ornl

ORNL/TM-10980

OAK RIDGE
NATIONAL
LABORATORY

MARTIN MARIETTA

A Scheme for Handling Rank Deficiency in the Solution of Sparse Linear Least Squares Problems

Esmond Ng

OAK RIDGE NATIONAL LABORATORY
CENTRAL RESEARCH LIBRARY
CIRCULATION SECTION
4500N ROOM 175
LIBRARY LOAN COPY
DO NOT TRANSFER TO ANOTHER PERSON
If you wish someone else to see this report, send in name with report and the library will arrange a loan.
UCN-7969 3 9-77

OPERATED BY
MARTIN MARIETTA ENERGY SYSTEMS, INC.
FOR THE UNITED STATES
DEPARTMENT OF ENERGY

Printed in the United States of America. Available from
National Technical Information Service
U.S. Department of Commerce
5285 Port Royal Road, Springfield, Virginia 22161
NTIS price codes—Printed Copy: A03; Microfiche A01

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

ORNL/TM-10980

Engineering Physics and Mathematics Division

Mathematical Sciences Section

A SCHEME FOR HANDLING RANK DEFICIENCY IN THE SOLUTION OF SPARSE LINEAR LEAST SQUARES PROBLEMS

Esmond Ng

Oak Ridge National Laboratory
Mathematical Sciences Section
P.O. Box 2009, Bldg. 9207-A
Oak Ridge, TN 37831-8083

Date Published: November 1988

Research was supported by the Applied Mathematical Sciences Research Program of the Office of Energy Research, U.S. Department of Energy.

Prepared by the
Oak Ridge National Laboratory
Oak Ridge, Tennessee 37831
operated by
Martin Marietta Energy Systems, Inc.
for the
U.S. DEPARTMENT OF ENERGY
under Contract No. DE-AC-05-84OR21400

MARTIN MARIETTA ENERGY SYSTEMS LIBRARIES

3 4456 0289110 0

Contents

1 Introduction 1

2 The updating algorithm 4

3 Implementation details 10

4 Conclusion 16

5 References 17

A SCHEME FOR HANDLING RANK DEFICIENCY IN THE SOLUTION OF SPARSE LINEAR LEAST SQUARES PROBLEMS

Esmond Ng

Abstract

Recently we have presented several schemes for computing sparse orthogonal factorizations using static data structures. The novel feature of each scheme is that the data structures are large enough to store both the orthogonal transformations and upper triangular factor explicitly. Thus, multiple least squares problems with the same observation matrix can be solved easily. However, in order to make use of the static data structures, the orthogonal factorization is computed without column interchanges. In this article we develop an algorithm that makes use of the resulting factorization to solve rank-deficient least squares problems. The techniques used are similar to those employed by Bjorck.

1. Introduction

In this article we consider the direct solution of a linear least squares problem

$$\min_{\boldsymbol{x}} \|\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}\|_2,$$

where $\boldsymbol{A}$ is a sparse $m \times n$ matrix, with $m \geq n$, and $\boldsymbol{b}$ is an m-vector. The solution scheme is based on an orthogonal decomposition of $\boldsymbol{A}$:

$$\boldsymbol{A} = \boldsymbol{Q} \begin{pmatrix} \boldsymbol{R} \\ \boldsymbol{O} \end{pmatrix},$$

where $\boldsymbol{Q}$ is an $m \times m$ orthogonal matrix and $\boldsymbol{R}$ is an $n \times n$ upper triangular matrix. When $\boldsymbol{A}$ has full rank, $\boldsymbol{R}$ is nonsingular and the solution to the least squares problem is obtained by solving $\boldsymbol{R}\boldsymbol{x} = \boldsymbol{c}$, where $\boldsymbol{c}$ contains the first n components of $\boldsymbol{Q}^T\boldsymbol{b}$.

There are several methods for computing the orthogonal decomposition of $\boldsymbol{A}$ when it is sparse and has full rank [10]. In [2], George and Heath described a novel way of computing the upper triangular factor $\boldsymbol{R}$, which is based on the observation that $\boldsymbol{R}$ is mathematically the Cholesky factor of $\boldsymbol{A}^T\boldsymbol{A}$ (apart from possible sign differences in some rows). If $\boldsymbol{A}^T\boldsymbol{A}$ is sparse, techniques developed for solving sparse symmetric positive definite systems can be employed to preserve and exploit the sparsity of $\boldsymbol{R}$. More precisely, a symbolic Cholesky factorization algorithm can be applied to $\boldsymbol{A}^T\boldsymbol{A}$ to obtain the structure of $\boldsymbol{R}$. This allows a storage scheme to be set up for storing the nonzeros of $\boldsymbol{R}$ [3]. Using the fixed (or static) storage scheme, the upper triangular factor $\boldsymbol{R}$ is then obtained by annihilating the rows of $\boldsymbol{A}$ using Givens rotations. Since the symbolic factorization algorithm does not predict the positions of the rotations, a static data structure cannot be set up to store the rotations. (It was shown in [7] that the symbolic factorization algorithm on $\boldsymbol{A}^T\boldsymbol{A}$ can be modified to predict the positions of the rotations, although the resulting data structures may be too generous.) In [2], the rotations are applied to the vector $\boldsymbol{b}$ during the annihilation process and are

not saved. It is well known that the sparsity of $\boldsymbol{R}$ depends not only on the structure of $\boldsymbol{A}^T\boldsymbol{A}$, but also depends on the ordering of the columns and rows of $\boldsymbol{A}^T\boldsymbol{A}$. Thus, a fill-reducing symmetric ordering for $\boldsymbol{A}^T\boldsymbol{A}$ should be chosen to preserve the sparsity of $\boldsymbol{R}$. See [3] for details. Note that reordering the columns and rows of $\boldsymbol{A}^T\boldsymbol{A}$ symmetrically is equivalent to reordering the columns of $\boldsymbol{A}$.

In [8], George and Ng described an efficient symbolic factorization algorithm that computes the structure of $\boldsymbol{R}$ directly from that of $\boldsymbol{A}$. If $\boldsymbol{Q}$ is a sequence of Householder transformations, the symbolic factorization algorithm also produces the structure of the transformations. Thus, not only can a static data structure be determined for storing the nonzeros of $\boldsymbol{R}$, but a fixed storage scheme can also be set up for storing the nonzeros of the Householder transformations. In [4], George, Liu and Ng proposed an efficient way of storing and computing $\boldsymbol{R}$ and the Householder transformations. It should be noted that the sparsity of $\boldsymbol{R}$ and the Householder transformations depend on both the structure of $\boldsymbol{A}$ and the initial column ordering.

To summarize, there are efficient methods for reducing a sparse matrix to upper triangular form using orthogonal transformations. However, the column ordering is often chosen beforehand to preserve sparsity of the upper triangular factor $\boldsymbol{R}$. This implies that if a static data structure is used, the orthogonal decomposition must be computed without any column interchanges during the numerical computation phase, since column pivoting will change the sparsity of $\boldsymbol{R}$. Hence, the approaches in [2], [4] and [8] work well when the matrix $\boldsymbol{A}$ has full rank and is well-conditioned.

We now consider the case in which $\boldsymbol{A}$ is rank deficient. Suppose $\boldsymbol{R}$ is obtained from $\boldsymbol{A}$ without column pivoting. In exact arithmetic, some of the diagonal elements of $\boldsymbol{R}$ will be zero. However, for those rows whose diagonal elements are zero, the off-diagonal elements need not be zero. Hence, the total rank-deficiency of $\boldsymbol{A}$ and $\boldsymbol{R}$ cannot be determined simply by counting the number of zeros on the diagonal in $\boldsymbol{R}$. The situation becomes fuzzier in the presence of round-off errors. Instead of having exact zeros on the diagonal, it is likely the case that some of

the diagonal elements of $\boldsymbol{R}$ will be close to zero or small in magnitude compared to the remaining diagonal elements.

In [11], Heath proposed a technique for handling rank-deficiency in the basic method described in [2]. Let δ be a user-specified tolerance, the choice of which will depend on the magnitude of the diagonal elements of $\boldsymbol{R}$ and the machine precision. The basic idea is to use additional Givens rotations to annihilate the offdiagonal nonzeros in any rows of $\boldsymbol{R}$ whose diagonal elements are less than δ in magnitude. When the annihilation process is completed, any rows in the resulting upper triangular matrix will be regarded as null if their diagonal elements are less than δ in magnitude. Using the resulting upper triangular matrix, Heath developed an algorithm for computing the minimum-norm solution to the least squares problem. In [1], Bjorck extended the ideas in [11] to develop an algorithm for solving a general sparse constrained linear least squares problem in which the least squares and constraint matrices can have arbitrary ranks.

