as

$$DH_L(\bar{p}_1,\bar{\beta}_1,e)\ldots\ldots H_L(\bar{p}_r,\bar{\beta}_r,e)B$$

using lemma II, parts 1 and 4.

Lemma IV

Let R be an upper triangular matrix and $H_U(p,\beta,\gamma)$ a special upper Hessenberg matrix. The product $\bar{H} = H_U(p,\beta,\gamma)R$ is an upper Hessenberg matrix which can be determined using either of the following recurrence relations:

Forward recurrence

1. Set $w^{(1)} = R^T p$,

$$\bar{h}_{1j} = \beta_1 w_j^{(1)} \quad , \quad j=1, 2, \ldots, n .$$

2. For i = 2, 3, ..., n , set

$$\bar{h}_{i,i-1} = \gamma_{i-1} r_{i-1,i-1} ,$$

$$\left.\begin{aligned} w_j^{(i)} &= w_j^{(i-1)} - p_{i-1} r_{i-1,j} \\ \bar{h}_{i,j} &= \gamma_{i-1} r_{i-1,j} + \beta_i w_j^{(i)} \end{aligned}\right\} \quad j=i, i+1, \ldots, n ,$$

Backward recurrence

1. $w_n^{(n)} = p_n r_{nn}$.

2. For i=n, n-1, ... 3, 2, set

$$\bar{h}_{i,i-1} = \gamma_{i-1} r_{i-1,i-1} , \quad w_{i-1}^{(i-1)} = p_{i-1} r_{i-1,i-1} ,$$

$$\left.\begin{aligned} \bar{h}_{ij} &= \gamma_{i-1} r_{i-1,j} + \beta_i w_j^{(i)} \\ w_j^{(i-1)} &= p_{i-1} r_{i-1,j} + w_j^{(i)} \end{aligned}\right\} \quad j=i, i+1, \ldots, n .$$

3. $\bar{h}_{1j} = \beta_1 w_j^{(1)}$, j=1, 2, ..., n .

Proof

This lemma is proved in a similar way to Lemma III.

Lemma V

Let R be upper triangular and $R(p,\beta,\gamma)$ a special upper triangular matrix. The product $\bar{R} = R(p,\beta,\gamma)R$ can be found using either of the following recurrence relations:

Forward recurrence

1. Set $w^{(1)} = R^I p$.

2. For $i=1, 2, \ldots\ldots n$, set

$$r_{ii} = \gamma_i r_{ii} ,$$

$$\left.\begin{aligned} w_j^{(i+1)} &= w_j^{(i)} - p_i r_{ij} \\ \bar{r}_{ij} &= \gamma_i r_{ij} + \beta_i w_j^{(i+1)} \end{aligned}\right\} \quad j=i+1, i+2, \ldots\ldots n .$$

Backward recurrence

1. For $i=n, n-1, \ldots\ldots 1$, set

$$w_i^{(i)} = p_i r_{ii} , \quad \bar{r}_{ii} = \gamma_i r_{ii} ,$$

$$\left.\begin{aligned} \bar{r}_{ij} &= \gamma_i r_{ij} + \beta_i w_j^{(i+1)} \\ w_j^{(i)} &= w_j^{(i+1)} + p_i r_{ij} \end{aligned}\right\} \quad j=i+1, i+2, \ldots\ldots n .$$

The forward recurrence relation can be formulated in the following alternative manner:

1. Set $w^{(1)} = R^T p$.

2. For $i=1,2,\ldots,n$, set

$$\bar{r}_{ii} = \gamma_i r_{ii} ,$$

$$\left.\begin{aligned} w_j^{(i+1)} &= w_j^{(i)} - p_i r_{ij} \\ r_{ij} &= (\gamma_i - \beta_i p_i) r_{ij} + \beta_i w_j \end{aligned}\right\} \quad j=i+1,\ldots,n .$$

This formulation requires an additional $n^2/2$ multiplications. It has been shown by Gentleman (1972) that the use of the more efficient relationship can lead to numerical instabilities in certain applications.

If the products of n 2×2 Givens matrices are accumulated into a single special matrix it has been demonstrated in lemmas I - V how certain savings can be made in subsequent computations. The nature of the forward and backward recurrence relations are such that when a value of s_j is very small underflow could occur in the subsequent computation of η_j. This will result in a division by zero during the computation of the next β_j. It will be shown in the following section how this difficulty can be avoided by judicious choice of the scalar π.

In certain applications the vector v which is such that

$$H_U(p,\beta,\gamma)v = \|v\|_2 e_1$$

is known. Since $H_U(p,\beta,\gamma)$ is orthogonal we have

$$v = \beta_1 \|v\|_2 p$$

and the vector v is parallel to the vector p . The value of π can be chosen such that the vector p is equal to v . This gives the modified algorithm:

Forward recurrence

1. $p_1 = v_1$, $\beta_1 = c_1/v_1$, $\gamma_1 = s_1$;

2. $\left.\begin{array}{l} p_j = v_j \, , \quad \gamma_j = s_j \\ \beta_j = -c_{j-1} c_j / v_j \end{array}\right\} \; j=2,3,\dots,n-1$;

3. $p_n = v_n$, $\beta_n = -c_{n-1}/v_n$.

We obtain this recurrence relation by writing $\pi = c_1/v_1$. A similar modification can be applied to the backward recurrence formula. The possible division by a near-zero v_j causes no problems since this only occurs when the corresponding Givens matrix is almost a permutation matrix and c_j is of the same order as v_j.

In the cases where v_j is not known a-priori, π can be set at 2^{-t}, where the computation is carried out on a machine with a t-digit binary mantissa. Since the value of η_j is such that

$$\eta_j = s_j s_{j-1} \cdots\cdots s_1/\pi$$

during forward recurrence, and

$$\eta_j = s_j s_{j+1} \cdots\cdots s_{n-1}/\pi$$

during backward recurrence, this choice of π is such that η_j is unlikely to underflow.

If even this strategy is insufficient the product of the Givens matrices can be broken into products of the form

$$\left[\begin{array}{c|c} I & 0 \\ \hline 0 & H_L(p',\beta',\gamma') \end{array}\right]_3 P^k_{k+1} \left[\begin{array}{c|c} H_L(p'',\beta'',\gamma'') & 0 \\ \hline 0 & I \end{array}\right]_3$$

where η_k is zero or intolerably small, and $H_L(p',\beta',\gamma')$ and $H_L(p'',\beta'',\gamma'')$ are smaller special matrices of dimension $(n-k) \times (n-k)$ and $k \times k$ respectively. Clearly a product of separate Givens matrices can be viewed as being a product of special matrices in which a "split" has occurred at every step.

3. Modification of the Cholesky factor

In this section we consider the case where a symmetric positive definite matrix A is modified by a symmetric matrix of rank one, i.e. we have

$$\bar{A} = A + \alpha zz^T .$$

Assuming that the Cholesky factors of A are known, viz.

$$A = LDL^T ,$$

we wish to determine the factors

$$\bar{A} = \bar{L}\bar{D}\bar{L}^T$$

It is necessary to make the assumption that A and $\bar{A}$ are positive definite since otherwise the algorithms for determining the modified factors are numerically unstable, even if the factorization of $\bar{A}$ exists. Several alternative algorithms will be presented and comments made upon their relative merits. Any of these general methods can be applied when A is of the form

$$A = B^T B$$

and rows or columns of the matrix B are being added or deleted. In this case it may be better to use specialized methods which modify the orthogonal factorization of B ,

$$QB = \begin{bmatrix} R \\ \hline 0 \end{bmatrix} .$$

The reader is referred to section 5 for further details. The methods in this section are all based upon the fundamental equality

$$\bar{A} = A + \alpha z z^T ,$$

$$= L(D + \alpha p p^T)L^T ,$$

where $$Lp = z .$$

If we form the factorization

$$D + \alpha p p^T = \tilde{L}\tilde{D}\tilde{L}^T , \qquad (9)$$

the required modified Cholesky factors are of the form

$$\bar{A} = L\tilde{L}\tilde{D}\tilde{L}^T L^T$$

giving

$$\bar{L} = L\tilde{L} \text{ and } \bar{D} = \tilde{D} ,$$

since the product of two lower triangular matrices is a lower triangular matrix. The manner in which the factorization (9) is performed will characterize a particular method.

Method Cl. Using classical Cholesky factorization

The Cholesky factorization of $D + \alpha pp^T$ can be formed directly. We will use this method to prove inductively that $\tilde{L}$ is special.

