LAPACK Working Note

Generalized QR Factorization and its Applications *

(Work in Progress)

E. Anderson, Z. Bai and J. Dongarra

December 9, 1991

August 9, 1994

Abstract

The purpose of this note is to re-introduce the generalized QR factorization with or without pivoting of two matrices A and B having the same number of rows, and whenever B is square and nonsingular, the factorization implicitly gives the orthogonal factorization with or without pivoting of $B^{-1}A$. The GQR factorization was introduced early by Hammarling[6] and Paige[9]. But from the general-purpose software development point of view, we proposed the different factorization forms. In addition to the factorization forms and implementation details, we show the applications of GQR factorization in solving the linear equality constraint least square problem, generalized linear model. It is intended to show the possible usage of LAPACK codes for solving a class of generalized least square problems who arise from optimization and statistics on high-performance machines.

1 Introduction

QR factorization of an n by m matrix A assumes the form

$$A = QR$$

where Q is an n by n orthogonal matrix, $R = Q^{\mathrm{T}}A$ is zero below its diagonal. If $n \geq m$, then $Q^{\mathrm{T}}A$ can be written in the form

$$Q^{\mathrm{T}}A = \begin{bmatrix} R \\ 0 \end{bmatrix}$$

where R is an n by n upper triangular. If $n < m$, then the QR factorization of A assumes the form

$$Q^{\mathrm{T}}A = \begin{bmatrix} R & S \end{bmatrix}$$

where R is an n by n upper triangular matrix. However, in practical applications, it is more convenient to represent the factorization in this case as

$$A = \begin{bmatrix} 0 & R \end{bmatrix} Q,$$

*This work was supported in part by NSF grant ASC-xxxx

which is known as the RQ factorization. As the variants of the QR and RQ factorization of matrix A, we also have QL and LQ factorization, which are orthogonal-lower triangular and lower triangular-orthogonal factorization, respectively. Moreover, it is well-known that the orthogonal factors of A provide information about its column and row spaces [4].

A column pivoting option in the QR factorization allows the user to detect dependencies among the columns of matrix A. If A has rank k, then there are orthogonal matrix Q and a permutation matrix P such that

$$Q^{\mathrm{T}}AP = \begin{array}{c} \left[\begin{array}{cc} R_{11} & R_{12} \\ 0 & 0 \end{array}\right] \begin{array}{l} k \\ n-k \end{array} \\ \begin{array}{cc} k & m-k \end{array} \end{array}$$

where R_{11} is a k by k upper triangular and nonsingular[4].

Householder transformation matrix or Givens rotation matrix provide numerical stable numerical methods to compute these factorizations with or without pivoting. The software for computing the QR factorization on sequential machines is available from public linear algebra library LINPACK[7]. Redesigned codes in block algorithm fashion that are better suited for today's high-performance architectures can be found in LAPACK.

The terminology *generalized QR factorizations* (GQR factorization), which has been introduced by Hammarling[6] and Paige[9], is to refer to orthogonal transformations that apply to n by m matrix A and n by p matrix B to transform them to triangular forms, respectively, but which corresponds to the QR factorization of $B^{-1}A$ in the case whenever B is square and nonsingular. For example, if $n \geq m$, $n \leq p$, then the GQR factorization of A and B assumes the form

$$Q^{\mathrm{T}}A = \left[\begin{array}{c} R \\ 0 \end{array}\right], \quad Q^{\mathrm{T}}BV = \left[\begin{array}{cc} 0 & S \end{array}\right],$$

where Q is an n by n orthogonal matrix, V is a p by p upper triangular matrix, R is a m by m upper triangular, S is a p by p upper triangular. If B is square and nonsingular, then the QR factorization of $B^{-1}A$ is given by

$$V^{\mathrm{T}}(B^{-1}A) = \left[\begin{array}{c} T \\ 0 \end{array}\right] = S^{-1}\left[\begin{array}{c} R \\ 0 \end{array}\right],$$

i.e., the upper triangular part T of QR factorization can be determined by solving the triangular matrix equation

$$ST = R.$$

The advantage of this implicit determination of the QR factorization of $B^{-1}A$ is obvious. We avoid the possible numerical difficulties to form B^{-1} and $B^{-1}A$.

As the powerful tool of QR factorization in least square and related linear regression problems, as examples, we shall show that the GQR factorization can been used to solve linear equality constrained least square problem

$$\min_{Bx=d} \|Ax - b\|,$$

where A and B are m by n and p by m matrices, respectively, and generalized linear regression model

$$\min_{x,u} u^{\mathrm{T}}u, \text{ subject to } b = Ax + Bu$$

where A is an n by m matrix, B is an n by p matrix. Indeed, the QR factorization approachs have been used for solving these problems, see Lawson and Hanson [8], Paige[10]. We will see that the GQR factorization of A and B provides an uniform approach to these problems. The benefit of this approach is threefold. First, it uses a single GQR factorization concept to solve these problems directly. Second, from software development point of view, it allows us to develop one subroutine that can be used for solving these problems. Third, as the QR factorization provides important information of conditioning of linear least square, classical linear regression model, we will show that the GQR factorization is the same. The condition numbers of these problems can be exploited from the triangular factors of the factorization.

The principle concepts about the GQR factorization discussed in this note have been presented in Paige's work on GQR[10]. However, from general-purpose software development point of view, we will take a different approach for the GQR factorization. The definition of the GQR factorization is different from the one presented in Paige's paper. As a guideline of the development of the GQR factorization for LAPACK library, in this note, we consider the different possible cases of the factorizations and practical implementation of the factorizations.

The outline of this LAPACK working note is as follows: In next two sections, we shall show how to use existing QR factorization and its variants to construct the GQR factorization with or without pivoting strategies of two matrices A and B having the same number of rows. The implementation details of the different factorizations are discussed in section 4. Then we show the applications of the GQR factorization in solving the linear equality constrained least square problem, generalized linear model problem, and estimating the conditioning of these problems.

Notations: ....

2 Generalized QR Factorization

In this section, we first introduce the GQR factorization of n by m matrix A and n by p matrix B when $n \geq m$, the most frequently occurring case. Then for the case $n < m$, we introduce the GRQ factorization of A and B.