For the approaches described in [4] and [8], Heath's approach can be employed to compute the minimum-norm solution as well, so long as only one right hand side vector $\boldsymbol{b}$ has to be solved, or when numerous right hand sides are available at the same time. If several right hand sides are to be solved and if each has to be processed one after another, then the Givens rotations needed to process the upper triangular factor $\boldsymbol{R}$ must be saved. However, the static data structures described in [4] and [8] have space to store only the nonzeros of $\boldsymbol{Q}$ (represented as a sequence of Householder transformations); they do not have space to store the extra transformations needed to process $\boldsymbol{R}$. Thus, we need an algorithm for solving a rank-deficient problem that uses the upper triangular factor obtained in the initial orthogonal factorization. The objective of this note is to describe such an algorithm, in which rows of $\boldsymbol{R}$ with diagonal elements less than δ are treated as updates to the problem. Rank-deficiency is determined using the singular value decomposition during the updating process. The techniques are similar to those employed in [1]. An outline of the article is as follows. In Section 2, the updating algorithm is derived. Some implementation issues are presented in Section 3.

Finally, some concluding remarks are given in Section 4.

2. The updating algorithm

Consider the linear least squares problem

$$\min_{\boldsymbol{x}} \|\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}\|_2, \tag{1}$$

where $\boldsymbol{A}$ is large and sparse. Recall that the upper triangular factor $\boldsymbol{R}$ is mathematically the Cholesky factor of $\boldsymbol{A}^T\boldsymbol{A}$. If some of the rows of $\boldsymbol{A}$ are relatively dense, then $\boldsymbol{A}^T\boldsymbol{A}$, and consequently $\boldsymbol{R}$, may be dense. In order to preserve sparsity, it may be desirable to withhold any dense rows from the initial sparse orthogonal factorization. Instead of solving (1), the sparse orthogonal decomposition is used to solve a smaller least squares problem. Then any dense rows that were withheld can be used to update the solution.

The technique of handling dense rows is useful in another context. It is quite often the case that after a least squares problem has been solved, more equations become available. For large and sparse problems, if the number of new equations is small, it is desirable to be able to update the solution to the first least squares problem using the new equations to obtain the new solutions.

With the discussion above, we assume that $\boldsymbol{A}$ and $\boldsymbol{b}$ are partitioned respectively into $\binom{\boldsymbol{C}}{\boldsymbol{D}}$ and $\binom{\boldsymbol{e}}{\boldsymbol{f}}$. The matrix $\boldsymbol{C}$ is assumed to be large and sparse, but $\boldsymbol{D}$ is small and may be dense. Let $\boldsymbol{C}$ be $m \times n$ and $\boldsymbol{D}$ be $p \times n$. We assume that $n \leq m$ and $p << n$. These assumptions on m, n and p are not essential for the correctness of the algorithm we will develop, but are vital for its efficiency. We do not make any assumptions on the ranks of $\boldsymbol{C}$ and $\boldsymbol{A}$. If $\boldsymbol{A}$ is rank deficient, (1) does not have a unique solution, and in that case, the minimum-norm solution is computed.

We assume that a sparse orthogonal factorization of $\boldsymbol{C}$ has been computed, using, for example, the approach and the static storage scheme described in [4]

or [8]:

$$C = Q \begin{pmatrix} R \\ O \end{pmatrix}. \tag{2}$$

Here Q is $m \times m$ and orthogonal, and R is $n \times n$ and upper triangular. As we pointed out in the previous section, an *a priori* column ordering of C is chosen to preserve sparsity in Q and R. Thus, the orthogonal decomposition of C is computed with a fixed column ordering. It is well known that, in floating point arithmetic, some of the diagonal elements in the upper triangular factor R will be close to zero or small in magnitude compared to the remaining diagonal elements if C is rank deficient. Furthermore, since the orthogonal factorization is computed without column interchanges, rows with small diagonal elements (in terms of magnitude) may occur anywhere in R. Without loss of generality, we assume that R has the form

$$R = \begin{pmatrix} R_1 & S_{12} & S_{13} \\ O & R_2 & S_{23} \\ O & O & R_3 \end{pmatrix}. \tag{3}$$

Here R_1, R_2 and R_3 are upper triangular. We assume that the diagonal elements of R_2 are less than δ in magnitude, where δ is some tolerance dependent on the magnitudes of the diagonal elements of R_1 and R_3 and the machine precision. That is, we assume that R_1 and R_3 are nonsingular, but R_2 may be rank deficient. Note that the offdiagonal nonzeros in R_2 and S_{23} need not be small, and hence the rank-deficiency of C may be less than the order of R_2. Let n_2 be the order of R_2. We assume that $n_2 << n$. Again the assumption on n_2 is for efficiency; it is not needed for the correctness of the algorithm.

It is important to note the difference in handling rank-deficiency between our scheme and Heath's approach [11]. In Heath's approach, diagonal elements that are less than δ in the final upper triangular factor are regarded as numerically zero. That is, rank-deficiency is determined when the final upper triangular factor is computed. Thus, the choice of suitable δ is important and is a delicate

issue. As we shall see below, δ is not used to determine rank-deficiency in our case; it is used to decide which rows should be in $\boldsymbol{R}_2$. Hence the choice of δ can be quite liberal. The determination of rank-deficiency is delayed until a dense least squares problem is solved using the singular value decomposition during the updating process.

To compute the minimum-norm solution to (1) using the orthogonal decomposition given by (2) and (3), we proceed as follows. First note that

$$\|\boldsymbol{A}\boldsymbol{x}-\boldsymbol{b}\|_2 = \left\| \begin{pmatrix} \boldsymbol{C} \\ \boldsymbol{D} \end{pmatrix} \boldsymbol{x} - \begin{pmatrix} \boldsymbol{e} \\ \boldsymbol{f} \end{pmatrix} \right\|_2 = \left\| \begin{pmatrix} \boldsymbol{R}_1 & \boldsymbol{S}_{12} & \boldsymbol{S}_{13} \\ \boldsymbol{O} & \boldsymbol{R}_2 & \boldsymbol{S}_{23} \\ \boldsymbol{O} & \boldsymbol{O} & \boldsymbol{R}_3 \\ \boldsymbol{O} & \boldsymbol{O} & \boldsymbol{O} \\ \boldsymbol{D}_1 & \boldsymbol{D}_2 & \boldsymbol{D}_3 \end{pmatrix} \boldsymbol{x} - \begin{pmatrix} \boldsymbol{Q}^T\boldsymbol{e} \\ \boldsymbol{f} \end{pmatrix} \right\|_2 .$$

We have partitioned $\boldsymbol{D}$ according to the partitioning in the upper triangular factor $\boldsymbol{R}$. Let $\boldsymbol{x} = \begin{pmatrix} \boldsymbol{u}^T & \boldsymbol{v}^T & \boldsymbol{w}^T \end{pmatrix}^T$ and $\boldsymbol{Q}^T\boldsymbol{e} = \begin{pmatrix} \boldsymbol{e}_1^T & \boldsymbol{e}_2^T & \boldsymbol{e}_3^T & \boldsymbol{e}_4^T \end{pmatrix}^T$ be partitioned accordingly. Then

$$\|\boldsymbol{A}\boldsymbol{x}-\boldsymbol{b}\|_2 = \left\| \begin{pmatrix} \boldsymbol{R}_1\boldsymbol{u} + \boldsymbol{S}_{12}\boldsymbol{v} + \boldsymbol{S}_{13}\boldsymbol{w} - \boldsymbol{e}_1 \\ \boldsymbol{R}_2\boldsymbol{v} + \boldsymbol{S}_{23}\boldsymbol{w} - \boldsymbol{e}_2 \\ \boldsymbol{R}_3\boldsymbol{w} - \boldsymbol{e}_3 \\ -\boldsymbol{e}_4 \\ \boldsymbol{D}_1\boldsymbol{u} + \boldsymbol{D}_2\boldsymbol{v} + \boldsymbol{D}_3\boldsymbol{w} - \boldsymbol{f} \end{pmatrix} \right\|_2 .$$

Hence, $\|\boldsymbol{A}\boldsymbol{x}-\boldsymbol{b}\|_2$ is minimized when $\boldsymbol{u}$, $\boldsymbol{v}$ and $\boldsymbol{w}$ are chosen to minimize

$$\begin{aligned} &\|\boldsymbol{R}_1\boldsymbol{u} + \boldsymbol{S}_{12}\boldsymbol{v} + \boldsymbol{S}_{13}\boldsymbol{w} - \boldsymbol{e}_1\|_2^2 + \|\boldsymbol{R}_2\boldsymbol{v} + \boldsymbol{S}_{23}\boldsymbol{w} - \boldsymbol{e}_2\|_2^2 \\ &+\|\boldsymbol{R}_3\boldsymbol{w} - \boldsymbol{e}_3\|_2^2 + \|\boldsymbol{D}_1\boldsymbol{u} + \boldsymbol{D}_2\boldsymbol{v} + \boldsymbol{D}_3\boldsymbol{w} - \boldsymbol{f}\|_2^2. \end{aligned} \tag{4}$$

If a unique solution does not exist (in the case when $\boldsymbol{A}$ is rank deficient), we will choose $\boldsymbol{u}$, $\boldsymbol{v}$ and $\boldsymbol{w}$ so that (4) and $\|\boldsymbol{x}\|_2$ are both minimized.

