Journal of Inequalities and Applications
a SpringerOpen Journal

RESEARCH Open Access

Updating *QR* factorization procedure for solution of linear least squares problem with equality constraints

Salman Zeb[*] and Muhammad Yousaf

[*]Correspondence:
zebsalman@gmail.com
Department of Mathematics,
University of Malakand, Dir (Lower),
Chakdara, Khyber Pakhtunkhwa,
Pakistan

Abstract
In this article, we present a *QR* updating procedure as a solution approach for linear least squares problem with equality constraints. We reduce the constrained problem to unconstrained linear least squares and partition it into a small subproblem. The *QR* factorization of the subproblem is calculated and then we apply updating techniques to its upper triangular factor *R* to obtain its solution. We carry out the error analysis of the proposed algorithm to show that it is backward stable. We also illustrate the implementation and accuracy of the proposed algorithm by providing some numerical experiments with particular emphasis on dense problems.

MSC: 65-XX; 65Fxx; 65F20; 65F25

Keywords: *QR* factorization; orthogonal transformation; updating; least squares problems; equality constraints

1 Introduction

We consider the linear least squares problem with equality constraints (LSE)

$$\min_x \|Ax - b\|_2, \quad \text{subject to } Bx = d, \tag{1}$$

where $A \in R^{m\times n}$, $b \in R^m$, $B \in R^{p\times n}$, $d \in R^p$, $x \in R^n$ with $m + p \geq n \geq p$ and $\|\cdot\|_2$ denotes the Euclidean norm. It arises in important applications of science and engineering such as in beam-forming in signal processing, curve fitting, solutions of inequality constrained least squares problems, penalty function methods in nonlinear optimization, electromagnetic data processing and in the analysis of large scale structure [1–8]. The assumptions

$$\operatorname{rank}(B) = p \quad \text{and} \quad \operatorname{null}(A) \cap \operatorname{null}(B) = \{0\} \tag{2}$$

ensure the existence and uniqueness of solution for problem (1).

The solution of LSE problem (1) can be obtained using direct elimination, the nullspace method and method of weighting. In direct elimination and nullspace methods, the LSE problem is first transformed into unconstrained linear least squares (LLS) problem and then it is solved via normal equations or *QR* factorization methods. In the method of weighting, a large suitable weighted factor γ is selected such that the weighted residual

$\gamma(d - Bx)$ remains of the same size as that of the residual $b - Ax$ and the constraints is satisfied effectively. Then the solution of the LSE problem is approximated by solving the weighted LLS problem. In [9], the author studied the method of weighting for LSE problem and provided a natural criterion of selecting the weighted factor γ such that $\gamma \geq \|A\|_2/\|B\|_2\epsilon_M$, where ϵ_M is the rounding unit. For further details as regards methods of solution for LSE problem (1), we refer to [2, 3, 7, 10–13].

Updating is a process which allow us to approximate the solution of the original problem without solving it afresh. It is useful in applications such as in solving a sequence of modified related problems by adding or removing data from the original problem. Stable and efficient methods of updating are required in various fields of science and engineering such as in optimization and signal processing [14], statistics [15], network and structural analysis [16, 17] and discretization of differential equations [18]. Various updating techniques based on matrix factorization for different kinds of problems exist in the literature [3, 11, 13, 19–22]. Hammarling and Lucas [23] discussed updating the *QR* factorization with applications to LLS problem and presented updating algorithms which exploited LEVEL 3 BLAS. Yousaf [24] studied repeated updating based on *QR* factorization as a solution tool for LLS problem. Parallel implementation on GPUs of the updating *QR* factorization algorithms presented in [23] is performed by Andrew and Dingle [25]. Zhdanov and Gogoleva [26] studied augmented regularized normal equations for solving LSE problems. Zeb and Yousaf [27] presented an updating algorithm by repeatedly updating both factors of the *QR* factorization for the solutions of LSE problems.