Assume at the jth stage of the computation that

$$\tilde{\ell}_{rs} = p_r\beta_s \,, \quad r=j, j+1, \ldots\ldots n \,, \quad s=1, 2, \ldots\ldots j-1 \tag{10}$$

and that all these elements have been determined. Explicitly forming the jth column of $\tilde{L}\tilde{D}\tilde{L}^T$ gives the following equations for $\tilde{d}_j$ and $\tilde{\ell}_{rj}$, $r=j+1, \ldots\ldots n$:

$$\sum_{i=1}^{j-1} \tilde{d}_i\tilde{\ell}_{ji}^2 + \tilde{d}_j = d_j + \alpha p_j^2 \tag{11}$$

and

$$\sum_{i=1}^{j-1} \tilde{d}_i\tilde{\ell}_{ji}\cdot\tilde{\ell}_{ri} + \tilde{d}_j\tilde{\ell}_{rj} = \alpha p_j p_r \,, \quad r=j+1, \ldots\ldots n \,. \tag{12}$$

Using the equation (10) with (11) and (12) gives

$$p_j^2 \sum_{i=1}^{j-1} \beta_i^2\tilde{d}_i + \tilde{d}_j = d_j + \alpha p_j^2$$

and

$$p_j p_r \sum_{i=1}^{j-1} \tilde{d}_i\beta_i^2 + \tilde{d}_j\tilde{\ell}_{rj} = \alpha p_j p_r \,, \quad r=j+1, \ldots\ldots n \,.$$

From the last equation we have

$$\tilde{\ell}_{rj} = \frac{p_j}{\tilde{d}_j}\left[\alpha - \sum_{i=1}^{j-1} \tilde{d}_i \beta_i^2\right] p_r \quad , \quad r=j+1, \ldots, n$$

and defining

$$\beta_j = \frac{p_j}{\tilde{d}_j}\left[\alpha - \sum_{i=1}^{j-1} \tilde{d}_i \beta_i^2\right]$$

gives $\tilde{\ell}_{rj} = p_r \beta_j$. Hence the subdiagonal elements of the jth column of L are multiples of the corresponding elements of the vector p .

Now forming the first column of $\tilde{L}\tilde{D}\tilde{L}^T$, we obtain the equations

$$\tilde{d}_1 = d_1 + \alpha p_1^2 ,$$

$$\tilde{d}_1 \tilde{\ell}_{r1} = \alpha p_1 p_r , \qquad r=2, \ldots . . n ,$$

which shows that the sub-diagonal elements of the first column of $\tilde{L}$ are multiples of the corresponding elements of p . Consequently we have proved that $\tilde{L}$ is special by induction.

This result implies that we need only compute the values of $\tilde{d}_j$, β_j , $j=1, \ldots . . n$ in order to obtain the factorization of $D + \alpha pp^T$. In practice we define the auxiliary quantity

$$\alpha_j = \alpha - \sum_{i=1}^{j-1} \tilde{d}_i \beta_i^2 .$$

The recurrence relations for α_j, $\tilde{d}_j$ and β_j then become

$$\left.\begin{aligned} \alpha_1 &= \alpha \\ \tilde{d}_j &= d_j + \alpha_j p_j^2 \\ \beta_j &= \alpha_j p_j / \tilde{d}_j \\ \alpha_{j+1} &= \alpha_j d_j / \tilde{d}_j \end{aligned}\right\} \quad j=1, 2, \ldots, n .$$

The product $L = L\tilde{L}$ can be computed in terms of the β_j by forward recurrence using Lemma V. Note that L and $\tilde{L}$ are both unit lower triangular matrices and that this results in some simplification of the algorithm. The vector $w^{(1)}$ needed to initialize the recurrence relations are known since $w^{(1)} = Lp = z$. Also each of the vectors $w^{(j)}(j=1, 2, \ldots, n)$ can be obtained during the jth stage of the initial back substitution $Lp = z$, since

$$w_r^{(j)} = \sum_{i=j}^{n} \ell_{ri} p_i = z_r - \sum_{i=1}^{j-1} \ell_{ri} p_i , \quad r=j, j+1, \ldots, n .$$

The final recurrence relations for modifying L and D are as follows:

Algorithm Cl

1. Define $\alpha_1 = \alpha$, $w^{(1)} = z$.

2. For $j=1, 2, \ldots, n$, compute

$$\begin{aligned} p_j &= w_j^{(j)} \\ \bar{d}_j &= d_j + \alpha_j p_j^2 \\ \beta_j &= p_j \alpha_j / \bar{d}_j \\ \alpha_{j+1} &= d_j \alpha_j / \bar{d}_j \end{aligned}$$

$$\left.\begin{aligned} w_r^{(j+1)} &= w_r^{(j)} - p_j \ell_{rj} \\ \bar{\ell}_{rj} &= \ell_{rj} + \beta_j w_r^{(j+1)} \end{aligned}\right\} \quad r=j+1, \ldots, n .$$

Using the expression for $w_r^{(j+1)}$ we can rearrange the equation for $\bar{\ell}_{rj}$ in the form

$$\begin{aligned}\bar{\ell}_{rj} &= \ell_{rj} + \beta_j(w_r^{(j)} - p_j\ell_{rj})\\ &= (1 - \beta_j p_j)\ell_{rj} + \beta_j w_r^{(j)}\\ &= (d_j/\bar{d}_j)\ell_{rj} + \beta_j w_r^{(j)} ,\end{aligned}$$

which is the form of the algorithm given by Gill and Murray (197'2). However, this increases the number of multiplications by 50%.

One of the earliest papers devoted to modifying matrix factorizations is that by Bennett (1965), in which LDU factors are updated following a rank m modification:

$$\bar{L}\bar{D}\bar{U} = LDU + XCY^T ,$$

where X, Y are $n \times m$ and C is $m \times m$. It should be noted that

(i) The algorithm given by Bennett is numerically stable only when $L = U^T$, $X = Y$ and both D and $\bar{D}$ are positive definite.

(ii) Algorithm Cl is identical to the special case of Bennett's algorithm when $m = 1$, $C = \alpha$, and $X = Y = z$.

The number of operations necessary to compute the modified factorization using algorithm Cl is $n^2 + O(n)$ multiplications and $n^2 + O(n)$ additions.

If the matrix A is sufficiently positive definite, that is, its smallest eigenvalue is sufficiently large relative to some norm of $\bar{A}$, then algorithm C1 is numerically stable. However, if $\alpha < 0$ and $\bar{A}$ is near to singularity it is possible that rounding error could cause the diagonal elements $\bar{d}_j$ to become zero or arbitrarily small. In such cases it is also possible that the $\bar{d}_j$ could change sign, even when the modification may be known from theoretical analysis to give a positive definite factorization. It may then be advantageous to use one of the following methods, because with these the resulting matrix will be positive definite <u>regardless of any numerical errors made</u>.

Method C2. Using Householder matrices

In this method the factorization (9) is performed using Householder matrices. To do this we must write

$$\bar{A} = LD^{\frac{1}{2}}(I + \alpha vv^T)D^{\frac{1}{2}}L^T ,$$

where v is the solution of the equations

$$LD^{\frac{1}{2}}v = z .$$

The matrix $I + \alpha vv^T$ can be factorized into the form

$$I + \alpha vv^T = (I + \sigma vv^T)(I + \sigma vv^T) \qquad (13)$$

by choosing

$$\sigma = \frac{\alpha}{1 + (1 + \alpha v^T v)^{\frac{1}{2}}} .$$

The expression under the root sign is a positive multiple of the determinant of $\bar{A}$. If $\bar{A}$ is positive definite σ will be real.

We now perform the Householder reduction of $I + \sigma vv^T$ to lower triangular form

$$\hat{L} = (I + \sigma vv^T)P_1P_2 \cdot \cdot \cdot P_{n\ 1} .$$

We will only consider application of the first Householder matrix P_1 . The effect of the remainder can easily be deduced.

Let

$$P_1 = I + \frac{1}{\tau} uu^T$$

and partition v in the form

$$v^T = [v_1 \quad w^T] \quad .$$

The (1,1) element of $I + \sigma vv^T$ is then

$$\theta = 1 + \sigma v_1^2$$

and P_1 must reduce the vector $[\theta \quad \sigma v_1 w^T]$ t0 a multiple of e_1^T . Using the relations of section 3 we define

$$\gamma^2 = \theta^2 + \sigma^2 v_1^2 w^T w \quad ,$$

$$u_1 = \theta + \gamma \quad ,$$

and

$$\tau = -\gamma u_1 \quad .$$

(Note that we have taken $\gamma = +\sqrt{\gamma^2}$, because we know that $\theta > 0$.) Now u has the form

$$u^T = [\, u_1 \quad \sigma v_1 w^T \,] \quad ,$$

i.e. elements $u_2, \ldots u_n$ are multiples of the vector w .