We define the residuals $\boldsymbol{r}_1$, $\boldsymbol{r}_2$, $\boldsymbol{r}_3$ and $\boldsymbol{r}_D$ by

$$\begin{aligned} \boldsymbol{r}_1 &= \boldsymbol{R}_1\boldsymbol{u} + \boldsymbol{S}_{12}\boldsymbol{v} + \boldsymbol{S}_{13}\boldsymbol{w} - \boldsymbol{e}_1, && (5)\\ \boldsymbol{r}_2 &= \boldsymbol{R}_2\boldsymbol{v} + \boldsymbol{S}_{23}\boldsymbol{w} - \boldsymbol{e}_2, && (6)\\ \boldsymbol{r}_3 &= \boldsymbol{R}_3\boldsymbol{w} - \boldsymbol{e}_3, && (7)\\ \boldsymbol{r}_D &= \boldsymbol{D}_1\boldsymbol{u} + \boldsymbol{D}_2\boldsymbol{v} + \boldsymbol{D}_3\boldsymbol{w} - \boldsymbol{f}. && (8) \end{aligned}$$

Note that $\boldsymbol{r}_2 = \boldsymbol{e}_2$ in Bjorck's approach since $\boldsymbol{R}_2$ and $\boldsymbol{S}_{23}$ would be zero. We rewrite (7) as

$$\boldsymbol{w} = \boldsymbol{R}_3^{-1}\boldsymbol{r}_3 + \boldsymbol{R}_3^{-1}\boldsymbol{e}_3 = \boldsymbol{R}_3^{-1}\boldsymbol{r}_3 + \hat{\boldsymbol{e}}_3. \tag{9}$$

Substituting (9) into (5), we obtain

$$\boldsymbol{r}_1 = \boldsymbol{R}_1\boldsymbol{u} + \boldsymbol{S}_{12}\boldsymbol{v} + \boldsymbol{S}_{13}\boldsymbol{R}_3^{-1}\boldsymbol{r}_3 - (\boldsymbol{e}_1 - \boldsymbol{S}_{13}\hat{\boldsymbol{e}}_3),$$

and, after rearranging,

$$\begin{aligned} \boldsymbol{u} &= \boldsymbol{R}_1^{-1}\boldsymbol{r}_1 - \boldsymbol{R}_1^{-1}\boldsymbol{S}_{13}\boldsymbol{R}_3^{-1}\boldsymbol{r}_3 - \boldsymbol{R}_1^{-1}\boldsymbol{S}_{12}\boldsymbol{v} + \boldsymbol{R}_1^{-1}(\boldsymbol{e}_1 - \boldsymbol{S}_{13}\hat{\boldsymbol{e}}_3)\\ &= \boldsymbol{R}_1^{-1}\boldsymbol{r}_1 - \boldsymbol{R}_1^{-1}\boldsymbol{S}_{13}\boldsymbol{R}_3^{-1}\boldsymbol{r}_3 - \boldsymbol{R}_1^{-1}\boldsymbol{S}_{12}\boldsymbol{v} + \hat{\boldsymbol{e}}_1. && (10) \end{aligned}$$

With (9) and (10), (6) and (8) can be expressed as

$$\begin{aligned} \boldsymbol{r}_2 &= \boldsymbol{S}_{23}\boldsymbol{R}_3^{-1}\boldsymbol{r}_3 + \boldsymbol{R}_2\boldsymbol{v} - (\boldsymbol{e}_2 - \boldsymbol{S}_{23}\hat{\boldsymbol{e}}_3)\\ &= \boldsymbol{S}_{23}\boldsymbol{R}_3^{-1}\boldsymbol{r}_3 + \boldsymbol{R}_2\boldsymbol{v} - \hat{\boldsymbol{e}}_2, && (11) \end{aligned}$$

$$\begin{aligned} \boldsymbol{r}_D &= \boldsymbol{D}_1\boldsymbol{R}_1^{-1}\boldsymbol{r}_1 + (\boldsymbol{D}_3\boldsymbol{R}_3^{-1} - \boldsymbol{D}_1\boldsymbol{R}_1^{-1}\boldsymbol{S}_{13}\boldsymbol{R}_3^{-1})\boldsymbol{r}_3\\ &\quad + (\boldsymbol{D}_2 - \boldsymbol{D}_1\boldsymbol{R}_1^{-1}\boldsymbol{S}_{12})\boldsymbol{v} - (\boldsymbol{f} - \boldsymbol{D}_1\hat{\boldsymbol{e}}_1 - \boldsymbol{D}_3\hat{\boldsymbol{e}}_3)\\ &= \boldsymbol{D}_1\boldsymbol{R}_1^{-1}\boldsymbol{r}_1 + \hat{\boldsymbol{D}}_3\boldsymbol{r}_3 + \hat{\boldsymbol{D}}_2\boldsymbol{v} - \hat{\boldsymbol{f}}. && (12) \end{aligned}$$

Equations (11) and (12) correspond to an $(n_2 + p) \times (n + p)$ underdetermined

system of linear equations

$$\boldsymbol{M}\boldsymbol{r} = \boldsymbol{K}\boldsymbol{v} - \boldsymbol{g}, \tag{13}$$

where $\boldsymbol{M}$, $\boldsymbol{K}$, $\boldsymbol{r}$ and $\boldsymbol{g}$ are defined below.

$$\boldsymbol{M} = \begin{pmatrix} \boldsymbol{O} & \boldsymbol{I} & -\boldsymbol{S}_{23}\boldsymbol{R}_3^{-1} & \boldsymbol{O} \\ -\boldsymbol{D}_1\boldsymbol{R}_1^{-1} & \boldsymbol{O} & -\hat{\boldsymbol{D}}_3 & \boldsymbol{I} \end{pmatrix} \quad , \quad \boldsymbol{K} = \begin{pmatrix} \boldsymbol{R}_2 \\ \hat{\boldsymbol{D}}_2 \end{pmatrix}$$

$$\boldsymbol{r} = \begin{pmatrix} \boldsymbol{r}_1 \\ \boldsymbol{r}_2 \\ \boldsymbol{r}_3 \\ \boldsymbol{r}_D \end{pmatrix} \quad , \quad \boldsymbol{g} = \begin{pmatrix} \hat{\boldsymbol{e}}_2 \\ \hat{\boldsymbol{f}} \end{pmatrix}$$

Note that the coefficient matrix $\boldsymbol{M}$ always has full rank. (In Bjorck's updating algorithm, $\boldsymbol{M} = \begin{pmatrix} -\boldsymbol{D}_1\boldsymbol{R}_1^{-1} & -\hat{\boldsymbol{D}}_3 & \boldsymbol{I} \end{pmatrix}$, $\boldsymbol{K} = \hat{\boldsymbol{D}}_2$, $\boldsymbol{r} = \begin{pmatrix} \boldsymbol{r}_1^T & \boldsymbol{r}_3^T & \boldsymbol{r}_D^T \end{pmatrix}^T$ and $\boldsymbol{g} = \hat{\boldsymbol{f}}$.) Hence the minimum-norm residual vector of the original least squares problem is obtained by computing the minimum-norm solution to (13).

Denote an orthogonal decomposition of $\boldsymbol{M}$ by

$$\boldsymbol{M}^T = \boldsymbol{Q}_L^T \begin{pmatrix} \boldsymbol{L}^T \\ \boldsymbol{O} \end{pmatrix},$$

where $\boldsymbol{Q}_L^T$ is orthogonal and $\boldsymbol{L}^T$ is upper triangular and nonsingular. Using the orthogonal decomposition of $\boldsymbol{M}$ and writing $\boldsymbol{r}$ as $\boldsymbol{Q}_L^T\binom{\boldsymbol{s}}{\boldsymbol{t}}$, (13) becomes

$$\begin{pmatrix} \boldsymbol{L} & \boldsymbol{O} \end{pmatrix} \begin{pmatrix} \boldsymbol{s} \\ \boldsymbol{t} \end{pmatrix} = \boldsymbol{K}\boldsymbol{v} - \boldsymbol{g}.$$

Thus, $\boldsymbol{s} = \boldsymbol{L}^{-1}\boldsymbol{K}\boldsymbol{v} - \boldsymbol{L}^{-1}\boldsymbol{g}$ and $\boldsymbol{t}$ is arbitrary. The minimum-norm solution to (13) is obtained by setting $\boldsymbol{t} = \boldsymbol{o}$ and by choosing $\boldsymbol{v}$ to be a solution to the following dense linear least squares problem

$$\min_{\boldsymbol{v}} \left\| \boldsymbol{L}^{-1}\boldsymbol{K}\boldsymbol{v} - \boldsymbol{L}^{-1}\boldsymbol{g} \right\|_2. \tag{14}$$

The coefficient matrix $\boldsymbol{L}^{-1}\boldsymbol{K}$ is $(n_2+p)\times n_2$ and may be rank deficient. A reliable way to solve (14) is to use the singular value decomposition of $\boldsymbol{L}^{-1}\boldsymbol{K}$. When (14) has been solved, $\boldsymbol{r}$, and hence $\boldsymbol{w}$, will be known. It should be noted that this is the only place in the algorithm where rank deficiency has to be determined.