In this article, we consider the LSE problem (1) in the following form:

$$\min_{x(\gamma)} \left\| \begin{pmatrix} \gamma B \\ A \end{pmatrix} x - \begin{pmatrix} \gamma d \\ b \end{pmatrix} \right\|_2, \tag{3}$$

which is an unconstrained weighted LLS problem where $\gamma \geq \|A\|_2/\|B\|_2\epsilon_M$ given in [9] and approximated its solution by updating Householder *QR* factorization. It is well known that Householder *QR* factorization is backward stable [11, 12, 28, 29]. The conditions given in (2) ensure that problem (3) is a full rank LLS problem. Hence, there exist a unique solution $x(\gamma)$ of problem (3) which approximated the solution x_{LSE} of the LSE problem (1) such that $\lim_{\gamma\to\infty} x(\gamma) = x_{\text{LSE}}$. In our proposed technique, we reduced problem (3) to a small subproblem using a suitable partitioning strategy by removing blocks of rows and columns. The *QR* factorization of the subproblem is calculated and its *R* factor is then updated by appending the removed block of columns and rows, respectively, to approximate the solution of problem (1). The presented approach is suitable for solution of dense problems and also applicable for those applications where we are adding new data to the original problem and *QR* factorization of the problem matrix is already available.

We organized this article as follows. Section 2 contains preliminaries related to our main results. In Section 3, we present the *QR* updating procedure and algorithm for solution of LSE problem (1). The error analysis of the proposed algorithm is provided in Section 4. Numerical experiments and comparison of solutions is given in Section 5, while our conclusion is given in Section 6.

2 Preliminaries

This section contains important concepts which will be used in the forthcoming sections.

2.1 The method of weighting

This method is based on the observations that while solving LSE problem (1) we are interested that some equations are to be exactly satisfied. This can be achieved by multiplying large weighted factor γ to those equations. Then we can solve the resulting weighted LLS problem (3). The method of weighting is useful as it allows for the use of subroutines for LLS problems to approximate the solution of LSE problem. However, the use of the large weighted factor γ can compromise the conditioning of the constrained matrix. In particular, the method of normal equations when applied to problem (3) fails for large values of γ in general. For details, see [3, 11–13].

2.2 *QR* factorization and householder reflection

Let

$$X = QR \tag{4}$$

be the QR factorization of a matrix $X \in \mathcal{R}^{m\times n}$ where $Q \in \mathcal{R}^{m\times m}$ and $R \in \mathcal{R}^{m\times n}$. This factorization can be obtained using Householder reflections, Givens rotations and the classical/modified Gram-Schmidt orthogonalization. The Householder and Givens QR factorization methods are backward stable. For details as regards QR factorization, we refer to [11, 12].

Here, we briefly discuss the Householder reflection method due to our requirements. For a non-zero Householder vector $v \in \mathcal{R}^n$, we define the matrix $H \in \mathcal{R}^{n\times n}$ as

$$H = I_n - \tau v v^T, \quad \tau = \frac{2}{v^T v}, \tag{5}$$

which is called the Householder reflection or Householder matrix. Householder matrices are symmetric and orthogonal. For a non-zero vector u, the Householder vector v is simply defined as

$$v = u \pm \|u\|_2 e_k,$$

such that

$$Hu = u - \tau v v^T u = \mp \alpha e_k, \tag{6}$$

where $\alpha = \|u\|_2$ and e_k denotes the kth unit vector in $\mathcal{R}^n$ in the following form:

$$e_k(i) = \begin{cases} 1 & \text{if } i = k, \\ 0 & \text{otherwise.} \end{cases}$$

In our present work given in next section, we will use the following algorithm for computation of Householder vector v. This computation is based on Parlett's formula [30]:

$$v = u_1 - \|u\|_2 = \frac{u_1^2 - \|u\|_2^2}{x_1 + \|u\|_2} = \frac{-\|u\|_2^2}{u_1 + \|u\|_2},$$

where $u_1 > 0$ is the first element of the vector $u \in \mathcal{R}^n$. Then we compute the Householder vector v as follows (Algorithm 1).