GQR factorization. *Let A be an n by m matrix, B be an n by p matrix, $n \geq m$, then there are orthogonal matrices $Q(n \times n)$ and $V(p \times p)$ such that*

$$Q^{\mathrm{T}} A = R, \qquad Q^{\mathrm{T}} B V = S \tag{1}$$

assumes one of the following forms:
if $n \leq p$,

$$R = \underset{m}{\begin{bmatrix} R_{11} \\ 0 \end{bmatrix}} \begin{matrix} m \\ n-m \end{matrix} \quad , \qquad S = \underset{\begin{matrix} p-n & n \end{matrix}}{\begin{bmatrix} 0 & S_{11} \end{bmatrix}} n \quad ,$$

where m by m matrix R_{11} and n by n matrix S_{11} are upper triangular, and if $n > p$,

$$R = \underset{m}{\begin{bmatrix} R_{11} \\ 0 \end{bmatrix}} \begin{matrix} m \\ n-m \end{matrix} \quad , \qquad S = \underset{p}{\begin{bmatrix} S_{11} \\ S_{21} \end{bmatrix}} \begin{matrix} n-p \\ p \end{matrix} \quad ,$$

where m by m matrix R_{11} and p by p matrix S_{21} are upper triangular.

Proof: The proof is constructive. By the QR factorization of A we have

$$Q^{\mathrm{T}}A = \underset{m}{\begin{bmatrix} R_{11} \\ 0 \end{bmatrix}} \begin{matrix} m \\ n-m \end{matrix} \quad .$$

Let Q^{T} premultiply on B, then the desired factorizations follow upon the RQ factorization of $Q^{\mathrm{T}}B$; if $n \le p$,

$$(Q^{\mathrm{T}}B)V = \begin{bmatrix} \underset{p-n}{0} & \underset{n}{S_{11}} \end{bmatrix} n \quad .$$

otherwise, the RQ factorization of $B^{-1}A$ is of the form

$$(Q^{\mathrm{T}}B)V = \underset{p}{\begin{bmatrix} S_{11} \\ S_{21} \end{bmatrix}} \begin{matrix} n-p \\ p \end{matrix} \quad .$$

$\square$.

Occasionally, one wishes to compute the QR factorizations of $B^{-1}A$, for example, to solve weighted least square problem

$$\min_x \|B^{-1}(Ax - b)\|.$$

To avoid forming B^{-1} and $B^{-1}A$, we note that the GQR factorization (1) of A and B implicitly gives the QR factorization $B^{-1}A$:

$$V^{\mathrm{T}}(B^{-1}A) = \begin{bmatrix} T \\ 0 \end{bmatrix} = S_{11}^{-1} \begin{bmatrix} R_{11} \\ 0 \end{bmatrix},$$

i.e, the upper triangular part T of the QR factorization of $B^{-1}A$ can be determined by

$$S_{11}T = R_{11}.$$

Hence, the possible numerical difficulties to use, explicitly or implicitly, the QR factorization of $B^{-1}A$ is confined to the condition number of S_{11}.

Moreover, if we partition $V = \begin{bmatrix} V_1 & V_2 \end{bmatrix}$ where V_1 has m columns, then

$$B^{-1}A = V_1(S_{11}^{-1}R_{11}).$$

This shows that if A is of rank m, the columns of V_1 form an orthonormal basis for the space spanned by the columns of $B^{-1}A$. The matrix $V_1V_1^{\mathrm{T}}$ is the orthogonal projection on the column space of $B^{-1}A$.

When A is n by m matrix with $n < m$, although it still can be presented the similar GQR factorization form of A and B, it is more useful in applications to represent the factorization as the following:

GRQ factorization: *Let A be an n by m matrix, B be an n by p matrix, $n < m$, then there are orthogonal matrices $Q(n \times n)$ and $U(m \times m)$ such that*

$$Q^{\mathrm{T}}AU = R, \qquad Q^{\mathrm{T}}B = S \tag{2}$$

assumes one of the following forms
if $n \le p$

$$R = \underset{\quad m-n \quad\; n}{\left[\begin{array}{cc} 0 & R_{11} \end{array}\right]} n, \qquad S = \underset{\quad n \quad\; p-n}{\left[\begin{array}{cc} S_{11} & S_{12} \end{array}\right]} n,$$

and if $n > p$

$$R = \underset{\quad m-n \quad\; n}{\left[\begin{array}{cc} 0 & R_{11} \end{array}\right]} n, \qquad S = \underset{p}{\left[\begin{array}{c} S_{11} \\ 0 \end{array}\right]} \begin{array}{l} p \\ n-p \end{array},$$

where n *by* n *matrix* R_{11} *and* $\min(n,p)$ *by* $\min(n,p)$ *matrix* S_{11} *are upper triangular.*

Proof : The proof is similar to the proof of the GQR factorization. In short, one first does the QR factorization of B: $B = QS$, then follows by the RQ factorization of $Q^{\mathrm{T}}A$. □

From the GRQ factorization form of A and B, we see that if B is square and nonsingular, then the RQ factorization of $B^{-1}A$ is given by

$$(B^{-1}A)U = \left[\begin{array}{cc} 0 & T \end{array}\right] = S_{11}^{-1}\left[\begin{array}{cc} 0 & R_{11} \end{array}\right].$$

With the decomposition forms (1) and (2), we present the formal definition for the generalized QR factorizations.

Definition: For n by m matrix A, n by p matrix B, we call (1) as *the generalized QR factorization* (in short the GQR factorization) of A and B, (2) as *the generalized RQ factorization* (in short the GRQ factorization) of A and B.

Discussion: the other possible generalized QR factorization forms. For example, the QR factorizaiton of $B^{\mathrm{T}}A$. — will write later.

3 Generalized QR Factorization with Pivoting

The previous section introduces the generalized QR factorization. As in the QR factorization of a matrix, we can also incorporate the pivoting technique into the GQR factorization to deal with the rank deficient case of the factorization.