Let $\bar{\boldsymbol{v}}$ be any solution to (14). Furthermore, suppose the columns of $\boldsymbol{N}$ form a basis of the null space of $\boldsymbol{L}^{-1}\boldsymbol{K}$. The matrix $\boldsymbol{N}$ can be obtained easily from the singular value decomposition of $\boldsymbol{L}^{-1}\boldsymbol{K}$. Thus, the general solution to (14) is given by $\boldsymbol{v} = \bar{\boldsymbol{v}} + \boldsymbol{N}\boldsymbol{z}$, where $\boldsymbol{z}$ is arbitrary. Now consider (5) again:

$$\begin{aligned} \boldsymbol{r}_1 &= \boldsymbol{R}_1\boldsymbol{u} + \boldsymbol{S}_{12}\boldsymbol{v} + \boldsymbol{S}_{13}\boldsymbol{w} - \boldsymbol{e}_1 \\ &= \boldsymbol{R}_1\boldsymbol{u} + \boldsymbol{S}_{12}\boldsymbol{N}\boldsymbol{z} - (\boldsymbol{e}_1 - \boldsymbol{S}_{12}\bar{\boldsymbol{v}} - \boldsymbol{S}_{13}\boldsymbol{w}). \end{aligned}$$

That is,

$$\begin{aligned} \boldsymbol{u} &= -\boldsymbol{R}_1^{-1}\boldsymbol{S}_{12}\boldsymbol{N}\boldsymbol{z} + \boldsymbol{R}_1^{-1}(\boldsymbol{e}_1 - \boldsymbol{S}_{12}\bar{\boldsymbol{v}} - \boldsymbol{S}_{13}\boldsymbol{w} + \boldsymbol{r}_1) \\ &= -\boldsymbol{R}_1^{-1}\boldsymbol{S}_{12}\boldsymbol{N}\boldsymbol{z} + \bar{\boldsymbol{e}}_1. \end{aligned} \tag{15}$$

Note that both $\boldsymbol{u}$ and $\boldsymbol{v}$ depend on $\boldsymbol{z}$. Hence $\boldsymbol{z}$ can be chosen to minimize $\left\|\binom{\boldsymbol{u}}{\boldsymbol{v}}\right\|_2$. (Note that $\boldsymbol{w}$ is uniquely determined when (14) is solved.) This is equivalent to solving the following linear least squares problem:

$$\min_{\boldsymbol{z}} \left\| \begin{pmatrix} -\boldsymbol{R}_1^{-1}\boldsymbol{S}_{12}\boldsymbol{N} \\ \boldsymbol{N} \end{pmatrix} \boldsymbol{z} + \begin{pmatrix} \bar{\boldsymbol{e}}_1 \\ \bar{\boldsymbol{v}} \end{pmatrix} \right\|_2 . \tag{16}$$

Since the order of $\boldsymbol{R}_2$ is assumed to be small and the columns of $\boldsymbol{N}$ are linearly independent, (16) is just a small dense least squares problem with full rank, which can be solved using standard dense techniques, such as dense orthogonal decomposition. The solution vector $\binom{\boldsymbol{u}}{\boldsymbol{v}}$ is given by the minimum-norm residual vector in (16).

3. Implementation details

In the algorithm described in the previous section, we have to deal with different portions of the upper triangular factor $\boldsymbol{R}$ obtained in the orthogonal factorization of $\boldsymbol{C}$ at various stages. Since $\boldsymbol{R}$ is normally stored in some compact data structure, the implementation of the algorithm appears to be complicated. We will demonstrate below that the implementation can actually be done cleanly.

We first define $\hat{\boldsymbol{R}}$ to be the upper triangular matrix

$$\hat{\boldsymbol{R}} = \begin{pmatrix} \boldsymbol{R}_1 & \boldsymbol{S}_{12} & \boldsymbol{S}_{13} \\ \boldsymbol{O} & \boldsymbol{I} & \boldsymbol{S}_{23} \\ \boldsymbol{O} & \boldsymbol{O} & \boldsymbol{R}_3 \end{pmatrix},$$

which can be obtained easily by scanning the data structure for the upper triangular factor $\boldsymbol{R}$ and replacing $\boldsymbol{R}_2$ by the identity matrix. We will assume that $\boldsymbol{R}_2$ is saved elsewhere. Let n_i be the order of $\boldsymbol{R}_i$, for $i = 1, 2, 3$. Hence $n_1 + n_2 + n_3 = n$. In the following discussion, we will make use of partitioned vectors and matrices frequently and we assume that the partitionings are consistent with the row partitionings in $\boldsymbol{R}$ and $\hat{\boldsymbol{R}}$.

The first three quantities we need are

$$\hat{\boldsymbol{e}}_3 = \boldsymbol{R}_3^{-1}\boldsymbol{e}_3 \quad , \quad \hat{\boldsymbol{e}}_2 = \boldsymbol{e}_2 - \boldsymbol{S}_{23}\hat{\boldsymbol{e}}_3 \quad , \quad \hat{\boldsymbol{e}}_1 = \boldsymbol{R}_1^{-1}(\boldsymbol{e}_1 - \boldsymbol{S}_{13}\hat{\boldsymbol{e}}_3).$$

Define $\boldsymbol{c} = \begin{pmatrix} \boldsymbol{e}_1^T & \boldsymbol{e}_2^T & \boldsymbol{e}_3^T \end{pmatrix}^T$. Then $\hat{\boldsymbol{e}}_1$, $\hat{\boldsymbol{e}}_2$ and $\hat{\boldsymbol{e}}_3$ can be obtained by solving $\hat{\boldsymbol{R}}\boldsymbol{d} = \boldsymbol{c}$ using a modified backward substitution algorithm, which is described below. Let $\boldsymbol{d} = \begin{pmatrix} \boldsymbol{d}_1^T & \boldsymbol{d}_2^T & \boldsymbol{d}_3^T \end{pmatrix}^T$. Solving the last $n_2 + n_3$ equations of $\hat{\boldsymbol{R}}\boldsymbol{d} = \boldsymbol{c}$ gives

$$\begin{aligned} \boldsymbol{d}_3 &= \boldsymbol{R}_3^{-1}\boldsymbol{e}_3 = \hat{\boldsymbol{e}}_3, \\ \boldsymbol{d}_2 &= \boldsymbol{e}_2 - \boldsymbol{S}_{23}\boldsymbol{d}_3 = \boldsymbol{e}_2 - \boldsymbol{S}_{23}\hat{\boldsymbol{e}}_3 = \hat{\boldsymbol{e}}_2. \end{aligned}$$

Now we save $\boldsymbol{d}_2$ elsewhere and set $\boldsymbol{d}_2 = \boldsymbol{o}$. The solution of the first n_1 equations

would then produce

$$\boldsymbol{d}_1 = \boldsymbol{R}_1^{-1}(\boldsymbol{e}_1 - \boldsymbol{S}_{13}\boldsymbol{d}_3) = \boldsymbol{R}_1^{-1}(\boldsymbol{e}_1 - \boldsymbol{S}_{13}\hat{\boldsymbol{e}}_3) = \hat{\boldsymbol{e}}_1.$$

Thus, $\boldsymbol{d} = \begin{pmatrix} \hat{\boldsymbol{e}}_1^T & \boldsymbol{o} & \hat{\boldsymbol{e}}_3^T \end{pmatrix}^T$.

Next we need the matrices $\boldsymbol{D}_1\boldsymbol{R}_1^{-1}$, $\hat{\boldsymbol{D}}_2 = \boldsymbol{D}_2 - \boldsymbol{D}_1\boldsymbol{R}_1^{-1}\boldsymbol{S}_{12}$ and $\hat{\boldsymbol{D}}_3 = \boldsymbol{D}_3\boldsymbol{R}_3^{-1} - \boldsymbol{D}_1\boldsymbol{R}_1^{-1}\boldsymbol{S}_{13}\boldsymbol{R}_3^{-1}$. These matrices can be computed using an approach similar to the modified backward substitution algorithm described above. Consider the p lower triangular systems $\hat{\boldsymbol{R}}^T\boldsymbol{E} = \boldsymbol{D}^T$. Let $\boldsymbol{E} = \begin{pmatrix} \boldsymbol{E}_1^T & \boldsymbol{E}_2^T & \boldsymbol{E}_3^T \end{pmatrix}^T$. Solving the first $n_1 + n_2$ equations yields

$$\begin{aligned} \boldsymbol{E}_1 &= \boldsymbol{R}_1^{-T}\boldsymbol{D}_1^T, \\ \boldsymbol{E}_2 &= \boldsymbol{D}_2^T - \boldsymbol{S}_{12}^T\boldsymbol{E}_1 = \boldsymbol{D}_2^T - \boldsymbol{S}_{12}^T\boldsymbol{R}_1^{-T}\boldsymbol{D}_1^T = \hat{\boldsymbol{D}}_2^T. \end{aligned}$$

Now we save $\boldsymbol{E}_2$, set $\boldsymbol{E}_2 = \boldsymbol{O}$, and continue the forward substitution. Then

$$\boldsymbol{E}_3 = \boldsymbol{R}_3^{-T}(\boldsymbol{D}_3^T - \boldsymbol{S}_{13}^T\boldsymbol{E}_1) = \boldsymbol{R}_3^{-T}(\boldsymbol{D}_3^T - \boldsymbol{S}_{13}^T\boldsymbol{R}_1^{-T}\boldsymbol{D}_1^T) = \hat{\boldsymbol{D}}_3^T.$$

That is, at the end of the modified forward substitution process, we have $\boldsymbol{E} = \begin{pmatrix} \boldsymbol{D}_1\boldsymbol{R}_1^{-1} & \boldsymbol{O} & \hat{\boldsymbol{D}}_3 \end{pmatrix}^T$.