Algorithm 1 Computation of parameter τ and Householder vector v [12]

```
Input: $u_1, u$
Output: $v, \tau$
1: $\alpha = \|u\|_2^2$
2: $v = u, v(1) = 1$
3: if $(\alpha = 0)$ then
4:     $\tau = 0$
5: else
6:     $\beta = \sqrt{u_1^2 + \alpha}$
7: end if
8: if $u_1 \leq 0$ then
9:     $v(1) = u_1 - \beta$
10: else
11:     $v(1) = -\alpha/(u_1 + \beta)$
12: end if
13: $\tau = 2v(1)^2/(\alpha + v(1)^2)$
14: $v = v/v(1)$
```

3 Updating *QR* factorization procedure for solution of LSE problem

Let

$$E = \begin{pmatrix} \gamma B \\ A \end{pmatrix} \in R^{(m+p)\times n} \quad \text{and} \quad f = \begin{pmatrix} \gamma d \\ b \end{pmatrix} \in R^{m+p}.$$

Then we can write problem (3) as

$$\min_{x(\gamma)} \|Ex - f\|_2. \tag{7}$$

Here, we will reduce problem (7) to a small incomplete subproblem using suitable partition process. In partition process, we remove blocks of rows and columns from the problem matrix E considered in (7) and from its corresponding right-hand side (RHS) without involving any arithmetics. That is, we consider

$$E = \begin{pmatrix} E(1:j-1, 1:n) \\ E(j:j+r, 1:n) \\ E(j+r+1:m+p, 1:n) \end{pmatrix} \quad \text{and} \quad f = \begin{pmatrix} f(1:j-1) \\ f(j:j+r) \\ f(j+r+1:m+p) \end{pmatrix}, \tag{8}$$

and removing blocks of rows from both E and f in equation (8) as

$$G_r = E(j:j+r, 1:n) \in \mathcal{R}^{r\times n} \quad \text{and} \quad f_r = f(j:j+r), \tag{9}$$

we get

$$E_1 = \begin{pmatrix} E(1:j-1, 1:n) \\ E(j+r+1:m+p, 1:n) \end{pmatrix}, \qquad f_1 = \begin{pmatrix} f(1:j-1) \\ f(j+r+1:m+p) \end{pmatrix}. \tag{10}$$

Algorithm 2 Calculating the QR factorization and computing matrix U_c

```
Input: $E_2 \in \mathcal{R}^{m_2 \times n_2}, f_2 \in \mathcal{R}^{m_2}, G_c \in \mathcal{R}^{m_2 \times c}$
Output: $R_2 \in \mathcal{R}^{m_2 \times n}, g_2 \in \mathcal{R}^{m_2}, U_c \in \mathcal{R}^{m_2 \times c}$
 1: $R_2 \longleftarrow E_2$
 2: for $k = 1$ to $\min(m_2, n_2)$ do
 3:     $[v, \tau, E_2(k,k)] = \text{house}(E_2(k,k), E_2(k+1:m_2,k))$
 4:     $V = E_2(k,k+1:n_2) + v^T E_2(k+1:m_2,k+1:n_2)$
 5:     $R_2(k,k+1:n_2) = E_2(k,k+1:n_2) - \tau V$
 6:     if $k < n$ then
 7:         $R_2(k+1:m_2,k+1:n_2) = E_2(k+1:m_2,k+1:n_2) - \tau v V$
 8:     end if
 9:     $g_2(j:m_2) = f_2(k:m_2) - \tau \binom{1}{v} \begin{pmatrix} 1 & v^T \end{pmatrix} f_2(k:m_2)$
10:     $U_c(k:m_2,k:end) = G_c(k:m_2,k:end) - \tau \binom{1}{v} \begin{pmatrix} 1 & v^T \end{pmatrix} U_c(k:m_2,k:end)$
11: end for
```