The result of applying the first Householder transformation can therefore be written as

$$(I + \sigma vv^T)(I + \frac{1}{\tau}\, uu^T) = \begin{bmatrix} -\gamma & 0 \\ \delta w & I + \bar{\sigma} ww^T \end{bmatrix}$$

for suitable values of the scalars δ and $\bar{\sigma}$ which will be determined as follows. The first column is given by

$$\begin{bmatrix} -\gamma \\ \delta w \end{bmatrix} = (I + \sigma vv^T)(e_1 + \frac{1}{\tau}\, u_1 u)$$

$$= \begin{bmatrix} 1 + \sigma v_1^2 & \sigma v_1 w^T \\ \sigma v_1 w & I + \sigma w w^T \end{bmatrix} \begin{bmatrix} 1 + \frac{1}{\tau} u_1^2 \\ \frac{1}{\tau} u_1 \sigma v_1 w \end{bmatrix}$$

which implies that

$$\delta w = (1 + \frac{1}{\tau} u_1^2)\sigma v_1 w + \frac{1}{\tau} u_1 \sigma v_1 (1 + \sigma w^T w) w \ .$$

so

A small amount of algebraic manipulation gives

$$\delta = -\sigma \frac{v_1}{\gamma} (2 + \sigma v^T v)$$

Similarly for the scalar $\bar{\sigma}$ we have

$$I + \bar{\sigma} w w^T = [\ \sigma v_1 w \quad I + \sigma w w^T\ ] \begin{bmatrix} \frac{1}{\tau} u_1 \sigma v_1 w^T \\ I + \frac{1}{\tau} \sigma^2 v_1^2 w w^T \end{bmatrix}$$

giving

$$\bar{\sigma} = \frac{1}{\tau} u_1 \sigma^2 v_1^2 + \sigma + \frac{1}{\tau} \sigma^2 v_1^2 + \frac{1}{\tau} \sigma^3 v_1^2 w^T w$$

which can be shown to be equal to

$$\bar{\sigma} = -\frac{1}{\tau} \sigma (1 + \gamma) = \frac{a(1 + \gamma)}{\gamma(\theta + \gamma)} \ .$$

The $(n-1) \times (n-1)$ submatrix $I + \bar{\sigma} w w^T$ has the same structure as $I + \sigma v v^T$ and a Householder matrix can be applied in exactly the same fashion. It can be shown that

$$1 + \bar{\sigma} w^T w = \frac{1}{\gamma}(1 + \sigma v^T v)$$

and so the sign choice in the definition of each of the Householder matrices remains the same.

For notational convenience we will write γ_j , θ_j , δ_j , and σ_{j+1} for the quantities γ , θ , δ , and $\bar{\sigma}$ at the jth step of the reduction, and use γ , δ for the vectors (γ_j) , (δ_j) .

The full reduction is now

$$(I + \sigma v v^T) P_1 P_2 \cdot\,.\,.P_{n\ 1} = R(v,\delta,-\gamma)^T$$

which gives

$$\bar{A} = L D^{\frac{1}{2}} R(v,\delta,-\gamma)^T R(v,\delta,-\gamma) D^{\frac{1}{2}} L^T .$$

From lemma II we have

$$\begin{aligned} R(v,\delta,-\gamma)D^{\frac{1}{2}} &= R(D^{\frac{1}{2}}v,\ 6,\ -D^{\frac{1}{2}}\gamma) , \\ &= D^{\frac{1}{2}} R(p,\ D^{-\frac{1}{2}}\delta,\ -\gamma) , \\ &= D^{\frac{1}{2}} \Gamma R(p,\beta,e) , \end{aligned}$$

where $\Gamma = \operatorname{diag}(\gamma_j)$

$$\left.\begin{aligned} p_j &= d_j^{\frac{1}{2}} v_j \\ \beta_j &= -\delta_j/(d_j^{\frac{1}{2}}\gamma_j) \end{aligned}\right\} \quad j=1,\ .\ .\ .\ .\ n\ .$$

(Note that p is the solution of $Lp = z$, as before.)

Following our convention for unit-triangular matrices we define

$$L(p,\beta,e) = R(p,\beta,e)^T .$$

The net result is that

$$\bar{L} = LL(p,\beta,e)$$

and
$$\bar{D} = \Gamma D \Gamma ,$$

which must be analytically equivalent to the factors obtained by algorithm C1. What we have done is find alternative expressions for β_j and $\bar{d}_j$, the most important being

$$\bar{d}_j = \gamma_j^2 d_j .$$

Since γ_j^2 is computed as a sum of squares, this expression guarantees that the computed $\bar{d}_3$ can never become negative. In algorithm C1, the corresponding relation is

$$\bar{d}_j = d_j + \alpha_j p_j^2$$

where $\text{sign}(\alpha_j) = \text{sign}(\alpha)$. If $\alpha < 0$ and $\bar{L}\bar{D}\bar{L}^T$ is nearly singular, it is possible that rounding errors could give $\bar{d}_j < 0$. In such cases algorithm C2 is to be preferred.

The analytical equivalence of the two algorithms can be seen through the relation between α_j and σ_j . For example, equation (13) implies that

$$\alpha_1 = \sigma_1(2 + \sigma_1 v^T v)$$

and if this is substituted into $\bar{d}_1 = d_1 + \alpha_1 p_1^2$ we get

$$\bar{d}_1 = \gamma_1^2 d_1 ,$$

which agrees with $\bar{D} = \Gamma D \Gamma$. In general if we define

$$\alpha_j = \sigma_j(2 + \sum_{i=j}^{n} v_i^T v_i)$$

the expression for δ_j simplifies, giving

$$\beta_j = -\frac{\delta_j}{d_j^{\frac{1}{2}}\gamma_j} = \frac{\alpha_j v_j}{d_j^{\frac{1}{2}}\gamma_j^2} = \frac{\alpha_j p_j}{d_j \gamma_j^2} = \frac{\alpha_j p_j}{\bar{d}_j}$$

which is the expression obtained for β_j in algorithm C1. In practice we retain this form for algorithm C2. The method for computing $\bar{L}$ from L and $L(p,\beta,e)$ is also the same as before. The iteration can be summarized as follows.

Algorithm C2

1. Solve $Lp = z$.
2. Define

$$\left.\begin{aligned} w_j^{(1)} &= z_j \\ s_j &= \sum_{i=j}^{n} p_i^2/d_i \equiv \sum_{i=j}^{n} q_i \end{aligned}\right\} \quad j=1, 2, \ldots n$$

$$\alpha_1 = \alpha,$$

$$\sigma_1 = \alpha/[1 + \sqrt{1 + \alpha s_1}\,].$$

3. For $j=1, 2, \ldots n$, compute

(a) $q_j = p_j^2/d_j$

(b) $\theta_j = 1 + \sigma_j q_j$

(c) $s_{j+1} = s_j - q_j$

(d) $\gamma_j^2 = \theta_j^2 + \sigma_j^2 q_j s_{j+1}$

(e) $\bar{d}_j = \gamma_j^2 d_j$

(f) $\beta_j = \alpha_j p_j/\bar{d}_j$

(g) $\alpha_{j+1} = \alpha_j/\gamma_j^2$

(h) $\sigma_{j+1} = \sigma_j(1 + \gamma_j)/[\gamma_j(\theta_j + \gamma_j)]$

$$\left.\begin{aligned}(i)\quad w_r^{(j+1)} &= w_r^{(j)} - p_j \ell_{rj} \\ \bar{\ell}_{rj} &= \ell_{rj} + \beta_j w_r^{(j+1)}\end{aligned}\right\} \quad r=j+1,\ j+2,\ .\ .\ .\ .\ n\ .$$

Note that the initial back substitution takes place separately from the computation of $L(p,\beta,e)$, because of the need to compute the vector p before computing s_1 . This adds $\frac{n^2}{2} + O(n)$ multiplications to the method but ensures that the algorithm will not break down under extreme circumstances and allows $\bar{L}$ to be computed by either the forward or backward recurrence relations given in Lemma V. The method requires $\frac{3}{2}n^2 + O(n)$ multiplications and $n+1$ square roots.

Method C3. Using Givens matrices

One of the most obvious methods of modifying the Cholesky factors of A in the particular case when $\alpha > 0$ is as follows.