The vector $\hat{\boldsymbol{f}}$ in (12) is obtained by computing $\boldsymbol{f} - \boldsymbol{D}\boldsymbol{d}$. To form $\boldsymbol{M}$ in (13), we need the matrix $-\boldsymbol{S}_{23}\boldsymbol{R}_3^{-1}$, which can be obtained by solving n_2 lower triangular systems $\hat{\boldsymbol{R}}^T\boldsymbol{F} = \hat{\boldsymbol{I}}$, with $\hat{\boldsymbol{I}} = \begin{pmatrix} \boldsymbol{O} & \boldsymbol{I} & \boldsymbol{O} \end{pmatrix}^T$. If $\boldsymbol{F} = \begin{pmatrix} \boldsymbol{F}_1^T & \boldsymbol{F}_2^T & \boldsymbol{F}_3^T \end{pmatrix}^T$, then it is easy to see that $\boldsymbol{F}_3^T = -\boldsymbol{S}_{23}\boldsymbol{R}_3^{-1}$. The matrix $\boldsymbol{K}$ is easy to construct since both $\boldsymbol{R}_2$ and $\hat{\boldsymbol{D}}_2$ have been saved.

Finally, we have to compute the matrix $\begin{pmatrix} -\boldsymbol{R}_1^{-1}\boldsymbol{S}_{12}\boldsymbol{N} \\ \boldsymbol{N} \end{pmatrix}$ and the vector $\begin{pmatrix} \bar{\boldsymbol{e}}_1 \\ \bar{\boldsymbol{v}} \end{pmatrix}$ in order to solve (16). The matrix can be obtained by solving the upper triangular systems $\hat{\boldsymbol{R}}\boldsymbol{G} = \hat{\boldsymbol{N}}$, where $\hat{\boldsymbol{N}} = \begin{pmatrix} \boldsymbol{O} & \boldsymbol{N}^T & \boldsymbol{O} \end{pmatrix}^T$. Note that the solution is $\boldsymbol{G} = \begin{pmatrix} -(\boldsymbol{R}_1^{-1}\boldsymbol{S}_{12}\boldsymbol{N})^T & \boldsymbol{N}^T & \boldsymbol{O} \end{pmatrix}^T$. To compute $\begin{pmatrix} \bar{\boldsymbol{e}}_1 \\ \bar{\boldsymbol{v}} \end{pmatrix}$, we proceed as follows. After the solution of (13), we construct the vector $\hat{\boldsymbol{r}} = \begin{pmatrix} \boldsymbol{r}_1^T & \boldsymbol{o} & \boldsymbol{r}_3^T \end{pmatrix}^T$

and solve the upper triangular system $\hat{\boldsymbol{R}}\boldsymbol{h} = \hat{\boldsymbol{r}}$ using a strategy similar to the modified backward substitution described above. Let $\boldsymbol{h} = \begin{pmatrix} \boldsymbol{h}_1^T & \boldsymbol{h}_2^T & \boldsymbol{h}_2^T \end{pmatrix}^T$. The solution of the last $n_2 + n_3$ equations gives $\boldsymbol{h}_3 = \boldsymbol{R}_3^{-1}\boldsymbol{r}_3$ and $\boldsymbol{h}_2 = -\boldsymbol{S}_{23}\boldsymbol{h}_3$. Now we replace $\boldsymbol{h}_2$ by $\bar{\boldsymbol{v}}$, which is a solution to (14). Then the solution of the first n_1 equations will give $\boldsymbol{h}_1 = \boldsymbol{R}_1^{-1}(\boldsymbol{r}_1 - \boldsymbol{S}_{12}\bar{\boldsymbol{v}} - \boldsymbol{S}_{13}\boldsymbol{R}_3^{-1}\boldsymbol{r}_3)$. Note that $\boldsymbol{h}_3 + \boldsymbol{d}_3 = \boldsymbol{R}_3^{-1}\boldsymbol{r}_3 + \boldsymbol{R}_3^{-1}\boldsymbol{e}_3 = \boldsymbol{w}$. Also $\boldsymbol{h}_2 + \boldsymbol{d}_2 = \bar{\boldsymbol{v}}$. Furthermore,

$$\begin{aligned} \boldsymbol{h}_1 + \boldsymbol{d}_1 &= \boldsymbol{R}_1^{-1}(\boldsymbol{r}_1 - \boldsymbol{S}_{12}\bar{\boldsymbol{v}} - \boldsymbol{S}_{13}\boldsymbol{R}_3^{-1}\boldsymbol{r}_3) + \hat{\boldsymbol{e}}_1 \\ &= \boldsymbol{R}_1^{-1}(\boldsymbol{r}_1 - \boldsymbol{S}_{12}\bar{\boldsymbol{v}} - \boldsymbol{S}_{13}\boldsymbol{R}_3^{-1}\boldsymbol{r}_3) + \boldsymbol{R}_1^{-1}(\boldsymbol{e}_1 - \boldsymbol{S}_{13}\boldsymbol{R}_3^{-1}\boldsymbol{e}_3), \end{aligned}$$

which is the same as $\bar{\boldsymbol{e}}_1$.

In practice, rows with small diagonal elements could occur anywhere in the upper triangular factor. Thus, the structure of $\boldsymbol{R}$ would not have the form given in (3). However, for efficient implementation of the algorithm, all one needs is the ability to distinguish those rows from the remaining rows in $\boldsymbol{R}$. This can be achieved by maintaining a flag for each row so that the flag is set if the corresponding row has a small diagonal element.

We now summarize the algorithm step by step below. We also indicate the time complexity at each step (in terms of the number of multiplicative operations required); the operation counts for some standard matrix operations are obtained from [9] and only high order terms are retained.

1. Compute a sparse orthogonal decomposition of $\boldsymbol{C}$.

$$\boldsymbol{C} = \boldsymbol{Q}\begin{pmatrix} \boldsymbol{R}_1 & \boldsymbol{S}_{12} & \boldsymbol{S}_{13} \\ \boldsymbol{O} & \boldsymbol{R}_2 & \boldsymbol{S}_{23} \\ \boldsymbol{O} & \boldsymbol{O} & \boldsymbol{R}_3 \\ \boldsymbol{O} & \boldsymbol{O} & \boldsymbol{O} \end{pmatrix}$$

   It is generally impossible to give the time complexity for this step, since it is problem-dependent. Thus, we will use τ_C to denote the number of operations required to compute the orthogonal factorization.

2. Save $\boldsymbol{R}_2$ and construct $\hat{\boldsymbol{R}}$. Denote by σ the number of nonzeros (including the diagonal) in $\hat{\boldsymbol{R}}$.

3. Compute $\boldsymbol{F}_3^T = -\boldsymbol{S}_{23}\boldsymbol{R}_3^{-1}$ by solving $\hat{\boldsymbol{R}}^T\boldsymbol{F} = \hat{\boldsymbol{I}}$, where $\hat{\boldsymbol{I}} = \begin{pmatrix} \boldsymbol{O} & \boldsymbol{I} & \boldsymbol{O} \end{pmatrix}^T$. The time complexity for this step is $\tau_3 = \sigma n_2$, since there are n_2 triangular systems to solve.

4. Use the modified forward substitution algorithm described above to solve $\hat{\boldsymbol{R}}^T\boldsymbol{E} = \boldsymbol{D}^T$ to obtain $\boldsymbol{E} = \begin{pmatrix} \boldsymbol{D}_1\boldsymbol{R}_1^{-1} & \boldsymbol{O} & \hat{\boldsymbol{D}}_3 \end{pmatrix}^T$ and $\hat{\boldsymbol{D}}_2$. The time complexity for this step is $\tau_4 = \sigma p$, since $\boldsymbol{D}$ is $p \times n$.