GQR factorization with column pivoting: *Let* A *be an* n *by* m *matrix,* B *be an* n *by* p *matrix, then there are orthogonal matrices* $Q(n \times n)$, $V(p \times p)$, *and* n *by* n *permutation matrix* P *such that*

$$Q^{\mathrm{T}}AP = R, \quad Q^{\mathrm{T}}BV = S \tag{3}$$

assumes one of the following forms:
if $n \le p$

$$R = \underset{\quad q \quad\; m-q}{\left[\begin{array}{cc} R_{11} & R_{12} \\ 0 & 0 \\ 0 & 0 \end{array}\right]} \begin{array}{l} q \\ k \\ n-q-k \end{array}, \qquad S = \underset{p-n \quad q \quad n-q}{\left[\begin{array}{ccc} 0 & S_{11} & S_{12} \\ 0 & 0 & S_{22} \\ 0 & 0 & 0 \end{array}\right]} \begin{array}{l} q \\ k \\ n-q-k \end{array},$$

where R_{11} and S_{22}, if which exists, are full row rank upper trapezoidal matrices. q by q matrix S_{11} is upper triangular. And if $n > p$,

$$R = \begin{array}{c} \left[\begin{array}{cc} R_{11} & R_{12} \\ 0 & 0 \\ 0 & 0 \end{array}\right] \begin{array}{l} q \\ k \\ n-q-k \end{array} \\ \begin{array}{cc} q & m-q \end{array} \end{array}, \qquad S = \begin{array}{c} \left[\begin{array}{cc} S_{11} & S_{12} \\ 0 & S_{22} \\ 0 & 0 \end{array}\right] \begin{array}{l} q \\ k \\ n-q-k \end{array} \\ \begin{array}{cc} p-n+q & n-q \end{array} \end{array},$$

where q by q matrix R_{11} are nonsingular and upper triangular, and k by $n-q$ matrix S_{22} is full row rank upper trapezoidal[1]*.*

Proof : The proof is also constructive. By the QR factorization with pivoting of A, we have

$$Q_1^{\mathrm{T}} AP = \begin{array}{c} \left[\begin{array}{cc} R_{11} & R_{12} \\ 0 & 0 \end{array}\right] \begin{array}{l} q \\ n-q \end{array} \\ \begin{array}{cc} q & m-q \end{array} \end{array}.$$

where $q = \mathrm{rank}(A)$. If $n \leq p$, let

$$(Q_1^{\mathrm{T}} B)V_1 = \begin{array}{c} \left[\begin{array}{ccc} 0 & S_{11} & \bar{S}_{12} \\ 0 & 0 & \bar{S}_{22} \end{array}\right] \begin{array}{l} q \\ n-q \end{array} \\ \begin{array}{ccc} p-n & q & n-q \end{array} \end{array}$$

be the RQ factorization of matrix $Q_1^{\mathrm{T}} B$. Then by the QR factorization with pivoting of submatrix $\bar{S}_{22}$, we have

$$Q_2^{\mathrm{T}} \bar{S}_{22} P_2 = \begin{array}{c} \left[\begin{array}{c} S_{22} \\ 0 \end{array}\right] \begin{array}{l} k \\ n-q-k \end{array} \\ n-q \end{array}.$$

The result for this case follows by setting $Q = Q_1 \left[\begin{array}{cc} I & 0 \\ 0 & Q_2 \end{array}\right]$, $V = V_1 \left[\begin{array}{cc} I & 0 \\ 0 & P_2 \end{array}\right]$.

If $n > p$, let Q_1^{T} premultiply on B, denote as

$$Q_1^{\mathrm{T}} B = \bar{B} = \begin{array}{c} \left[\begin{array}{c} \bar{S}_{11} \\ \bar{S}_{21} \end{array}\right] \begin{array}{l} q \\ n-q \end{array} \\ p \end{array}.$$

if $p \leq n-q$, then by the QR factorization with pivoting of $\bar{S}_{21}$, we have

$$Q_2^{\mathrm{T}} \bar{S}_{21} P_2 = \begin{array}{c} \left[\begin{array}{c} S_{21} \\ 0 \end{array}\right] \begin{array}{l} k \\ n-q-k \end{array} \\ p \end{array}$$

where $k = \mathrm{rank}(\bar{S}_{21})$. The desired factorization forms are obtained by setting $Q = Q_1 \left[\begin{array}{cc} I & 0 \\ 0 & Q_2 \end{array}\right]$ and $V = P_2$.

[1] if $p < n-q$, then set $p-n+q$ equal to zero, set $n-q$ equal p in the factorization form of S.

If $p > n - q$, by the RQ factorization of $\bar{B}$ we have

$$(Q_1^{\mathrm{T}} B)V = \begin{array}{c} \left[\begin{array}{c} \bar{S}_{11} \\ \bar{S}_{21} \end{array} \right] \begin{array}{l} n-p \\ p \end{array} \\ p \end{array} .$$

The conclusion for this case follows upon the QR factorization with pivoting of $n - q$ by $n - q$ bottom right corner submatrix of S_{21}. □

We note that in this case, if B is square and nonsingular, then the QR factorization with column pivoting of $B^{-1}A$ is given by

$$V^{\mathrm{T}}(B^{-1}A)P = S_{11}^{-1} \left[\begin{array}{c} R_{11} \\ 0 \end{array} \right].$$

Hence we present

Definition: For n by m matrix A, n by m matrix B, we call (3) as *the generalized QR factorization with column pivoting* (in short *the GQR with pivoting*) of matrices A and B.

4 Implementations

In this section, we shall discuss the algorithms and implementation details of GQR factorization and its variants. We will see that the implementations of all these factorizations can be easily kept in the block algorithm fashion as QR factorization does.

The "MATLAB" style notation $A(i:j, p:q)$ is used in this section to specify the ith to jth rows and pth to qth columns submatrix of matrix A.

As was mentioned in section 2, the existence proofs of the GQR factorization and its variants are constructive. They strongly depends on the regular QR factorization and its variants. At the beginning of this note we briefly reviewed the QR factorizations and its variants, the correspondent LAPACK subroutine names (in `typewriter font`) and its functions are listed in the followings:

`SGEQRF`: QR factorization of a n by m matrix.

`SGEQPF`: QR factorization of a n by m matrix with column pivoting.

`SGERQF`: RQ factorization of a n by m matrix.

`SGELQF`: LQ factorization of a n by m matrix.

All these subroutines are the implementations of the block algorithms. That is to say, instead of traditional algorithms to generate and apply Householder transformations one by one (BLAS 2 matrix-vector operations), the block algorithms aggragate a series (say a block with size b) of Householder transformations, represent in a block matrix form and apply them to reduce a block at the same time (BLAS 3 matrix-matrix operations). There is a common parameters called blocksize `NB` of these subroutines that allows user to choice to find the best blocksize one their machines. In particularly, if blocksize is one, then they are just traditional QR factorizations and its variants.

To call this routine, the input array `A` contains the n by m matrix. On the return of these subroutines, the upper (lower) triangle of the array contains the triangular part of the factorizations, the orthogonal matrix can be recovered from the elements below (above) the diagonal with another one dimensional array.

Turning to compute GQR factorization, we first present the GQR factorization of n by m matrix A and n by m matrix B with $n \geq m$, then the GRQ factorizations for the case $n \leq m$ and GQR factorization with column pivoting.