Consider the reduction of the matrix $\begin{bmatrix} \alpha^{\frac{1}{2}}z & R^T \end{bmatrix}$ to lower triangular form, i.e.

$$\begin{bmatrix} \alpha^{\frac{1}{2}}z & R^T \end{bmatrix} P = \begin{bmatrix} \bar{R}^T & 0 \end{bmatrix} ,$$

where P is a sequence of Givens matrices of the form

$$P = P_2^1 P_3^2 \cdots \bullet \quad {}'E{+}l .$$

We have --

$$\begin{bmatrix} \bar{R}^T & 0 \end{bmatrix} \begin{bmatrix} \bar{R} \\ 0 \end{bmatrix} = \bar{R}^T\bar{R} ,$$

$$= \begin{bmatrix} \alpha^{\frac{1}{2}}z & R^T \end{bmatrix} PP^T \begin{bmatrix} \alpha^{\frac{1}{2}}z^T \\ R \end{bmatrix} ,$$

$$= R^TR + \alpha zz^T .$$

Consequently $\bar{R}^T$ is the required factor.

This algorithm can be generalized when $\alpha < 0$. The rank-one modification will be written as

$$\bar{R}^T\bar{R} = R^TR - \alpha zz^T , \quad \alpha > 0 ,$$

for convenience. The vector p is computed such that

$$R^Tp = z ,$$

and we set

$$\delta_n^2 = \frac{1 - \alpha p^T p}{\alpha}$$

We now form the matrix

$$\begin{bmatrix} p & R \\ \delta_n & 0 \end{bmatrix}$$

and pre-multiply by an orthogonal matrix P of the form

$$P = P_1^{n+1} \ . \ . \ . P_{n-1}^{n+1} P_n^{n+1}$$

such that the vector p is reduced to zero. This gives

$$P \begin{bmatrix} p & R \\ \delta_n & 0 \end{bmatrix} = \begin{bmatrix} 0 & \bar{R} \\ \delta_0 & r^T \end{bmatrix}$$

in which case the following relations must hold

$$p^T p + \delta_n^2 = \delta_0^2 \ , \qquad (14)$$

$$R^T p = \delta_0 r \ , \qquad (15)$$

$$R^T R = \bar{R}^T \bar{R} + r r^T \qquad (16)$$

Equation (14) implies that $\delta_0^2 = \frac{1}{\alpha}$, Equation (15) implies that $r = \frac{1}{\delta_0} z = \alpha^{\frac{1}{2}} z$, and finally (16) gives

$$R^T R = \bar{R}^T \bar{R} + \alpha z z^T \ ,$$

as required. This method requires $\frac{5}{2} n^2 + O(n)$ multiplications and $n+1$ square roots.

Method C4. Using Givens matrices II

For this method we will be modifying the factorization

$$\bar{R}^T\bar{R} = R^TR + \alpha zz^T .$$

From this equation we have

$$\bar{A} = R^T(I + \alpha pp^T)R , \tag{17}$$

where $R^Tp = z$.

We can write $\bar{A}$ in the form

$$\bar{A} = R^TP^TP(I + \alpha pp^T)P^TPR , \tag{18}$$

where P is an orthogonal matrix. The matrix P is chosen as a product of Givens matrices such that

$$Pp = P_2^1P_3^2 . . . P_{n-1}^{n-2}P_n^{n-1}p = \gamma e_1 , \tag{19}$$

where $|\gamma| = \|p\|_2$. The equation (17) can be written as

$$\bar{A} = R^TP^T(I + \alpha\gamma^2e_1e_1^T)PR .$$

As each Givens matrix P_{j+1}^j is formed it is multiplied into the upper triangular matrix R . This has the effect of filling in the sub-diagonal elements of R to give an upper Hessenberg matrix H . We have

$$H = PR ,$$

$$\bar{A} = H^TJ^TJH ,$$

where J is an identity matrix except for the (1,1) element which has the value $(1 + \alpha p^Tp)^{\frac{1}{2}}$. If $\bar{A}$ is positive definite, the square root will be real. The formation of the product JH modifies the first row of H to give

$$\bar{H} = J H$$

which is still upper Hessenberg. A second sequence of Givens matrices are now chosen to reduce $\bar{H}$ to upper triangular form, i.e.

$$\bar{P}\bar{H} = P_n^{n-1} P_{n-1}^{n-2} \cdots P_3^2 P_2^1 \bar{H} ,$$

$$= \bar{R} .$$

Then

$$\bar{A} = \bar{H}^T \bar{H}$$

$$= \bar{H}^T \bar{P}^T \bar{P} \bar{H}$$

$$= \bar{R}^T \bar{R}$$

as required. This algorithm requires $\frac{9}{2} n^2 + O(n)$ multiplications and $2n-1$ square roots.

Method C5. Using Givens matrices III

If we write equation (17) as in method C2, viz.

$$\bar{A} = R^T(I + \sigma pp^T)(I + \sigma pp^T)R ,$$

where $$\sigma = \frac{\alpha}{1 + (1 + \alpha p^T p)^{\frac{1}{2}}} .$$

If P is the matrix defined in (19) we can write

$$\bar{A} = R^T(I + \sigma pp^T)P^T P(I + \sigma pp^T)R .$$

$$= R^T H^T H R , \qquad (20)$$

where $$H = P(I + \sigma pp^T)$$

$$= P + \sigma \gamma e_1 p^T .$$

According to lemma I, P is a special upper Hessenberg matrix of the form

$$P = H_U(\bar{p}, \beta, \gamma)$$

for some vectors $\bar{p}$, β and γ . Now the first row of P is a multiple of $\bar{p}$ by definition, and furthermore $Pp = \gamma e_1$ implies that $p = \gamma P^T e_1$, so the first row of P is also a multiple of p . From lemma II it follows that by choosing $\bar{p}_n = p_n$ when forming P as a special matrix, we can ensure that

$$P = H_U(p, \beta, \gamma)$$

for some β and γ .

Assuming this choice of $\bar{p}_n$ is made, we have

$$H = H_U(p, \beta, \gamma) + \sigma \gamma e_1 p^T$$

$$= H_U(p,\bar{\beta},\gamma)$$

where $\bar{\beta}$ differs from β only in the first element, i.e.

$$\bar{\beta} = \beta + \sigma\gamma e_1 \ .$$

Now H can be reduced to upper triangular form $\tilde{R}$ by a second sequence of Givens matrices $\bar{P}$:

$$\bar{P}H = P_n^{n-1}P_{n-1}^{n-2}\ldots P_3^2P_2^1H = \tilde{R} \ .$$

It can be readily shown that $\tilde{R}$ is of the form

$$\tilde{R} = R(p,\tilde{\beta},\tilde{\gamma})$$

where the vectors $\tilde{\beta}$ and $\tilde{\gamma}$ are given by the following recurrence relations:

1. $\eta_1 = \bar{\beta}_1$;

2. $$\left.\begin{aligned} \tilde{\beta}_j &= c_j\eta_j + s_j\bar{\beta}_j \\ \gamma_j &= c_j\eta_j p_j + s_j\gamma_j \\ \eta_{j+1} &= s_j\eta_j - c_j\bar{\beta}_j \end{aligned}\right\} \ j=1, 2, \ldots \ldots n-1 ;$$

3. $\tilde{\gamma}_n = \eta_n$.

The quantities c_j and s_j are the elements of the Givens matrices in $\bar{P}$. They reduce the sub-diagonal elements γ_j of H to zero at each stage, and are defined in the usual way. The final product $\bar{R} = \tilde{R}R$ can be computed using lemma V.

This algorithm requires $2n^2 + O(n)$ multiplications and $2n-1$ square roots. The work has been reduced, relative to method C4, by accumulating both sequences of Givens matrices into the special matrix $\tilde{R}$ and modifying R just once, rather than twice.

4. Modification of the complete orthogonal factorization

If A is an $m \times n$ matrix of rank t, $m \geq n$, $t \leq n$, the complete orthogonal factorization of A is

$$QAZ = \begin{bmatrix} R & 0 \\ 0 & 0 \end{bmatrix} \tag{21}$$

where Q is a $m \times m$ orthogonal matrix, Z an $n \times n$ orthogonal matrix and R a $t \times t$ upper triangular matrix (see Fadeev et. al. (1968), Hanson and Lawson (1969)).

The pseudo-inverse of A is given by

$$A^{+} = Z \begin{bmatrix} R^{-1} & 0 \\ 0 & 0 \end{bmatrix} Q .$$

In order to obtain the pseudo-inverse of $\bar{A} = A + yz^T$, where y and z are m and n vectors respectively, we consider modifying the complete orthogonal factorization of A. (With no loss of generality we have omitted the scalar α.)