Hence, we obtain the following problem:

$$\min_{x_1} \|E_1 x_1 - f_1\|_2, \quad E_1 \in \mathcal{R}^{m_1 \times n_1}, f_1 \in \mathcal{R}^{m_1}, x_1 \in \mathcal{R}^{n_1}, \tag{11}$$

where $m_1 = m + p - r$, $n_1 = n$, and $f_r \in \mathcal{R}^r$. Furthermore, by removing block of columns $G_c = E_1(:,j:j+c)$ from the jth position by considering the partition of E_1 in the incomplete problem (11) as

$$E_2 = \begin{pmatrix} E_1(:,1:j-1) & E_1(:,j:j+c) & E_1(:,j+c+1:n) \end{pmatrix}, \tag{12}$$

we obtain the following reduced subproblem:

$$\min_{x_2} \|E_2 x_2 - f_2\|_2, \quad E_2 \in \mathcal{R}^{m_2 \times n_2}, f_2 \in \mathcal{R}^{m_2}, x_2 \in \mathcal{R}^{n_2}, \tag{13}$$

where $E_2 = [E_1(:,1:j-1), E_1(:,j+1:n_1)]$, $n_2 = n_1 - c$, $m_2 = m_1$, and $f_1 = f_2$.

Now, we calculated the QR factorization of the incomplete subproblem (13) is order to reduce it to the upper triangular matrix R_2 using the following algorithm (Algorithm 2).

Here house denotes the Householder algorithm and the Householder vectors are calculated using Algorithm 1 and V is a self-explanatory intermediary variable.

Hence, we obtain

$$R_2 = H_{n_2} \cdots H_1 E_2, \qquad g_2 = H_{n_2} \cdots H_1 f_2 \tag{14}$$

and

$$U_c = H_{n_2} \cdots H_1 G_c.$$

Here, the QR factorization can also be obtained directly using the MATLAB built-in command *qr* but it is not preferable due to its excessive storage requirements for orthogonal matrix Q and by not necessarily providing positive sign of diagonal elements in the matrix R_2 [12].

Algorithm 3 Updating R_2 factor after appending a block of columns G_c

```
Input: $R_2 \in \mathcal{R}^{m_2 \times n_2}, G_c \in \mathcal{R}^{m_2 \times c}, g_2 \in \mathcal{R}^{m_2}$
Output: $R_1 \in \mathcal{R}^{m_2 \times (n_2+c)}, g_1 \in \mathcal{R}^{m_2}$
 1: $G_c(1:m_2,1:c) \longleftarrow R_2(1:m_2,n_2+1:n_2+c)$
 2: if $(m_2 \geq n_2)$ then
 3:     $R_1 \longleftarrow \text{triu}(R_2)$
 4:     $g_1 \longleftarrow g_2$
 5: else
 6:     for $k = j$ to $\min(m_2, n_2+c)$ do
 7:         $[v, \tau, R_2(k,k)] = \text{house}(R_2(k,k), R_2(k+1:m_2,k))$
 8:         $V = R_2(k,k+1:n_2+c) + v^T R_2(k+1:m_2,k+1:n_2+c))$
 9:         $R_1(k,k+1:n_2+c) = R_2(k,k+1:n_2+c) - \tau V$
10:         if $k < n_2 + c$ then
11:             $R_1(k+1:m_2,k+1:n_2+c) = R_2(k+1:m_2,k+1:n_2+c) - \tau v V$
12:         end if
13:         $g_1(k:m_2) = g_2(k:m_2) - \tau \binom{1}{v} \begin{pmatrix} 1 & v^T \end{pmatrix} g_2(k:m_2)$
14:     end for
15:     $R_1 = \text{triu}(R_2)$
16: end if
```

To obtain the solution of problem (7), we need to update the upper triangular matrix R_2. For this purpose, we append the updated block of columns G_c to the R_2 factor in (14) at the jth position as follows:

$$R_{2c} = \begin{pmatrix} R_2(:,1:j-1) & G_c(:,j:j+c) & R_2(:,j+c+1:n) \end{pmatrix}. \tag{15}$$

Here, if the R_2c factor in (15) is upper trapezoidal or in upper triangular form then no further calculation is required, and we get $R_1 = R_{2c}$. Otherwise, we will need to reduce equation (15) to the upper triangular factor R_1 by introducing the Householder matrices $H_{n_2+c}, \ldots, H_j$:

$$R_1 = H_{n_2+c} \cdots H_j R_{2c} \quad \text{and} \quad g_1 = H_{n_2+c} \cdots H_j g_2, \tag{16}$$

where $R_1 \in \mathcal{R}^{m_1 \times n_1}$ is upper trapezoidal for $m_1 < n_1$ or it is an upper triangular matrix. The procedure for appending the block of columns and updating of the R_2 factor in algorithmic form is given as follows (Algorithm 3).