5. Construct $\boldsymbol{K} = \binom{\boldsymbol{R}_2}{\hat{\boldsymbol{D}}_2}$ and

$$\boldsymbol{M} = \begin{pmatrix} \boldsymbol{O} & \boldsymbol{I} & -\boldsymbol{S}_{23}\boldsymbol{R}_3^{-1} & \boldsymbol{O} \\ -\boldsymbol{D}_1\boldsymbol{R}_1^{-1} & \boldsymbol{O} & -\hat{\boldsymbol{D}}_3 & \boldsymbol{I} \end{pmatrix} = \begin{pmatrix} \boldsymbol{F}^T & \boldsymbol{O} \\ -\boldsymbol{E}^T & \boldsymbol{I} \end{pmatrix}.$$

Compute an orthogonal decomposition of $\boldsymbol{M}^T = \boldsymbol{Q}_L^T\binom{\boldsymbol{L}^T}{\boldsymbol{O}}$. Since $\boldsymbol{M}^T$ is $(n+p) \times (n_2+p)$, the number of operations required to compute the orthogonal decomposition is $\tau_5 = (n+p)(n_2+p)^2 - \frac{1}{3}(n_2+p)^3$.

6. Form the $(n_2+p) \times n_2$ matrix $\boldsymbol{L}^{-1}\boldsymbol{K}$ and compute a singular value decomposition of $\boldsymbol{L}^{-1}\boldsymbol{K}$. Determine $\boldsymbol{N}$ whose columns form a basis of the null space of $\boldsymbol{L}^{-1}\boldsymbol{K}$. The time complexity for the singular value decomposition is approximately $\tau_6 = \left[\frac{1}{2}(n_2+p)^2 n_2\right] + \left[2(n_2+p)^2 n_2 + 4(n_2+p)n_2^2 + \frac{14}{3}n_2^3\right]$.

7. Solve $\hat{\boldsymbol{R}}\boldsymbol{G} = \hat{\boldsymbol{N}}$, where $\hat{\boldsymbol{N}} = \begin{pmatrix} \boldsymbol{O} & \boldsymbol{N}^T & \boldsymbol{O} \end{pmatrix}^T$, to obtain

$$\boldsymbol{G} = \begin{pmatrix} -(\boldsymbol{R}_1^{-1}\boldsymbol{S}_{12}\boldsymbol{N})^T & \boldsymbol{N}^T & \boldsymbol{O} \end{pmatrix}^T.$$

Compute an orthogonal decomposition of $\boldsymbol{G}$. Since $\boldsymbol{N}$ is at most $n_2 \times n_2$, computing $\boldsymbol{G}$ requires at most σn_2 operations, Similarly, the orthogonal decomposition of $\boldsymbol{G}$ requires at most $nn_2^2 - \frac{1}{3}n_2^3$ operations. Thus, the time complexity for this step is bounded by $\tau_7 = \sigma n_2 + nn_2^2 - \frac{1}{3}n_2^3$.

8. Form $\boldsymbol{Q}^T\boldsymbol{e}$. Let $\boldsymbol{c}$ be the first n components of $\boldsymbol{Q}^T\boldsymbol{e}$. Here $\boldsymbol{Q}$ is the orthogonal matrix we obtain at step 1, which is in factored form. As in step 1, it is difficult to say how many operations are required in forming $\boldsymbol{Q}^T\boldsymbol{e}$. Hence, we will use τ_Q to denote the time complexity at this step.

9. Solve $\hat{\boldsymbol{R}}\boldsymbol{d} = \boldsymbol{c}$ in the way described above to obtain $\boldsymbol{d} = \begin{pmatrix} \hat{\boldsymbol{e}}_1^T & \mathbf{o} & \hat{\boldsymbol{e}}_3^T \end{pmatrix}^T$ and $\hat{e}_2$. The time complexity at this step is $\tau_9 = \sigma$.

10. Compute $\hat{\boldsymbol{f}} = \boldsymbol{f} - \boldsymbol{D}\boldsymbol{d}$. The number of operations required to form $\hat{\boldsymbol{f}}$ is $\tau_{10} = pn$.

11. Compute $\boldsymbol{L}^{-1}\boldsymbol{g} = \boldsymbol{L}^{-1}\binom{\hat{e}_2}{\hat{\boldsymbol{f}}}$. Since $\boldsymbol{L}$ is $(n_2 + p) \times (n_2 + p)$, the time complexity for this step is $\tau_{11} = \frac{1}{2}(n_2 + p)^2$.

12. Solve the linear least squares problem $\min_{\boldsymbol{v}} \left\|\boldsymbol{L}^{-1}\boldsymbol{K}\boldsymbol{v} - \boldsymbol{L}^{-1}\boldsymbol{g}\right\|_2$ using the singular value decomposition computed at step 6. Let $\bar{\boldsymbol{v}}$ be any solution. Determine the minimum-norm residual vector $\boldsymbol{s}$. The number of operations required at this step is approximately $\tau_{12} = 2(n_2 + p)^2 + n_2^2$.

13. Compute the minimum-norm residual vector of the original problem $\boldsymbol{r} = \boldsymbol{Q}_L^T\binom{\boldsymbol{s}}{\mathbf{o}}$. Let $\boldsymbol{r} = \begin{pmatrix} \boldsymbol{r}_1^T & \boldsymbol{r}_2^T & \boldsymbol{r}_3^T & \boldsymbol{r}_D^T \end{pmatrix}^T$. The time complexity for this step is $\tau_{13} = [2(n + p) - (n_2 + p)]\,(n_2 + p)$.

14. Construct $\hat{\boldsymbol{r}} = \begin{pmatrix} \boldsymbol{r}_1^T & \mathbf{o} & \boldsymbol{r}_3^T \end{pmatrix}^T$ and solve $\hat{\boldsymbol{R}}\boldsymbol{h} = \hat{\boldsymbol{r}}$ as described above. The number of operations required in the solution of the triangular system is $\tau_{14} = \sigma$.

15. Compute $\boldsymbol{h} + \boldsymbol{d}$. At this point, $(\boldsymbol{h} + \boldsymbol{d}) = \begin{pmatrix} \bar{\boldsymbol{e}}_1^T & \bar{\boldsymbol{v}}^T & \boldsymbol{w}^T \end{pmatrix}^T$.

16. Solve the linear least squares problem $\min_{\boldsymbol{z}} \|\boldsymbol{G}\boldsymbol{z} - (\boldsymbol{h} + \boldsymbol{d})\|_2$ using the orthogonal decomposition computed at step 7. The minimum-norm residual vector is the minimum-norm solution to $\min_{\boldsymbol{x}} \|\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}\|_2$. The time complexity for solving the least squares problem is at most $\tau_{16} = 2(2n - n_2)n_2$.

In the case when several vectors $\boldsymbol{b}$ are to be solved, steps 1-7 have to be performed only once. Steps 8-16 have to be repeated for each different $\boldsymbol{b}$. When new equations are added, the matrix $\boldsymbol{D}$ changes. In this case, steps 4-16 have to be repeated.

Note that computing the orthogonal decomposition of $\boldsymbol{C}$ in step 1 is usually the most expensive step. Thus, if n is large and p and n_2 are small, and if a sparse orthogonal factorization of $\boldsymbol{C}$ has been computed, then the computation involves basically the solutions of a few sparse triangular systems and a few small dense problems. When a single least squares problem is solved, the time complexity of the entire algorithm is given by

$$\begin{gathered}\tau_C + \tau_Q + \sigma\,(2n_2 + p + 2) + n\,(2n_2^2 + p^2 + 2n_2p + 6n_2 + 3p) + \\ O(n_3^3 + p^3 + n_2^2p + n_2p^2).\end{gathered} \tag{17}$$

The first three terms are respectively the times for computing the orthogonal decomposition of $\boldsymbol{C}$ in step 1, for applying the transformations to $\boldsymbol{e}$ in step 8, and for solving various linear systems using the upper triangular matrix $\hat{\boldsymbol{R}}$. The fourth term in (17) is essentially the time for computing the orthogonal decompositions of $\boldsymbol{M}$ and $\boldsymbol{G}$.

Assuming p and n_2 are small, the storage requirement is dominated by the space for $\boldsymbol{Q}$ and $\boldsymbol{R}$, as the following discussion shows. As in the time complexity analysis, it is in general difficult to say how much storage is required by $\boldsymbol{Q}$ and $\boldsymbol{R}$. Thus, we simply denote it by θ_C. In the updating algorithm, we need space for the matrices $\boldsymbol{K}$ and $\boldsymbol{M}$, the dimensions of which are respectively $(n_2+p)\times n_2$ and $(n_2+p)\times(n+p)$. We also need space for $\boldsymbol{G}$, the dimension of which is at most $n \times n_2$. Moreover, we have to store the singular value decomposition of $\boldsymbol{K}$ and various vectors throughout the algorithm. It is easy to see that the total storage requirement is given approximately by

$$\theta_C + n\,(2n_2 + p) + O(n + n_2^2 + p^2 + n_2p), \tag{18}$$

and the dominating term is $\theta_C + n(2n_2 + p)$, assuming n_2 and p are small.