From equation (21) we have

$$Q\bar{A}Z = \begin{bmatrix} R & 0 \\ 0 & 0 \end{bmatrix} + pq^T$$

where $p = Qy$ and $q = Z^Tz$. If the vectors p and q are partitioned as follows:

$$p = \begin{bmatrix} u \\ \bar{u} \end{bmatrix} \begin{matrix} \}t \\ \}m-t \end{matrix} , \quad q = \begin{bmatrix} w \\ \bar{w} \end{bmatrix} \begin{matrix} \}t \\ \}n-t \end{matrix} ,$$

we can choose QI and ZI to be either single Householder matrices or products of Givens matrices such that

$$Q_I\bar{u} = \alpha e_1 \quad \text{and} \quad \bar{w}^T Z_I = \beta e_1^T ,$$

where α and β are scalars such that. $|\alpha| = \|\bar{u}\|_2$ and $|\beta| = \|\bar{w}\|_2$.
Note that application of these matrices leaves the matrix R unchanged.
For convenience we will' now work with the $(t+1) \times (t+1)$ matrix SI which is defined as

$$S_- = \begin{bmatrix} R & \\ & \end{bmatrix} \quad \begin{bmatrix} \\ \\ \end{bmatrix}^T \quad \beta] .$$

We next perform two major steps which will be called <u>sweeps</u>.

<u>First Sweep</u>

Choose an orthogonal matrix Q_{II} such that

$$Q_{II}\begin{bmatrix} u \\ \alpha \end{bmatrix} = P_2^1 P_3^2 . . . P_t^{t-1} P_{t+1}^t \begin{bmatrix} u \\ \alpha \end{bmatrix} = \gamma_1 e_1$$

where $\gamma_1^2 = \|u\|_2^2 + \alpha^2$. If S_I is multiplied on the left by Q_{II} and the resulting product defined as SII , we have

$$S_{II} = Q_{II}S_I = \begin{bmatrix} r_{II}^T & 0 \\ R_{II} & 0 \end{bmatrix} + \gamma_1 e_1 [w^T \ \beta] = \begin{bmatrix} \bar{r}_{II}^T & \gamma_1\beta \\ R_{II} & 0 \end{bmatrix} ,$$

where R_{II} is an upper triangular matrix. The t diagonal elements of R_{II} are filled in one at a time by the application of each 2 x 2 orthogonal matrix. We have defined

$$\bar{r}_{II}^T = r_{II}^T + \gamma_1 w^T .$$

Second Sweep

We now construct an orthogonal matrix Q_{III} which, when applied to S_{II} from the left, reduces S_{II} to upper triangular form. If this triangular matrix is defined as $\tilde{S}_{III}$ we have

$$\tilde{S}_{III} = Q_{III} \begin{bmatrix} \bar{r}_{II}^T & \gamma_1\beta \\ R_{II} & 0 \\ & I \end{bmatrix} = \begin{bmatrix} R_{III} & s_{III} \\ 0 & \delta_{III} \end{bmatrix} ,$$

where Q_{III} is of the form

$$Q_{III} = \tilde{P}_{t+1}^{t} \cdots P_3^2 P_2^1 .$$

The matrix $\tilde{S}_{III}$ may or may not be the upper triangular matrix required, depending upon $\rho(\bar{A})$, the rank of $\bar{A}$. The different cases that can arise are summarized in the following table:

$\alpha \backslash \beta$	$= 0$	$\neq 0$
$= 0$	$\rho(\bar{A}) = t$ or $t-1$	$\rho(A) = t$
$\neq 0$	$\rho(A) = t$	$\rho(A) = t + 1$

Case I. $\alpha \neq 0, \beta \neq 0$

In this case S_{III} has full rank and

$$\begin{bmatrix} R_{III} & s_{III} \\ 0 & \delta_{III} \end{bmatrix} = \bar{R} .$$

The final orthogonal matrix $\bar{Q}$ is given by

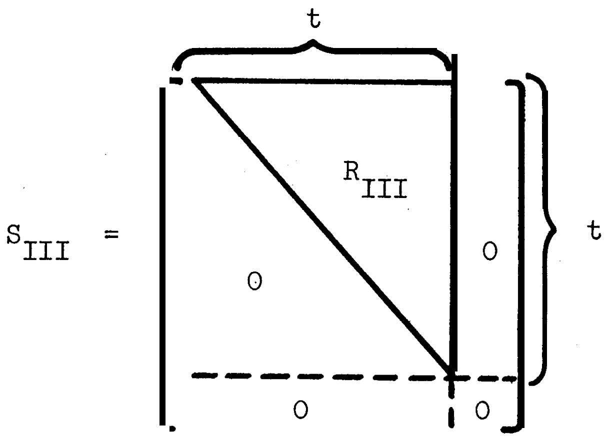
$$\bar{Q} = \underbrace{\left[\begin{array}{c|c} Q_{III} & 0 \\ \hline 0 & I \end{array}\right]}_{t+1} \underbrace{\left[\begin{array}{c|c} Q_{II} & 0 \\ \hline 0 & I \end{array}\right]}_{t+1} \underbrace{\left[\begin{array}{c|c} I & 0 \\ \hline 0 & Q_I \end{array}\right]}_{t} Q \qquad (22)$$

and

$$\bar{Z} = Z \underbrace{\left[\begin{array}{c|c} I & 0 \\ \hline 0 & Z_I \end{array}\right]}_{t} .$$

Case II. $\alpha \neq 0$. $\beta = 0$

If the first and second sweeps are followed carefully it can be seen that SIII is of the form

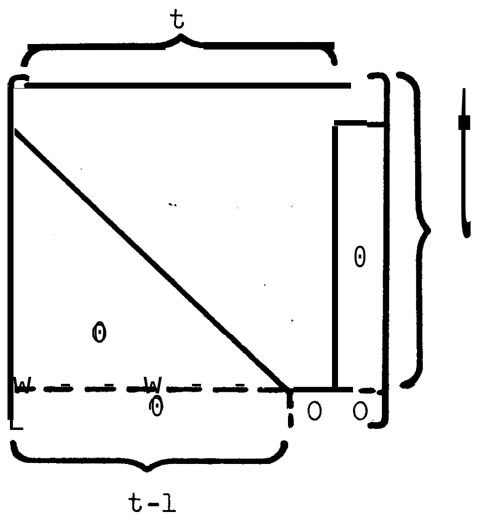
S_{III} =


i.e. $s_{III} = 0$ and $\delta_{III} = 0$. As in Case I, S_{III} is in the required form and we define the modified factors accordingly.

Case III. $\alpha = 0$, $\beta \neq 0$

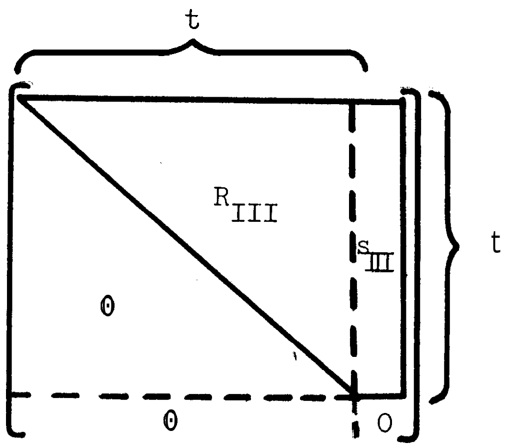
The first orthogonal transformation of the first sweep is an identity, and the matrix SII has the form

S_{II} =



Application of the second sweep (Q_{III}) gives the matrix SIII in the form

S_{III} =



i.e. $\delta_{III} = 0$.