Algorithm 1 (GQR factorization, $n \geq m$)

1). QR factoriztion of A: $Q^{\mathrm{T}} A = R$ and $B = Q^{\mathrm{T}} B$.

2). RQ factorization of B: $BV^{\mathrm{T}} = S$.

To implement Algorithm 1, we note that the block transformations can be applied to B at the same time while A is reduced to upper triangular form block by block. At the end of step 1, the upper diagonal line part of A stores the factor R, the strictly lower diagonal line of A and another vector stores Householder vectors from which the orthogonal matrix Q can be recovered later if necessary. At step 2, we just need to implement the block RQ factorization on updated B.

When A is n by m with $n \leq m$, B is n by p matrix, the following algorithm computes the GRQ factorization.

Algorithm 2 (GRQ factorization, $n \leq m$)

1.) QR factorization of B: $Q^{\mathrm{T}} B = S$, and $A = Q^{\mathrm{T}} A$.

2.) RQ factorization of A: $A = RU$.

The remarks for algorithm 1 can be established similarly in this case. Finally, to treat with ill-conditioned or rank deficient cases, we can use the GQR factorization with column pivoting.

Algorithm 3 (GQR factorization with pivoting)

1). QR factorization with pivoting of A: $Q^{\mathrm{T}} AP = R$ and $B = Q^{\mathrm{T}} B$.

2). determine the rank q of A.

3). if $(n \leq p)$ then

    RQ factorization of B: $BV^{\mathrm{T}} = S$

    QR factorization with pivoting of $S(q+1:n, p-n+q+1:p)$

  else

    if $(p \leq n-q)$ then

      QR factorization with pivoting of $B(q+1:n, 1:p)$.

    else

      RQ factorization of B: $BV^{\mathrm{T}} = S$,

      QR factorization with pivoting of $S(q+1:n, q+1, p)$.

    endif

  endif

  Update the $q+1$ to n columns of Q correspondently.

To numerically determine the rank of A, according to the properties of the QR factorization with column pivoting, we can use the the following simple strategy:

$$
\begin{array}{l}
q = 0; \\
k = \min(n, m); \\
\text{for } \ i = 1, m \\
\qquad \text{if } \ (|r_{ii}| \leq tol * |r_{11}|) \quad q = q + 1
\end{array}
$$

where tol is a user given tolerance value, for example, tol can be chosed as machine precision. As a more careful method to determine the rank of a matrix, the so-called rank revealing QR factorization can also be used to replace the normal column pivoting QR scheme[4, 2].

The MATLAB codes of algorithm 1, 2 and 3 are in Appendix.

5 Applications

In the above sections, we introduced the GQR factorization with or without pivoting. In this section, as examples, we will show that the GQR factorization provide a simplier and more efficient way to solve the linear equality constrained linear least square problem and generalized linear regression problem, and to assess the conditioning of these problems. The results presented in this section show that the GQR factorization for solving these generalized optimization or linear regression problems is just as power as the QR factorization for solving least square and linear regression problems.

5.1 Linear Equality Constrained Linear Least Squares

Linear equality constrained linear least squares problem (Problem LSE) arises in constrained surface fitting, constrained optimizing, geodetic least-squares adjustment, and beamforming etc. The problem is stated as follows: find a n-vector x that solves

$$\min_{Bx=d} \|Ax - b\|, \tag{4}$$

where A is m by n matrix, $m \geq n$, B is p by n matrix, $p \leq n$, b is n column vector, d is p column vector. Clearly, the Problem LSE has a solution if and only if the equation $Bx = d$ is consistent. In addition, for simplicity, we will also assume that

$$\text{rank}(B) = p \tag{5}$$

and that the null spaces of A and B intersect only trivially:

$$\mathcal{N}(A) \cap \mathcal{N}(B) = \{0\} \Longleftrightarrow \text{rank}\left(\begin{bmatrix} A \\ B \end{bmatrix} \right) = n. \tag{6}$$

These conditions ensure that the problem LSE has a unique solution which we denote by x_{e}.

Several methods for solving the problem LSE are discussed in Lawson and Hanson[8, Chaps.20-22], Van Loan[12]. For large sparse matrices case, see Barlow *et al* [1]. The QR style approach is one of the most common approach. Now this approach can be more easily presented in terms of the GQR factorization of A and B.

By the GQR factoization of B^{T} and A^{T}, we know that there are orthogonal matrices Q and U

$$Q^{\mathrm{T}} A^{\mathrm{T}} U = \begin{array}{c} \left[\begin{array}{ccc} 0 & R_{11}^{\mathrm{T}} & R_{12}^{\mathrm{T}} \\ 0 & 0 & R_{22}^{\mathrm{T}} \end{array}\right] \begin{array}{l} p \\ n-p \end{array} \\ \begin{array}{ccc} m-n & p & n-p \end{array} \end{array}, \quad Q^{\mathrm{T}} B^{\mathrm{T}} = \begin{array}{c} \left[\begin{array}{c} S_{11}^{\mathrm{T}} \\ 0 \end{array}\right] \begin{array}{l} p \\ n-p \end{array} \\ p \end{array}$$

and moreover, from the assumptions (4) and (6), we know S_{11}^{T} and R_{22}^{T} are upper triangular and nonsingular. If we partition

$$Q = \left[\begin{array}{cc} Q_1 & Q_2 \end{array}\right], \quad U = \left[\begin{array}{ccc} U_1 & U_2 & U_3 \end{array}\right],$$

where Q_1 has p columns, U_1 has $m-n$ columns, and U_2 has p columns, and set

$$y = Q^{\mathrm{T}} x = \left[\begin{array}{c} y_1 \\ y_2 \end{array}\right] \begin{array}{l} p \\ n-p \end{array}, \quad c = U^{\mathrm{T}} b = \left[\begin{array}{c} c_1 \\ c_2 \\ c_3 \end{array}\right] \begin{array}{l} m-n \\ p \\ n-p \end{array},$$

where $y_i = Q_i^{\mathrm{T}} x, i = 1, 2$, $c_i = U_i^{\mathrm{T}} b, i = 1, 2, 3$. The Problem LSE is then transformed to

$$\min \left\| \left[\begin{array}{cc} 0 & 0 \\ R_{11} & 0 \\ R_{12} & R_{22} \end{array}\right] \left[\begin{array}{c} y_1 \\ y_2 \end{array}\right] - \left[\begin{array}{c} c_1 \\ c_2 \\ c_3 \end{array}\right] \right\|$$

subject to

$$\left[\begin{array}{cc} S_{11} & 0 \end{array}\right] \left[\begin{array}{c} y_1 \\ y_2 \end{array}\right] = d.$$