As we have mentioned above, if a sparse orthogonal factorization of $\boldsymbol{C}$ has been computed and if n_2 and p are small, the remaining computation involves mainly the solutions of a few sparse triangular systems and a few small dense problems. There are numerically stable algorithms for solving the small dense problems [9]. Stability may be an issue if the matrix $\boldsymbol{C}$ is ill-conditioned, since the orthogonal factorization of $\boldsymbol{C}$ is computed without any column interchanges. However, any ill-conditioning can be detected by computing an estimate of the condition number of $\hat{\boldsymbol{R}}$. See [1] and [11] for more discussions.

4. Conclusion

In this article, we have derived an algorithm for solving a rank-deficient sparse least squares problem using an orthogonal factorization of the least squares matrix that is computed without any column interchanges for stability. If the approach described in [4] or [8] is employed to compute the sparse orthogonal decomposition, then enough information can be saved easily and explicitly in a static data structure so that multiple problems with the same least squares matrix can be handled effectively. The algorithm is very similar to that described in [1]. One major difference is in the solution of multiple problems. In order to solve multiple problems using the algorithm in [1], either a dynamic data structure or auxiliary storage is needed to store the Givens rotations. Another difference is in the handling of rank-deficiency. In the algorithm described in this article, rows of the upper triangular factor $\boldsymbol{R}$ with diagonal elements close to zero are treated as row updates and the determination of rank-deficiency is delayed until the singular value decomposition of $\boldsymbol{L}^{-1}\boldsymbol{K}$ is computed. Thus the criterion for deciding whether a diagonal element of $\boldsymbol{R}$ is close to zero is not as crucial as in Heath's approach. We have also presented techniques for implementing the algorithm described in this article. Once the initial sparse orthogonal factorization has been computed, the solution can be obtained by solving a few sparse triangular systems of equations and some small dense matrix problems. An ef-

ficient implementation of the algorithm will be available in a future release of SPARSPAK [5,6]. The algorithm described in this article will be effective if the order of $\boldsymbol{R}_2$ (n_2) and the number of rows in $\boldsymbol{D}$ (p) are small.

5. References

[1] A. Bjorck. A general updating algorithm for constrained linear least squares problems. *SIAM J. Sci. Stat. Comput.*, 5:394–402, 1984.

[2] J. A. George and M. T. Heath. Solution of sparse linear least squares problems using Givens rotations. *Linear Algebra and its Appl.*, 34:69–83, 1980.

[3] J. A. George and J. W-H. Liu. *Computer Solution of Large Sparse Positive Definite Systems.* Prentice-Hall Inc., Englewood Cliffs, New Jersey, 1981.

[4] J. A. George, J. W-H. Liu, and E. G-Y. Ng. A data structure for sparse QR and LU factors. *SIAM J. Sci. Stat. Comput.*, 9:100–121, 1988.

[5] J. A. George and E. G-Y. Ng. A brief description of sparspak, waterloo sparse linear equation package. *ACM SIGNUM Newsletter*, 16(2):17–20, 1982.

[6] J. A. George and E. G-Y. Ng. A new release of sparspak - the waterloo sparse matrix package. *ACM SIGNUM Newsletter*, 19(4):9–13, 1984.

[7] J. A. George and E. G-Y. Ng. Orthogonal reduction of sparse matrices to upper triangular form using Householder transformations. *SIAM J. Sci. Stat. Comput.*, 7:460–472, 1986.

[8] J. A. George and E. G-Y. Ng. Symbolic factorization for sparse Gaussian elimination with partial pivoting. *SIAM J. Sci. Stat. Comput.*, 8:877–898, 1987.

[9] G. Golub and C. Van Loan. *Matrix Computations.* Johns Hopkins Press, 1983.

[10] M. T. Heath. Numerical methods for large sparse linear least squares problems. *SIAM J. Sci. Stat. Comput.*, 26:497–513, 1984.

[11] M. T. Heath. Some extensions of an algorithm for sparse linear least squares problems. *SIAM J. Sci. Stat. Comput.*, 3:223–237, 1982.

INTERNAL DISTRIBUTION

1. B. R. Appleton
2. J. B. Drake
3. G. A. Geist
4-5. R. F. Harbison
6. M. T. Heath
7-11. J. K. Ingersoll
12. M. R. Leuze
13-17. F. C. Maienschein
18-22. E. G. Ng
23. G. Ostrouchov
24. B. W. Peyton
25. C. H. Romine
26-30. R. C. Ward
31. P. H. Worley
32. A. Zucker
33. J. J. Dorning (Consultant)
34. R. M. Haralick (Consultant)
35. Central Research Library
36. ORNL Patent Office
37. K-25 Plant Library
38. Y-12 Technical Library /Document Reference Station
39. Laboratory Records - RC
40-41. Laboratory Records Department

EXTERNAL DISTRIBUTION

42. Dr. Donald M. Austin, Office of Scientific Computing, Office of Energy Research, ER-7, Germantown Building, U.S. Department of Energy, Washington, DC 20545

43. Dr. Robert G. Babb, Department of Computer Science and Engineering, Oregon Graduate Center, 19600 N.W. Walker Road, Beaverton, OR 97006

44. Lawrence J. Baker, Exxon Production Research Company, P.O. Box 2189, Houston, TX 77252-2189

45. Dr. Jesse L. Barlow, Department of Computer Science, Pennsylvania State University, University Park, PA 16802

46. Dr. Chris Bischof, Mathematics and Computer Science Division, Argonne National Laboratory, 9700 South Cass Avenue, Argonne, IL 60439

47. Prof. Ake Bjorck, Department of Mathematics, Linkoping University, Linkoping 58183, Sweden

48. Dr. James C. Browne, Department of Computer Sciences, University of Texas, Austin, TX 78712

49. Dr. Bill L. Buzbee, Scientific Computing Division, National Center for Atmospheric Research, P.O. Box 3000, Boulder, CO 80307

50. Dr. Donald A. Calahan, Department of Electrical and Computer Engineering, University of Michigan, Ann Arbor, MI 48109

51. Dr. Tony Chan, Department of Mathematics, University of California, Los Angeles, 405 Hilgard Avenue, Los Angeles, CA 90024

52. Dr. Jagdish Chandra, Army Research Office, P.O. Box 12211, Research Triangle Park, North Carolina 27709

53. Dr. Eleanor Chu, Department of Computer Science, University of Waterloo, Waterloo, Ontario, Canada N2L 3G1

54. Prof. Tom Coleman, Department of Computer Science, Cornell University, Ithaca, NY 14853

55. Dr. Paul Concus, Mathematics and Computing, Lawrence Berkeley Laboratory, Berkeley, CA 94720

56. Prof. Andy Conn, Department of Combinatorics and Optimization, University of Waterloo, Waterloo, Ontario, Canada N2L 3G1

57. Dr. Jane K. Cullum, IBM T. J. Watson Research Center, P.O. Box 218, Yorktown Heights, NY 10598

58. Dr. George Cybenko, Computer Science Department, University of Illinois, Urbana, IL 61801

59. Dr. George J. Davis, Department of Mathematics, Georgia State University, Atlanta, GA 30303

60. Dr. Jack J. Dongarra, Mathematics and Computer Science Division, Argonne National Laboratory, 9700 South Cass Avenue, Argonne, IL 60439

61. Dr. Iain Duff, CSS Division, Harwell Laboratory, Didcot, Oxon OX11 ORA, England

62. Prof. Pat Eberlein, Department of Computer Science, SUNY/Buffalo, Buffalo, NY 14260

63. Dr. Stanley Eisenstat, Department of Computer Science, Yale University, P.O. Box 2158 Yale Station, New Haven, CT 06520

64. Dr. Lars Elden, Department of Mathematics, Linkoping University, 581 83 Linkoping, Sweden

65. Dr. Howard C. Elman, Computer Science Department, University of Maryland, College Park, MD 20742

66. Dr. Albert M. Erisman, Boeing Computer Services, 565 Andover Park West, Tukwila, WA 98188

67. Dr. Peter Fenyes, General Motors Research Laboratory, Department 15, GM Technical Center, Warren, MI 48090

68. Prof. David Fisher, Department of Mathematics, Harvey Mudd College, Claremont, CA 91711

69. Dr. Geoffrey C. Fox, Booth Computing Center 158-79, California Institute of Technology, Pasadena, CA 91125

70. Dr. Paul O. Frederickson, Computing Division, Los Alamos National Laboratory, Los Alamos, NM 87545

71. Dr. Fred N. Fritsch, L-300, Mathematics and Statistics Division, Lawrence Livermore National Laboratory, P.O. Box 808, Livermore, CA 94550