An orthogonal matrix ZII is now applied on the right to reduce s_{III} to zero, thus:

$$S_{III}Z_{II} = S_{III}P^{t}_{t+1}\cdot P^{t-1}_{t+1}\cdot P^{t-2}_{t+1}\cdot\,.\,\cdot P_{t+1}$$

$$= \left[\begin{array}{c|c} \bar{R} & 0 \\ \hline 0 & 0 \end{array}\right] .$$

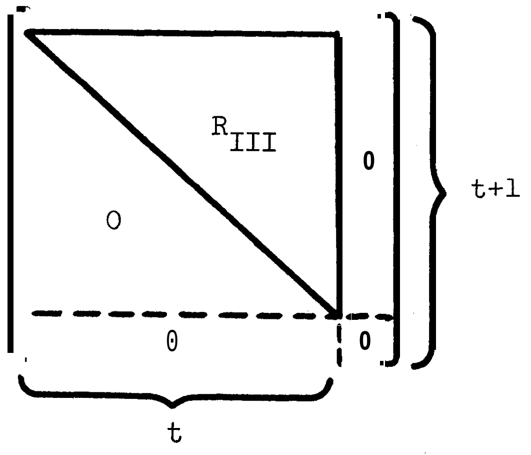
The modified factors are $\overline{Q}$ as defined in (22), and

$$\overline{Z} = Z \left[\begin{array}{c:c} I & \\ \hdashline & Z_I \end{array}\right] \left[\begin{array}{c:c} Z_{II} & \\ \hdashline & I \end{array}\right] .$$

<u>Case IV. $\alpha = 0$, $\beta = 0$, $\rho(\overline{A}) = t$</u>

The matrix SIII has the form

SIII =

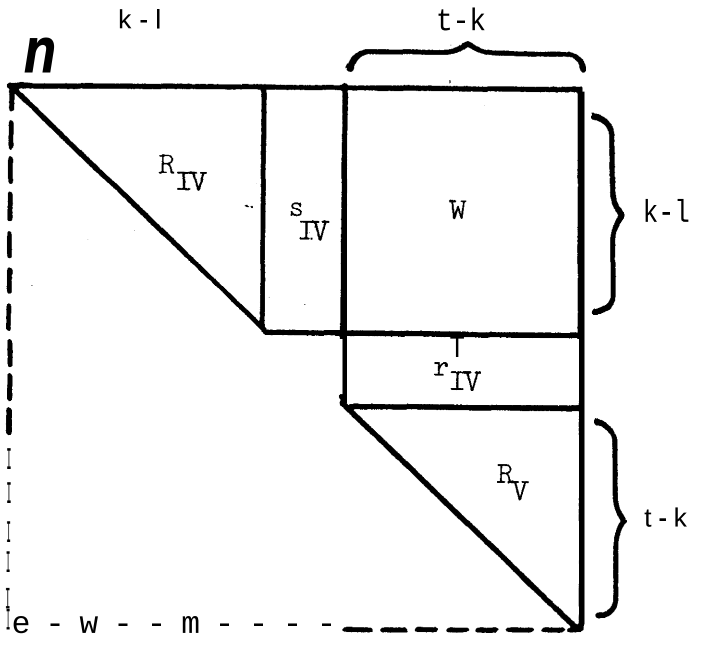


If the diagonal elements of RIII are all non-zero then rank $(\overline{A})$ = rank (RIII) = t and the factors are completely determined. Otherwise, exactly one of the diagonal elements of R_{III} may be zero, since the rank of $\overline{A}$ can drop to t-1 . In this case, two more partial sweeps must be made to reduce RIII to strictly upper triangular form, as follows.

<u>Case V. $\alpha = 0$, $\beta = 0$, $\rho(\overline{A}) = t-1$</u>

Suppose that the k-th diagonal of RIII is zero. The matrix can be partitioned in the form

$R_{III} =$



where R_{IV}, R_V are upper triangular with dimensions (k-l) $\times$ (k-l) and (t-k) $\times$ (t-k) respectively. An orthogonal transformation Q_{IV} is now applied on the left to reduce the submatrix $\begin{bmatrix} r_{IV}^T \\ R_V \end{bmatrix}$ to upper triangular form in exactly the same way as the first sweep. Similarly, a transformation Z_{II} is applied (independently) from the right to reduce s_{IV} to zero in the submatrix $[R_{IV}\ s_{IV}]$. Thus

$$Q_{IV}R_{III}Z_{II} = \left[\begin{array}{c:c:c} \bar{R}_{IV} & 0 & W \\ \hdashline \multicolumn{2}{c:}{0} & \bar{R}_V \\ \hdashline \multicolumn{2}{c:}{} & 0 \end{array}\right]$$

where $Q_{IV} = P_t^k P_{t-1}^k \cdot\ .\ \bullet^k\ P_{k+2}^k P_{k+1}^k$

and $Z_{II} = P_k^{k-1} P_k^{k-2}\ .\ .\ .\ P_k^2 P_k^1$.

Finally a permutation matrix Z_{III} is applied to move the column of zeros to the right:

$$\left[\begin{array}{c|c|c} \bar{R}_{IV} & 0 & W \\ \hline 0 & & \bar{R}_V \\ \cline{3-3} & & 0 \end{array}\right] Z_{III} = \left[\begin{array}{cc|c} \bar{R}_{IV} & W & \\ & & 0 \\ & \bar{R}_V & \\ \hline 0 & & 0 \end{array}\right] = \left[\begin{array}{c|c} \bar{R} & 0 \\ \hline 0 & 0 \end{array}\right] .$$

The modified factors are

$$\bar{Q} = \left[\begin{array}{c|c} Q_{IV} & \\ \hline & I \end{array}\right] \left[\begin{array}{c|c} Q_{III} & \\ \hline & I \end{array}\right] \left[\begin{array}{c|c} Q_{II} & \\ \hline & I \end{array}\right] \left[\begin{array}{c|c} I & \\ \hline & Q_I \end{array}\right] Q$$

and $\bar{Z} = Z \left[\begin{array}{c|c} I & \\ \hline & Z_I \end{array}\right] \left[\begin{array}{c|c} Z_{II} & \\ \hline & I \end{array}\right] \left[\begin{array}{c|c} Z_{III} & \\ \hline & I \end{array}\right] .$

The number of operations necessary to compute the modified factors are summarized in the following table:

Description	Order of multiplications
Compute p , q .	$m^2 + n^2$
Determine α , β .	$4m(m-t) + 4n(n-t)$
First sweep	$2t^2 + 4mt$
Second sweep	$2t^2 + 4mt$
Additional computation for case III	$2t^2 + 4nt$
*Additional computation for case V	$\frac{4}{3}t^2 + 2t(n+m)$

*It has been assumed that if $W(k)$ is the amount of work when the kth diagonal element of R_{III} is zero, then the expected work is

$$\frac{1}{t}\sum_{k=1}^{t} \cdots .$$

The maximum amount of computation necessary, which is of the order of $6\frac{2}{3}t^2 + 5(m^2+n^2) + 2t(3m-n)$ multiplications, will occur when case V applies. In the special case when $\bar{A}$ and A are both of full column rank then Z is the identity and the amount of computation is of the order of $5m^2 + 4n^2 + 4mn$ multiplications. This reduces to $13n^2$ when m=n .

4.1 Use of special matrices

The-number of operations can be decreased if some of the properties of special matrices outlined in section 2 are utilized. Each Givens matrix must be multiplied into a Q matrix, Z matrix or upper triangular matrix, depending upon the current stage of the algorithm. These multiplications can be performed by accumulating the product of each set of Givens matrices into the associated special matrix. Each Q_I , Z_I , Q_{II} , Z_{II} ,. . .etc. will be either a special matrix or a permutation matrix. The orthogonal matrices Q_I , ZI ,. . .etc. will be formed, using Lemma I and Lemma II, as products of the form $\Delta_I\tilde{Q}_I$, $\nabla_I Z_I$, $\Delta_{II}Q_{II}$, VIIZII ,...etc. where Δ_I ,∇_I, Δ_{II}, ∇_{II},...etc. are diagonal matrices and $\tilde{Q}_I$, $\tilde{Z}_I$,. . .etc. are special upper (lower) Hessenberg matrices with unit sub-(super-) diagonals. In addition we assume that we modify the factorization

$$QAZ = \left[\begin{array}{c|c} DL^T & 0 \\ \hline 0 & 0 \end{array}\right] .$$

At the initial stage DL^T is unaffected by the pre- and post- multiplication with $\Delta_I \tilde{Q}_I$ and $\tilde{Z}_I \nabla_I$. The products

$$\left[\begin{array}{c|c} I & 0 \\ \hline 0 & \Delta_I \tilde{Q}_I \end{array}\right] Q \; ; \qquad Z \left[\begin{array}{c|c} I & 0 \\ \hline 0 & \tilde{Z}_I \nabla_I \end{array}\right]$$

can be formed using Lemma III, the diagonal matrices being kept separate from the orthogonal products.