Hence we can compute y_1 from constrained equality by solving the triangular system

$$S_{11} y_1 = d.$$

Then the Problem LSE is truncated to the ordinary linear least square problem

$$\min_{y_2} \| R_{22} y_2 - (c_3 - R_{12} y_1) \|$$

Since R_{22} is nonsingular and lower triangular, y_2 is given by

$$y_2 = R_{22}^{-1}(c_3 - R_{12} y_1),$$

a triangular solver. The solution of Problem LSE is then given by

$$x_e = Qy = Q_1 y_1 + Q_2 y_2$$

or in a more straightforward form

$$x_e = Q_2 R_{22}^{-1} U_3^{\mathrm{T}} b + Q \left[\begin{array}{c} I \\ -R_{22}^{-1} R_{12} \end{array}\right] S_{11}^{-1} d$$

and the residual sum of squares $\rho^2 = \|r_e\|^2 = \|Ax_e - b\|^2$ is given by

$$\rho^2 = \|c_1\|^2 + \|R_{11} y_1 - c_2\|^2.$$

The Sensitivity of Problem LSE: The condition numbers of A and B were introduced by Elden [3] to assess the perturbation behavior of the Problem LSE. Specifically, let E be an error matrix of A, F be an error matrix of B, e and f be errors of b and d, respectively. We assume that $B+F$ also has full row rank and $\mathcal{N}(A+E)\cap\mathcal{N}(B+F)=\{0\}$. Let $\bar{x}_e$ be the solution of the same problem with A, B, b, d replaced by $A+E$, $B+F$, $b+e$, $d+f$, respectively. Elden uses the numbers

$$\kappa_B(A) = \|A\|\,\|(AG)^+\|, \quad \kappa_A(B) = \|B\|\,\|B^+_{IA}\|$$

to measure the sensitivity of the problem LSE, where

$$G = I - B^+B, \qquad B^+_{IA} = (I-(AG)^+A)B^+.$$

His asymptotic perturbation bound, modified slightly here, can be presented as following, where the bound has a $+O(\epsilon^2)$ term (i.e., the higher order of the perturbation matrices E and F) has not been written.

Problem LSE Perturbation Bound:

$$\frac{\|x_e-\bar{x}_e\|}{\|x_e\|} \le \kappa_B(A)\left(\frac{\|E\|}{\|A\|}+\nu_e\right) + \kappa_A(B)\left(\frac{\|F\|}{\|B\|}+\gamma_e\right) + \kappa_B^2(A)\left(\frac{\|E\|}{\|A\|}+\kappa_A(B)\frac{\|F\|}{\|B\|}\right)\rho_e$$

where

$$\nu_e = \frac{\|e\|}{\|A\|\,\|x_e\|}, \quad \gamma_e = \frac{\|f\|}{\|B\|\,\|x_e\|},$$

$$\rho_e = \frac{\|r_e\|}{\|A\|\,\|x_e\|}, \quad r_e = Ax_e - b.$$

The interpretation of this result is that the sensitivity of $\bar{x}_e$ is measured by $\kappa_B(A)$ and $\kappa_A(B)$ if the residual r_e is zero or relatively small, and otherwise by $\kappa_B^2(A)(\kappa_A(B)+1)$.

We note that if matrix B is zero (hence $F=0$), then the problem LSE is just the ordinary linear least square problem. The perturbation bound for the problem LSE is then reduced to

$$\frac{\|x_e-\bar{x}_e\|}{\|x_e\|} \le \kappa(A)\left(\frac{\|E\|}{\|A\|}+\frac{\|e\|}{\|A\|\,\|x_e\|}\right) + \kappa^2(A)\frac{\|E\|}{\|A\|}\frac{\|r_e\|}{\|A\|\,\|x_e\|} + O(\epsilon^2)$$

where $\kappa_B(A) = \kappa(A) = \|A\|\,\|A^+\|$. This is just the perturbation bound of the linear least square problem eariest othe btained by Golub and Wilkinson(1966) [4].

Esitmation of the condition numbers: The condition numbers of Problem LSE $\kappa_B(A)$ and $\kappa_A(B)$ involve B^+, B^+B, $(AG)^+$ etc, and computing these matrices can be relatively expensive. Fortunately, it is possible to compute inexpensive estimates of $\kappa_B(A)$ and $\kappa_A(B)$ without forming B^+, B^+B or $(AG)^+$. This can be done using a method of Hager (1984) and Higham (1988) [5] that computes a lower bound for $\|B\|_\infty$, where B is a matrix, given a mean for evluating matrix-vector products Bu and $B^{\mathrm{T}}u$. Typically, 4 or 5 products are required, and the lower bound is almost always within a factor 3 of $\|B\|_\infty$. The corresponding subroutine, named as `SLACON`, is available in LAPACK. To estimate $\kappa_B(A)$ and $\kappa_A(B)$, we apparently need to estimate vector norms $\|Kz\|_\infty$, where $K=(AG)^+$, or $K=B^+_{IA}$ and $z\ge 0$ is a vector that is readily computed. Given GQR factorization of A and B, after tedious computations, we have

$$(AG)^+z = Q_2R_{22}^{-1}U_3^{\mathrm{T}}z$$

$$B_{IA}^{+}z = Q \begin{bmatrix} I \\ -R_{22}^{-1}R_{12} \end{bmatrix} S_{11}^{-1}z$$

where we do not form R_{22}^{-1} or S_{11}^{-1} but rather solve the triangular system and do matrix-vector operations.

5.2 Solve Generalized Linear Model

The generalized linear regression model (Problem GLM) can be written as

$$b = Ax + w, \tag{7}$$

where the vector b, the n by m matrix A, n by n symmetric nonnegative definite matrix W are known, while w is a random error with mean 0 and covariance $\sigma^2 W$. The problem is that of estimating the unknown parameters x on the basis of observation b. If W has rank p, then W has a factorization

$$W = BB^{\mathrm{T}},$$

where the n by p matrix B has linearly independent columns, for example, the Cholesky factorization of W could be carried out to get B. In some practical problems, the matrix B might be available directly. Form numerical computational reasons it is preferable to use B rather than W. Since W could be ill-conditioned in the sene discussed by Golub and Styan (1973). Then the condition of B will be much better. Thus we replace (7) by

$$b = Ax + Bu, \tag{8}$$

where A is a n by m matrix, B is n by p matrix having full column rank, while u is a random error with mean 0 and covariance $\sigma^2 I$. Then the estimator of x in (8) is the solution to the following algebraic generalized linear least squares problem:

$$\min_{x,u} u^{\mathrm{T}}u \quad \text{subject to} \quad b = Ax + Bu \tag{9}$$

For convenience, we assume that $n \geq m$, $n \geq p$, and $\text{rank}(B) = p$, the most frequently ocurring case. When $B = I$, (9) is just an ordinary linear regression problem.