72. Dr. Robert E. Funderlic, Department of Computer Science, North Carolina State University, Raleigh, NC 27650

73. Dr. Dennis B. Gannon, Computer Science Department, Indiana University, Bloomington, IN 47405

74. Dr. David M. Gay, Bell Laboratories, 600 Mountain Avenue, Murray Hill, NJ 07974

75. Dr. C. William Gear, Computer Science Department, University of Illinois, Urbana, Illinois 61801

76. Dr. W. Morven Gentleman, Division of Electrical Engineering, National Research Council, Building M-50, Room 344, Montreal Road, Ottawa, Ontario, Canada K1A 0R8

77. Dr. Alan George, Vice President, Academic and Provost, Needles Hall, University of Waterloo, Waterloo, Ontario, Canada N2L 3G1

78. Dr. John Gilbert, Xerox Palo Alto Research Center 3333 Coyote Hill Road Palo Alto, CA 94304

79. Dr. Jacob D. Goldstein, The Analytic Sciences Corporation, 55 Walkers Brook Drive, Reading, MA 01867

80. Prof. Gene H. Golub, Department of Computer Science, Stanford University, Stanford, CA 94305

81. Dr. Joseph F. Grcar, Division 8331, Sandia National Laboratories, Livermore, CA 94550

82. Dr. Per Christian Hansen, Technical University of Denmark, Danish University Computing Center, UCI-C Lyngby, Building 305, DK-2800 Lyngby, Denmark

83. Dr. Don E. Heller, Physics and Computer Science Department, Shell Development Co., P.O. Box 481, Houston, TX 77001

84. Dr. F. J. Helton, GA Technologies, P.O. Box 81608, San Diego, CA 92188

85. Dr. Charles J. Holland, Air Force Office of Scientific Research, Building 410, Bolling Air Force Base, Washington, DC 20332

86. Dr. Robert E. Huddleston, Computation Department, Lawrence Livermore National Laboratory, P.O. Box 808, Livermore, CA 94550

87. Dr. Ilse Ipsen, Department of Computer Science, Yale University, P.O. Box 2158 Yale Station, New Haven, CT 06520

88. Ms. Elizabeth Jessup, Department of Computer Science, Yale University, P.O. Box 2158, Yale Station, New Haven, CT 06520

89. Prof. Barry Joe, Department of Computer Science, University of Alberta, Edmonton, Alberta, Canada T6G 2H1

90. Dr. Harry Jordan, Department of Electrical and Computer Engineering, University of Colorado, Boulder, CO 80309

91. Dr. Bo Kagstrom, Institute of Information Processing, University of Umea, 5-901 87 Umea, Sweden

92. Dr. Hans Kaper, Mathematics and Computer Science Division, Argonne National Laboratory, 9700 South Cass Avenue, Argonne, IL 60439

93. Dr. Linda Kaufman, Bell Laboratories, 600 Mountain Avenue, Murray Hill, NJ 07974

94. Dr. Robert J. Kee, Applied Mathematics Division 8331, Sandia National Laboratories, Livermore, CA 94550

95. Ms. Virginia Klema, Statistics Center, E40-131, MIT, Cambridge, MA 02139

96. Dr. Richard Lau, Office of Naval Research, 1030 E. Green Street, Pasadena, CA 91101

97. Dr. Alan J. Laub, Department of Electrical and Computer Engineering, University of California, Santa Barbara, CA 93106

98. Dr. Robert L. Launer, Army Research Office, P.O. Box 12211, Research Triangle Park, North Carolina 27709

99. Dr. Charles Lawson, Applied Mathematics Group, Jet Propulsion Laboratory, California Institute of Technology, M/S 506-232, 4800 Oak Grove Drive, Pasadena, CA 91103

100. Prof. Peter D. Lax, Director, Courant Institute of Mathematical Sciences, New York University, 251 Mercer Street, New York, NY 10012

101. Dr. John G. Lewis, Boeing Computer Services, P.O. Box 24346, M/S 7L-21, Seattle, WA 98124-0346

102. Dr. Heather M. Liddell, Director, Center for Parallel Computing, Department of Computer Science and Statistics, Queen Mary College, University of London, Mile End Road, London E1 4NS, England

103. Dr. Joseph Liu, Department of Computer Science, York University, 4700 Keele Street, Downsview, Ontario, Canada M3J 1P3

104. Dr. Franklin Luk, Electrical Engineering Department, Cornell University, Ithaca, NY 14853

105. James G. Malone, General Motors Research Laboratories, Warren, Michigan 48090-9055

106. Dr. Thomas A. Manteuffel, Computing Division, Los Alamos National Laboratory, Los Alamos, NM 87545

107. Dr. Bernard McDonald, National Science Foundation, 1800 G Street, NW, Washington, DC 20550

108. Dr. Paul C. Messina, California Institute of Technology, Mail Code 158-79, Pasadena, CA 91125

109. Dr. Cleve Moler, Ardent Computers, 550 Del Ray Avenue, Sunnyvale, CA 94086

110. Dr. Brent Morris, National Security Agency, Ft. George G. Meade, MD 20755

111. Dr. Dianne P. O'Leary, Computer Science Department, University of Maryland, College Park, MD 20742

112. Maj. C. E. Oliver, Office of the Chief Scientist, Air Force Weapons Laboratory, Kirtland Air Force Base, Albuquerque, NM 87115

113. Dr. James M. Ortega, Department of Applied Mathematics, University of Virginia, Charlottesville, VA 22903

114. Prof. Chris Paige, Department of Computer Science, McGill University, 805 Sherbrooke Street W., Montreal, Quebec, Canada H3A 2K6

115. Dr. John F. Palmer, NCUBE Corporation, 915 E. LaVieve Lane, Tempe, AZ 85284

116. Prof. Roy P. Pargas, Department of Computer Science, Clemson University, Clemson, SC 29634-1906

117. Prof. Beresford N. Parlett, Department of Mathematics, University of California, Berkeley, CA 94720

118. Prof. Merrell Patrick, Department of Computer Science, Duke University, Durham, NC 27706

119. Dr. Robert J. Plemmons, Departments of Mathematics and Computer Science, North Carolina State University, Raleigh, NC 27650

120. Dr. Alex Pothen, Department of Computer Science, Pennsylvania State University, University Park, PA 16802

121. Dr. John K. Reid, CSS Division, Building 8.9, AERE Harwell, Didcot, Oxon, England OX11 0RA

122. Dr. John R. Rice, Computer Science Department, Purdue University, West Lafayette, IN 47907

123. Dr. Garry Rodrigue, Numerical Mathematics Group, Lawrence Livermore Laboratory, Livermore, CA 94550

124. Dr. Donald J. Rose, Department of Computer Science, Duke University, Durham, NC 27706

125. Dr. Ahmed H. Sameh, Computer Science Department, University of Illinois, Urbana, IL 61801

126. Dr. Michael Saunders, Systems Optimization Laboratory, Operations Research Department, Stanford University, Stanford, CA 94305

127. Dr. Robert Schreiber, Department of Computer Science, Rensselaer Polytechnic Institute, Troy, NY 12180

128. Dr. Martin H. Schultz, Department of Computer Science, Yale University, P.O. Box 2158 Yale Station, New Haven, CT 06520

129. Dr. David S. Scott, Intel Scientific Computers, 15201 N.W. Greenbrier Parkway, Beaverton, OR 97006

130. Dr. Lawrence F. Shampine, Mathematics Department, Southern Methodist University, Dallas, TX 75275

131. Dr. Kermit Sigmon, Department of Mathematics, University of Florida, Gainesville, FL 32611

132. Dr. Horst Simon, Mail Stop 258-5, NASA Ames Research Center, Moffett Field, CA 94035

133. Dr. Danny C. Sorensen, Mathematics and Computer Science Division, Argonne National Laboratory, 9700 South Cass Avenue, Argonne, IL 60439

134. Prof. G. W. Stewart, Computer Science Department, University of Maryland, College Park, MD 20742

135. Dr. Kosmo D. Tatalias, Atlantic Aerospace Electronics Corporation, 6404 Ivy Lane, Suite 300, Breenbelt, MD 20770-1406

136. Prof. Charles Van Loan, Department of Computer Science, Cornell University, Ithaca, NY 14853

137. Dr. Robert G. Voigt, ICASE, MS 132-C, NASA Langley Research Center, Hampton, VA 23665

138. Dr. Andrew B. White, Computing Division, Los Alamos National Laboratory, Los Alamos, NM 87545

139. Dr. Arthur Wouk, Army Research Office, P.O. Box 12211, Research Triangle Park, North Carolina 27709

140. Dr. Margaret Wright, Bell Laboratories, 600 Mountain Avenue, Murray Hill, NJ 07974

141. Dr. A. Yeremin, Department of Numerical Mathematics of the USSR Academy of Sciences, Gorki Street 11, Moscow, 103905, USSR

142. Office of Assistant Manager for Energy Research and Development, U.S. Department of Energy, Oak Ridge Operations Office, P.O. Box 2001 Oak Ridge, TN 37831-8600

143-152. Office of Scientific & Technical Information, P.O. Box 62, Oak Ridge, TN 37831