During the first sweep we require the product

$$Q_{II} \left[\begin{array}{c|c} R & 0 \\ \hline 0 & 0 \end{array}\right] .$$

If this matrix is written in the form

$$\Delta_{II} \tilde{Q}_{II} \left[\begin{array}{c|c} DL^T & 0 \\ \hline 0 & 0 \end{array}\right] ,$$

it can be evaluated by bringing the diagonal matrix D to the left of $\tilde{Q}_{II}$ by suitably altering the special matrix $\tilde{Q}_{II}$ to $\tilde{Q}'_{II}$ as in Lemma II. The remaining product involving $\tilde{Q}'_{II}$ and L^T can be formed using Lemma III with backward recurrence. The multiplication of $\tilde{Q}'_{II}$ by the current orthogonal matrix is performed similarly to that involving $\tilde{Q}_I$ except that again the diagonal Δ_I must be brought through by altering $\tilde{Q}_{II}$ to $\tilde{Q}''_{II}$ (say).

If the remainder of the computation is carried out using the same techniques as those just described, the number of multiplications can be summarized as follows:

Description	Order of multiplications
Compute p , q	$m^2 + n^2$
Determine α , β	$2m(m-t) + 2n(n-t)$
First sweep	$t^2 + 2mt$
Second sweep	$2t^2 + 2mt$
Additional computation for case III	$2t^2 + 2nt$
Additional computation for case V	$\frac{4}{3}t^2 + t(n+m)$

The maximum amount of computation necessary is now of the order of $4\frac{1}{3}t^2 + 3(m^2 + n^2) + t(3m-n)$ multiplications, and this reduces to $3(m^2 + n^2) + 2mn$ multiplications in the full rank case. When $n=m=t$ the algorithm requires $8n^2 + O(n)$ operations.

5. Special rank-one modifications

We now consider some special cases of the complete orthogonal factorization which occur frequently, namely adding and deleting rows and columns from A . These cases deserve special attention because the modifications can be done in approximately half as many operations as in the general case. Since in most applications A is of full column rank, we will deal specifically with this case and modify the factorization

$$QA = \begin{bmatrix} R \\ \hline 0 \end{bmatrix}$$

where A is $m \times n$, $m \geq n$.

5.1 Adding and deleting rows of A

We first consider adding a row a^T to A . Assuming without loss of generality that this row is added in the (m+1)th position, we have

$$\left[\begin{array}{c|c} Q & 0 \\ \hline 0 & 1 \end{array}\right] \left[\begin{array}{c} A \\ \hline a^T \end{array}\right] = \left[\begin{array}{c} R \\ \hline 0 \\ \hline a^T \end{array}\right] \equiv T .$$

Elementary orthogonal transformations are now applied from the left to reduce a^T to zero while maintaining the triangularity of R . This is done by defining the sequence

$$T^{(1)} = T , \quad T^{(j+1)} = P^j_{m+1} T^{(j)} , \quad j=1,2,\ldots,n ,$$

where P^j_{m+1} reduces the $(m+1,j)$ element of $T^{(j)}$ to zero. Note in particular the effect on the column e_{m+1} which has been added to Q .

The first n elements are filled in one by one, thereby forming the last column of $\bar{Q}$:

$$P^n_{m+1} P^{n-1}_{m+1} \cdot\ .\ \cdot P^1_{m+1} \left[\begin{array}{c|c} Q & e_{m+1} \\ \hline 0 & \end{array}\right] = \bar{Q}$$

$$= \left[\begin{array}{c|c} \bar{Q}_m & \bar{q}_{m+1} \end{array}\right] \quad \text{say.}$$

Elements n+1, n+2, . . .,m of $\bar{q}_{m+1}$ remain 7ero.

To <u>remove</u> a row from A , we now simply reverse the above process. This time we have

$$QA = \left[\begin{array}{c|c} Q_m & q_{m+1} \end{array}\right] \left[\begin{array}{c} \bar{A} \\ \hline a^T \end{array}\right] \begin{array}{c} \left[\begin{array}{c} R \\ \hline 0 \\ \hline 0 \end{array}\right] \end{array} \begin{array}{l} \}\ n \\ \}\ m-n \\ \}\ 1 \end{array}$$

giving

$$Q_m\bar{A} + q_{m+1}a^T = QA\ .$$

Transformations $P^{m+1}_m, P^{m+1}_{m\ 1}, \ldots, P^{m+1}_1$ are chosen such that

$$Pq_{m+1} \equiv P^{m+1}_1 \cdot\ .\ \bullet\ P^{m+1}_{m-1} P^{m+1}_m q_{m+1} = e_{m+1}\ .$$

The last n transformations each introduce a non-zero into the bottom row of

$$\left[\begin{array}{c} R \\ \hline 0 \\ \hline 0 \end{array}\right]$$

(from right to left), giving

$$PQA = \left[\begin{array}{c} \bar{R} \\ \hline 0 \\ \hline r^T \end{array}\right] .$$

Looking at the effect on the various partitions of Q, we have

$$PQ = \left[\begin{array}{c|c} \bar{Q} & 0 \\ \hline u^T & 1 \end{array}\right]$$

and since PQ is orthogonal it follow immediately that u = 0 . Thus

$$PQ \left[\begin{array}{c} \bar{A} \\ \hline a^T \end{array}\right] = \left[\begin{array}{c|c} \bar{Q} & 0 \\ \hline 0 & 1 \end{array}\right] \left[\begin{array}{c} \bar{A} \\ \hline a^T \end{array}\right]$$

$$= \left[\begin{array}{c} \bar{R} \\ \hline 0 \\ \hline r^T \end{array}\right]$$

so that r = a , and also

$$\bar{Q}\,\bar{A} = \left[\begin{array}{c} \bar{R} \\ \hline 0 \end{array}\right]$$

as required.

Often it is necessary to modify R without the help of Q. In this case we really want $\bar{R}$ such that

$$\bar{R}^T\bar{R} = R^TR \pm aa^T .$$

so clearly the methods of section 3 would be applicable. Alternatively we can continue to use elementary orthogonal transformations as just described. Adding a row to A is simple because Q was not required in any case. To delete a row we first solve $R^Tp = a$ and compute $\delta^2 = 1 - \|p\|^2$. The vector

$$\begin{bmatrix} p \\ \hdashline 0 \\ \hdashline \delta \end{bmatrix} \begin{matrix} \}\ n \\ \}\ m-n \\ \}\ 1 \end{matrix} \qquad (23)$$

now plays exactly the same role as q_{m+1} above. Dropping the unnecessary zeros in the center of this vector, we have

$$P_1^{n+1} \cdots P_{n-1}^{n+1}P_n^{n+1} \left[\begin{array}{c:c} p & R \\ \hdashline \delta & 0 \end{array}\right] = \left[\begin{array}{c:c} 0 & \bar{R} \\ \hdashline 1 & r^T \end{array}\right]$$

where as usual, the sequence $\{P_j^{n+1}\}$ has the effect of reducing p in (23) to zero and introducing the vector r^T beneath $\bar{R}$. Since the P_j^{n+1} are orthogonal it follows that

$$\left[\begin{array}{c:c} 0 & 1 \\ \hdashline \bar{R}^T & r \end{array}\right] \left[\begin{array}{c:c} 0 & \bar{R} \\ \hdashline 1 & r^T \end{array}\right] = \left[\begin{array}{c:c} p^T & \delta \\ \hdashline R^T & 0 \end{array}\right] \left[\begin{array}{c:c} p & R \\ \hdashline \delta & 0 \end{array}\right]$$

or

$$\left[\begin{array}{c:c} 1 & r^T \\ \hdashline r & \bar{R}^T\bar{R} + rr^T \end{array}\right] = \left[\begin{array}{c:c} \|p\|^2 + \delta^2 & p^TR \\ \hdashline R^Tp & R^TR \end{array}\right]$$

so that $r = R^T p = a$, and

$$\bar{R}^T\bar{R} = R^T R - aa^T$$

as required.

5.2 Adding and deleting columns of A

Suppose a column is added to the matrix A , giving

$$\bar{A} = \left[\begin{array}{c|c} A & a \end{array}\right] .$$

Since

$$QA = \left[\begin{array}{c} R \\ \hline 0 \end{array}\right] ,$$

we have

$$Q\bar{A} = \left[\begin{array}{c|c} R & u \\ \hline 0 & v \end{array}\right] , \qquad (24)$$

where $[u^T \mid v^T] = a^T Q^T$ and u and v are n and m-n vectors respectively. If an orthogonal matrix P is constructed such that

$$Pv = \left[\begin{array}{c} u \\ \hline \gamma \\ \hline 0 \end{array}\right] ,$$

where $\gamma = \pm \|v\|_2$, then pre-multiplying (24) by P leaves the upper triangular matrix R unchanged and the new factors of $\bar{A}$ are

$$\bar{R} = \left[\begin{array}{c|c} R & u \\ \hline 0 & \gamma \end{array}\right] \text{ and } \bar{Q} = PQ .$$

This method represents jutt a column-wise recursive definition of the QR factorization of A .