The Problem GLM can be formulated as the problem LSE which is discussed in the last subsection:

$$\min \left\| \begin{bmatrix} 0 & I \end{bmatrix} \begin{bmatrix} x \\ u \end{bmatrix} \right\| \quad \text{subject to} \quad \begin{bmatrix} A & B \end{bmatrix} \begin{bmatrix} x \\ u \end{bmatrix} = b.$$

Hence, it is easy to see that the problem GLM has a unique solution under the following conditions:

$$\text{rank}\left(\begin{bmatrix} A & B \end{bmatrix} \right) = n \quad \text{and} \quad \text{rank}(A) = m.$$

To avoid the overhead of computation cost, storage and possible numerical difficulty of the combination of the matrices A and B, it is not suggested to solve the problem GLM directly by the method of the problem LSE. Early 1979, Paige [10] proposed two steps QR decomposition approach to the problem GLM to treat with A and B separately. Now, his approach can be simplified with GQR factorization terminology. For sake of arguments, we assume that the Problem GLM has unique solutions of x and u which we denote by x_e and u_e.

By the GQR factorization with pivoting of A and B, under the uniqueness assumption, we have orthogonal matrices $Q(n \times n)$, $V(p \times p)$, and m by m permutation matrix P such that

$$Q^{\mathrm{T}} AP = \underset{\begin{matrix} q & m-q \end{matrix}}{\begin{bmatrix} R_{11} & R_{12} \\ 0 & 0 \end{bmatrix}} \begin{matrix} q \\ n-q \end{matrix}, \qquad Q^{\mathrm{T}} BV = \underset{\begin{matrix} p-n+q & n-q \end{matrix}}{\begin{bmatrix} S_{11} & S_{12} \\ 0 & S_{22} \end{bmatrix}} \begin{matrix} q \\ n-q \end{matrix}$$

where q by q matrix R_{11}, $n-q$ by $n-q$ matrix S_{22} are nonsingular upper triangular. If we partition

$$Q = \begin{bmatrix} Q_1 & Q_2 \end{bmatrix}, \quad V = \begin{bmatrix} V_1 & V_2 \end{bmatrix}, \quad P = \begin{bmatrix} P_1 & P_2 \end{bmatrix},$$

where Q_1 has q columns, V_2 has $n-q$ columns, P_1 has q columns, and set

$$c = Q^{\mathrm{T}} b \equiv \begin{bmatrix} c_1 \\ c_2 \end{bmatrix}, \quad v = V^{\mathrm{T}} u \equiv \begin{bmatrix} v_1 \\ v_2 \end{bmatrix}, \quad y = P^{\mathrm{T}} x \equiv \begin{bmatrix} y_1 \\ y_2 \end{bmatrix}$$

i.e., $c_i = Q_i^{\mathrm{T}} b$, $v_i = V_i^{\mathrm{T}} u$, $y_i = P_i^{\mathrm{T}} x$, $i = 1, 2$, then the problem GLM is transformed to

$$\begin{bmatrix} c_1 \\ c_2 \end{bmatrix} = \begin{bmatrix} R_{11} & R_{12} \\ 0 & 0 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} + \begin{bmatrix} S_{11} & S_{12} \\ 0 & S_{22} \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \end{bmatrix}. \tag{10}$$

Hence v_2 can be determined from the "bottom" equation by solving a triangular system:

$$v_2 = S_{22}^{-1} c_2.$$

Then from the "top" equation, we have

$$c_1 = R_{11} y_1 + R_{12} y_2 + S_{11} v_1 + S_{12} v_2.$$

It is obvious that the rest of solutions are given by

$$v_1 = 0, \qquad y_2 = 0, \qquad y_1 = R_{11}^{-1}(c_1 - S_{12} v_2).$$

The solution of the original problem GLM can be written in the forms:

$$x_e = P_1 y_1; \qquad u_e = V_2 v_2$$

or in a more straightforward form

$$x_e = P_1 R_{11}^{-1}(Q_1^{\mathrm{T}} - S_{12} S_{22}^{-1} Q_2^{\mathrm{T}}) b$$

and

$$u_e = V_2 S_{22}^{-1} Q_2^{\mathrm{T}} b.$$

The Sensitivity of the Problem GLM: Regarding to the sensitivity of the problem solutions to perturbations, we shall consider the effects of the perturbations in the vector b and perturbations in the matrices A and B. Let the perturbed problem GLM be defined as

$$\min_{\bar{x}, \bar{u}} \bar{u}^{\mathrm{T}} \bar{u}, \quad \text{subject to} \quad b + e = (A + E)\bar{x} + (B + F)\bar{u},$$

and also assume that the perturbated system has unique solution, i.e., $\mathrm{rank}(A+E, B+F) = n$, and $\mathrm{rank}(A+E) = m$. The solutions are denoted as $\bar{x}_e$ and $\bar{u}_e$. Then we have the following bounds on the relative error in $\bar{x}_e$ and $\bar{u}_e$ due to the perturbations of b, A and B, where each asymptotic bound has a $+O(\epsilon^2)$ term (i.e., the higher order of the perturbation data E, F etc) which is not written.

Problem GLM Perturbation Bounds:

$$\frac{\|\bar{x}_e - x_e\|}{\|x_e\|} \le \kappa_B(A)\left(\frac{\|E\|}{\|A\|} + \frac{\|e\|}{\|A\|\,\|x_e\|}\right) + \kappa_B(A)\left(\kappa_B(A)\frac{\|E\|}{\|A\|} + \frac{\|F_1\|}{\|B\|}\right)\frac{\|B\|^2\,\|p\|}{\|A\|\,\|x_e\|}, \quad (11)$$

and

$$\frac{\|\bar{u}_e - u_e\|}{\|b\|} \le \kappa_B(A)\frac{\|E\|}{\|A\|}\frac{\|B\|\,\|p\|}{\|b\|} + \frac{\kappa_A^2(B)}{\|B\|}\left(\|E\|\,\frac{\|x_e\|}{\|b\|} + \frac{\|e\|}{\|b\|} + \|F_1\|\,\frac{\|p\|}{\|b\|}\right) + \|F\|\,\frac{\|p\|}{\|b\|}, \quad (12)$$

where

$$\kappa_B(A) = \|A\|\,\|A_{IB}^+\|, \qquad \kappa_A(B) = \|B\|\,\|(GB)^+\|,$$

$G = I - AA^+$, $A_{IB}^+ = A^+(I - B(GB)^+)$ and $p = (GB(GB)^{\mathrm{T}})^+ b$, $F_1 = BF^{\mathrm{T}} + F^{\mathrm{T}}B$.

The proof is going to be presented later.

The bounds are quite complicated. If we note that

$$\|B\|^2\|p\| \le \kappa_A^2(B)\|b\|.$$

then the bounds (11) and (12) can be simplified a little bit. We see that the sensitivities of $\bar{x}_e$ and $\bar{u}_e$ depend on the $\kappa_B(A)$ and $\kappa_A(B)$. For this reason, $\kappa_B(A)$ and $\kappa_A(B)$ are defined as the *condition numbers* of Problem GLM. They can be used to predict the effects of errors in the regression variables on regression coefficients.

As special case, we note that if $B = I$, then the problem GLM is reduced to the classical linear regression problem. u_e is just the residual vector, $u_e = p = r_e = b - Ax_e = (I - AA^+)b$, $\bar{u} = \bar{r} = (b+e) - (A+E)\bar{x}$, $F = 0$, and

$$\kappa_B(A) = \kappa(A) = \|A\|\,\|A^+\|, \qquad \kappa_A(B) = 1.$$

Hence we have

$$\frac{\|\bar{x}_e - x_e\|}{\|x\|} \le \kappa(A)\left(\frac{\|E\|}{\|A\|} + \frac{\|e\|}{\|A\|\,\|x_e\|}\right) + \kappa^2(A)\frac{\|E\|}{\|A\|}\frac{\|r_e\|}{\|A\|\,\|x_e\|} + O(\epsilon^2)$$

and

$$\frac{\|\bar{r}_e - r_e\|}{\|b\|} \le \kappa(A)\frac{\|E\|}{\|A\|}\frac{\|r_e\|}{\|b\|} + \|E\|\,\frac{\|x_e\|}{\|b\|} + \frac{\|e\|}{\|b\|} + O(\epsilon^2)$$

These are the well-known perturbation results for the solution and residual of ordinary linear least regression problem [11, 4].

Estimation of condition numbers: Concerning about the estimating of the condition numbers $\kappa_B(A)$ and $\kappa_A(B)$ of the Problem GLM, we still can use the Hager and Higham's method without forming expensive A^+ or $(GB)^+$. To use their method, the required vector norms $\|Kz\|_\infty$ can be computed by GQR factorization of A and B, where $K = (GB)^+$, or $K = A_{IB}^+$ and $z \ge 0$ is a vector that is readily computed. After tedious computations, we have

$$(GB)^+ z = V_2 S_{22}^{-1} Q_2^{\mathrm{T}} z$$

and

$$A_{IB}^{+}z = P_1 R_{11}^{-1}(Q_1^{\mathrm{T}}z - S_{12}S_{22}^{-1}Q_2^{\mathrm{T}}z).$$

Hence, we can just use triangular system solver and matrix-vector operations to give the estimation of condition numbers of the Problem GLM.

5.3 Other possible applications:

Will briefly mention the possible use of GQR in

Preprocessing step of computing the generalized singular value decomposition.

Structural equations:

$$f = A^{\mathrm{T}}t, \quad e = BB^{\mathrm{T}}t, \quad e = -Ad,$$

where f is given, and we wish to find d.