When Q is not stored or unavailable, the vector u can be found by solving the system

$$R^T u = A^T a .$$

The scalar γ is then given by the relation

$$\gamma^2 = \|a\|_2^2 - \|u\|_2^2$$

Rounding errors could cause this method to fail, however, if the new column a is nearly dependent on the columns of A . In fact if R is built up by a sequence of these modifications, in which the columns of A are added one by one, the process is exactly that of computing the product $B = A^T A$ and finding the Cholesky factorization

$$B = R^T R .$$

It is well known that this is numerically less satisfactory than computing R using orthogonal matrices. In some applications the s-th column of Q is available even when Q is not and consequently γ can be computed more accurately from the relationship

$$\gamma = a^T q_s ,$$

where q_s is the s-th column of Q .

Some improvement in accuracy can also be obtained on machines which have the facility for performing the double-length accumulation of inner-products. In this case the i-th element of u is set to

$$u_i = \frac{1}{r_{ii}} \left\{ \sum_{j=1}^{n} a_{ij} a_j - \sum_{j=1}^{i-j} u_j r_{ij} \right\} ,$$

where the two inner products are formed as a single sum. Despite these improvements this is still numerically less satisfactory than the previous method where Q was available.

A further possibility of improving the method arises when one column is being deleted and another is being added. A new column replacing the deleted column is equivalent to a rank two change in $A^T A$ and can be performed by any one of the methods given in section 3. Even this is still not ideal, since the computation of the rank one vectors require the matrix vector product $A^T(a - \bar{a})$ where a is the column being added and $\bar{a}$ is the column being deleted.

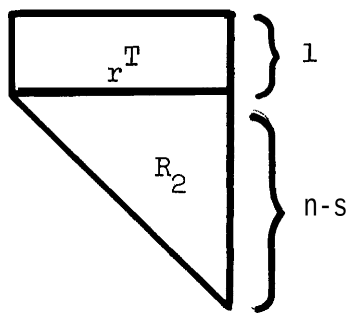
Finally we describe how to modify the factors when a column is deleted from A . It will be assumed that $\bar{A}$ is obtained from A by deleting the s-th column, which as usual will be denoted by a . Deleting the s-th column of R gives

$$Q\bar{A} = \begin{array}{c} \left[\begin{array}{c|c} R_1 & T_1 \\ \hline 0 & T_2 \\ \hline 0 & 0 \end{array} \right] \end{array} \begin{array}{l} \} \; s-1 \\ \} \; n-s+1 \\ \} \; m-n \end{array}$$

where R_1 is an $(s-1) \times (s-1)$ upper triangular matrix, T_1 is an $(s-1) \times (n-s)$ rectangular matrix and T_2 is an $(n-s+1) \times (n-s)$ upper Hessenberg matrix. For example, with $n=5, s=3$ and $m=7$ we have

$$\left[\begin{array}{c|c} R_1 & T_1 \\ \hline 0 & T_2 \\ \hline 0 & 0 \end{array}\right] \qquad \left[\begin{array}{cc|cc} X & x & x & x \\ 0 & x & x & x \\ \hline 0 & 0 & x & x \\ 0 & 0 & x & x \\ 0 & 0 & 0 & x \\ \hline 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{array}\right]$$

Let partition T_2 be of the form

$T_2 =$ 

We now choose an orthogonal matrix P which reduces T_2 to upper triangular form, using one of the methods described earlier. Thus

$$PT_2 = \left[\begin{array}{c} \bar{R}_2 \\ \hline 0 \end{array}\right] \begin{array}{l} \}\ n-s \\ \}\ 1 \end{array}$$

where P is of the form $P = P^{n-s}_{n-s+1} \cdot \cdot \cdot P^2_3 P^1_2$. The modified triangular factor for $\bar{A}$ is

$$\bar{R} = \left[\begin{array}{c|c} R_1 & T_1 \\ \hline 0 & \bar{R}_2 \\ \hline 0 & 0 \end{array}\right] \begin{array}{l} \} \ s-1 \\ \} \ n-s \\ \} \ m-n+1 \end{array}$$

If Q is to be updated also, the appropriate rows must be modified, thus:'

$$Q = \left[\begin{array}{c} Q_1 \\ \\ Q_3 \end{array}\right] \begin{array}{l} \} \ s-1 \\ \} \ n-s+1 \\ \} \ m-n \end{array} , \quad \bar{Q} = \left[\begin{array}{c} Q_1 \\ \hline PQ_2 \\ \hline Q_3 \end{array}\right] .$$

It is sometimes profitable to regard this computation from a different point of view. The partitions of T_2 satisfy the relation $\bar{R}_2^T\bar{R}_2 = R_2^T R_2 + rr^T$, and this is analogous to the equation $\bar{R}^T\bar{R} = R^T R + a a^T$ which holds when we add a row a^T to A . We conclude that <u>deleting a column</u> may be accomplished by essentially the same techniques as used for <u>adding a row</u>.

6. Conclusions

In this report we have presented a comprehensive set of' methods which can be used to modify nearly all the factorizations most frequently used in numerical linear algebra, It has not been our purpose to recommend a particular method where more than one exist. Although the amount of computation required for each is given, this will not be the only consideration since the relative efficiencies of the algorithms may alter when applied to particular problems. An example of this is when the Cholesky factors of a positive definite matrix are stored in product form. In this case the choice of algorithm is restricted to those that form the special matrices explicitly. The relative efficiency of methods Cl and C2 are consequently altered.

Bibliography

1. *Bartels, R. H., Golub, G. H., and Saunders, M. A., 1970, "Numerical techniques in mathematical programming," Nonlinear Programming, Academic Press, New York.

2. Bennett, J. M., 1965, "Triangular factors of modified matrices," Numerische Mathematik 7, pp. 217-221.

3. Fadeev, D. K., Kublanovskaya, V. N., and Fadeeva, V. N., 1968, "Sur les systemes lineaires algebriques de matrices rectangulaires et malconditionnées," Programmation en Mathematiques Numeriques (Editions du Centre National de la Recherche Scientific, Paris VII).

4. *Gentleman, W. M., 1972, "Least squares computations by Givens transformations without square roots," Department of Applied Analysis and--Computer Science Report CSRR-2062, University of Waterloo, Waterloo, Ontario.

5. *Gill, P. E., and Murray, W., 1970, "A numerically stable form of the simplex algorithm," National Physical Laboratory, DNAM Report Number 87, Teddington, England.

6. *Gill, P. E., and Murray, W., 1972, "Quasi-Newton methods for unconstrained optimization," J. Inst. Maths. Applics.9, pp. 91-108.

7. *Gill, P. E., and Murray, W., 1972, "Two methods for the solution of linearly constrained and unconstrained optimization problems," National Physical Laboratory, DNAC Report Number 25, Tedding-ton, England.

8. *Golub, G. H., and Saunders, M. A., 1970, "Linear least squares and quadratic programming," Integer and Nonlinear Programming, North-Holland Publishing Co., Amsterdam.

9. *Golub, G. H., and Styan, P. H., 1971, "Numerical computations for univariate linear models," Computer Science Department Report Number CS-236-71, Stanford University, Stanford, California.

10. Hanson, R. J., and Lawson, C. L., 1969, "Extensions and applications of the Householder algorithm for solving linear least squares problems," Maths. Comp. 23, pp. 787-812.

11. Martin, R. S., Peters, G., and Wilkinson, J. H., 1971, "The QR algorithm for real Hessenberg matrices," pp. 359-371 in: Handbook for automatic computation, Volume 2, Eds. J. H. Wilkinson and C. Reinsch, Springer-Verlag.

12. *Murray, W., 1971, "An algorithm for indefinite quadratic programming," National Physical Laboratory, DNAC Report Number 1, Teddington, England.

13. Peters, G., and Wilkinson, J. H., 1970, "The least squares problem and pseudo-inverses," Comp. J. 13, pp. 309-316.

14. *Saunders, M. A., 1972, "Large-scale linear programming using the Cholesky factorization," Computer Science Department Report Number CS-72-252, Stanford University, Stanford, California.

15. *Saunders, M. A., 1972, "Product form of the Cholesky factorization for large-scale linear programming," Computer Science Department Report Number CS-72-301, Stanford University, Stanford, California.

16. Wilkinson, J. H., 1965, "The algebraic eigenvalue problem," Oxford University Press, London.