A Matlab Codes

In this appendix, we list the MATLAB codes for computing the GQR factorizations with and without pivoting.

```
        function [Q,V,R,S] = gqr(A,B)
%
%  GQR -----------------------------------------------------------
%
%       Compute the generalized QR factorization of A(nxm) and
%       B(nxp):
%
%               A = Q*R;        B = Q*S*V;
%
%       where n >= m, Q(nxn), V(pxp) are orthogonal matrices,
%       R and S assumes one of the forms:
%       if n <= p
%
%               R = [ R11 ] m         S = [ 0  S11 ] n
%                   [  0  ] n-m            p-n  n
%                      m
%       where m by m matrix R11 and n by n matrix S11 are upper
%       triangular, and
%       if n > p
%
%               R = [ R11 ] m         S = [ S11 ] n-p
%                   [  0  ] n-m           [ S21 ] p
%                      m                     p
%       where m by m matrix R11 and p by p matrix S21 are upper
%       triangular.
%
%  ---------------------------------------------------------------
%
        [n,m] = size(A);
        [n1,p] = size(B);
%
        if n1 ~= n
           error('Matrices A and B have different rows')
        end
%
        if n < m
           error('suggest to use GRQ factorization')
        end
%
%       QR factorization of A:
%                       Q'*A = RA
%
        [Q,R] = qr(A);
```

```
%
%       RQ factorization of Q'*B:
%                       Q'*B = S*V;
%
        [S,V] = rq(Q'*B);
%
%       Test the backward error
%
        resida = norm(A - Q*R)
        residb = norm(B - Q*S*V)



        function [Q,U,R,S] = grq(A,B)
%
%  GRQ ------------------------------------------------------------
%
%       Compute the generalized RQ factorization of A(nxm) and
%       B(nxp):
%
%               A = Q*R;        B = Q*S*V;
%
%       where n <= m, Q(nxn), V(pxp) are orthogonal matrices,
%       R and S assumes one of the forms:
%       if n <= p
%
%               R = [ 0  R11 ] n         S = [ S11 S12 ] n
%                    m-n  n                    n  p-n
%
%       if n > p
%
%               R = [ 0  R11 ] n         S = [ S11 ] p
%                    m-n  n                  [  0  ] n-p
%                                              p
%
%       where n by n matrix R11 and min(n,p) by min(n,p) matrix
%       S11 are upper triangular.
%
%  ---------------------------------------------------------------
%
        [n,m] = size(A);
        [n1,p] = size(B);
%
        if n1 ~= n
           error('Matrices A and B have different rows')
        end
%
        if n > m
```

```
          error('suggest to use GQR factorization')
        end
%
%       QR decomposition of B:
%                        Q'*B = S
%
        [Q,S] = qr(B);
%
%       RQ decomposition of Q'*A:
%                        Q'*A = R*U;
%
        [R,U] = rq(Q'*A);
%
%       Test the backward error
%
        resida = norm(A - Q*R*U)
        residb = norm(B - Q*S)



        function [Q,P,V,R,S] = gqrp(A,B)
%
%  GQRP ------------------------------------------------------------
%
%       Generalized QR factorization with partial pivoting of
%       matrices A(nxm) and B(nxp).
%
%       Find orthogonal matrices Q, V and permutation matrix P
%       such that
%
%               A = Q*R*P';     B = Q*S*V';
%
%       where
%
%              R = [ R11 R12 ] q
%                  [  0   0  ] n-q
%                      q  m-q
%
%       R11(qxq) is nonsingular upper trinagular,
%       and if n <= p, then
%
%              S = [  0  S11 S12  ] q
%                  [  0   0  S22  ] k
%                  [  0   0   0   ] n-k-q
%                    p-n  q  n-q
%
%       S22(kx(n-q)) is full row rank upper trapezoidal, if exists.
%
```

```
%       else if n > p, then
%
%              S = [ S11 S12 ] q
%                  [  0  S22 ] k
%                  [  0   0  ] n-q-k
%                  p-n+q  n-q
%
%       where S21(kxp) is full row rank upper trapezoidal form.
%
%  ----------------------------------------------------------------
%
        [n,m] = size(A);
        [n1,p] = size(B);
%
        if n1 ~= n
           error('Matrices A and B have different rows')
        end
%
%       QR decomposition with column pivoting of A:
%                       Q'*A*P = R
%
        [Q,R,P] = qr(A);
        q = rank(R);
%
        if n<= p
           [S,V] = rq(Q'*B);
           V = V';
           [Q1,S(q+1:n,p-n+q+1:p),P1] = qr(S(q+1:n,p-n+q+1:p));
           S(1:q,p-n+q+1:p) = S(1:q,p-n+q+1:p)*P1;
           Q(1:n,q+1:n) = Q(1:n,q+1:n)*Q1;
           V(1:p,p-n+q+1:p) = V(1:p,p-n+q+1:p)*P1;
%
        else   % n > p case
           if p <= n-q
              S = Q'*B;
              [Q1,S(q+1:n,1:p),P1] = qr(S(q+1:n,1:p));
              S(1:q,1:p) = S(1:q,1:p)*P1;
              Q(1:n,q+1:n) = Q(1:n,q+1:n)*Q1;
              V = P1;
           else  % p > n-q
              [S,V] = rq(Q'*B);
              V = V';
              [Q1,S(q+1:n,p-(n-q)+1:p),P1] = qr(S(q+1:n,p-(n-q)+1:p));
              S(1:q,p-(n-q)+1:p)  = S(1:q,p-(n-q)+1:p)*P1;
              Q(1:n,q+1:n) = Q(1:n,q+1:n)*Q1;
              V(1:p,p-(n-q)+1:p) = V(1:p,p-(n-q)+1:p)*P1;
           end
        end
```

```
%
%       Test the backward error if wanted.
%
        resida = norm(A - Q*R*P')
        residb = norm(B - Q*S*V')



        function [R,Q] = rq(A)
%
%  RQ  -----------------------------------------------------------
%
%         Compute RQ decomposition of matrix A.
%
%                       A = R*Q
%
%  ---------------------------------------------------------------
%
        [n,m] = size(A);
        Q = eye(m);
%
           if n <= m
              for i = n:-1:1
                 x = A(i,1:m-n+i);
                 v = housen(x);
                 v = v';
                 A(1:i,1:m-n+i) = colhouse(A(1:i,1:m-n+i),v);
                 Q(1:m,1:m-n+i) = colhouse(Q(1:m,1:m-n+i),v);
              end
           else
              for i = n:-1:n-m+2
                 x = A(i,1:m-n+i);
                 v = housen(x);
                 v = v';
                 A(1:i,1:m-n+i) = colhouse(A(1:i,1:m-n+i),v);
                 Q(1:m,1:m-n+i) = colhouse(Q(1:m,1:m-n+i),v);
              end
           end
%
        R = A;
        Q = Q';
%
        function A = colhouse(A,v)
%
%       Apply the Householder matrix from right left
%           A <== A*H
%
        beta = -2/(v'*v);
```

```
        w = beta*A*v;
        A = A + w*v';
%
        function v = house(x)
%
%       HOUSE computes the Householdr vectors
%
         n = max(size(x));
         mu = norm(x);
         v = x;
         if mu ~= 0,
            beta = x(1) + sign(x(1))*mu;
            v(2:n) = v(2:n)/beta;
         end
         v(1) = 1;
```

References

[1] J. L. Barlow, N. K. Nichols, and R. J. Plemmons, Iterative methods for equality constrained least squares problems, SIAM J. Sci. and Stat. Comp. 9, pp.892-906, 1988.

[2] C. H. Bischof and P. C. Hansen Structure-Perserving and Rank-Revealing QR-Factorizations, Argonal National Laboratory, priprint MCS-P100-0989, 1989

[3] L. Elden, Perturbation theory for the least squares problem with linear equality constraints, SIAM J. Numer. Anal. Vol.17, pp.338-350, 1980

[4] G. H. Golub and C. F. Van Loan, Matrix computations, The Johns Hopkins Univ. Press, Baltimore, MD, 1989

[5] N. J. Higham, *FORTRAN codes for estimating the one-norm of a real or complex matrix, with applications to condition estimation*, ACM TOMS, 14: 381-396(1988).

[6] S. Hammarling, The numerical solution of the general Gauss-Markov linear model, NAG Technical Report, TR2/85, 1985.

[7] J. J. Dongarra, J. R. Bunch, C. B. Moler and G. W. Stewart, LINPACK User's Guide, Society for Industrial and Applied Mathematics, Philadelphia, 1978.

[8] C. L. Lawson and R. J. Hanson, Solving least squares problems, Prentice-Hall, Englewood Cliffs, NJ, 1974.

[9] C. Paige, Some aspects of generalized QR factorization.

[10] C. Paige, Computer solution and perturbation analysis of generalized linear least squares problems. Math. Comp. Vol. 33, pp. 171-183, 1979

[11] G. W. Stewart, On the perturbation of pseudo-inverses, projections and linear least squares problems, SIAM Rev. Vol.19, pp.634-662, 1977

[12] C. F. Van Loan, On the method of weighting for equality-constrained least-squares problems, SIAM J. Numer. Anal. 22, pp.851-864. 1985.