Here the term triu denotes the upper triangular part of the concerned matrix and V is the intermediary variable.

Now, we append a block of rows G_r to the R_1 factor and f_r to its corresponding RHS at the jth position in the following manner.

$$R_r = \begin{pmatrix} R_1(1:j-1,:) \\ G_r(1:r,:) \\ R_1(j:m_1,:) \end{pmatrix}, \qquad g_r = \begin{pmatrix} g_1(1:j-1) \\ f_r(1:r) \\ g_1(j:m_1) \end{pmatrix}.$$

Algorithm 4 Updating R_1 factor after appending a block of rows G_r

```
Input: $R_1 \in \mathcal{R}^{m_1 \times n}, G_r \in \mathcal{R}^{r \times n}, g_1 \in \mathcal{R}^{m_1}, f_r \in \mathcal{R}^{m_1}$
Output: $\tilde{R} \in \mathcal{R}^{(m_1+r) \times n}, \tilde{g} \in \mathcal{R}^{m_1+r}$
1:  for $k = 1$ to $\min(m_1, n)$ do
2:      $[v, \tau, R_1(k,k)] = \text{house}(R_1(k,k), G_r(1:r,k))$
3:      $V = R_1(k, k+1:n) + v^T G_r(1:r, k+1:n)$
4:      $\tilde{R}(k, k+1:n) = R_1(k, k+1:n) - \tau V$
5:      if $k < n$ then
6:          $G_r(1:r, k+1:n) = G_r(1:r, k+1:n) - \tau v V$
7:      end if
8:      $g_k = g(k)$
9:      $g(k) \equiv (1-\tau) g(k) - \tau v^T f_r(1:r)$
10:     $f_r(1:r) \equiv f_r(1:r) - \tau v g_k - \tau v v^T f_r(1:r)$
11: end for
12: if $m_1 < n$ then
13:     $[Q_r, R_r] = qr(G_r(:, (m_1+1:n)))$
14:     $R_r(m_1+1 : m_1+r, m_1+1:n) \longleftarrow \tilde{R}_r$
15:     $f_r \equiv Q_r^T f_r$
16: end if
17: $\tilde{R} \longleftarrow \binom{R}{0}$
18: $\tilde{g} \longleftarrow \binom{g}{f_r}$
```

We use the permutation matrix P in order to bring the block of rows G_r and f_r at the bottom position if required. Then

$$PR_r = \begin{pmatrix} R_1 \\ G_r \end{pmatrix}, \qquad Pg_r = \begin{pmatrix} g_1 \\ f_r \end{pmatrix}. \tag{17}$$

The equation (17) is reduced to upper triangular form by constructing the matrices $H_1, \ldots, H_n$ using Algorithm 1 as given by

$$\tilde{R} = H_n H_{n-1} \cdots H_1 \begin{pmatrix} R_1 \\ G_r \end{pmatrix}$$

and

$$\tilde{g} = H_n H_{n-1} \cdots H_1 \begin{pmatrix} g_1 \\ f_r \end{pmatrix}.$$

The procedure is performed in algorithmic form as follows (Algorithm 4).

Here *qr* is the MATLAB command of *QR* factorization and V is a self-explanatory intermediary variable.

The solution of problem (7) can then be obtained by applying the MATLAB built-in command *backsub* for a back-substitution procedure.

The description of *QR* updating algorithm in compact form is given as follows (Algorithm 5).

Algorithm 5 Updating QR algorithm for solution of LSE problem

```
Input: $A \in \mathcal{R}^{m\times n}, b \in \mathcal{R}^m, B \in \mathcal{R}^{p\times n}, d \in \mathcal{R}^p$
Output: $x_{\text{LSE}} \in \mathcal{R}^n$
1: $E = \binom{\gamma B}{A}$ and $f = \binom{\gamma d}{b}$
2: $[E_1, f_1, G_r, E_2, G_c, f_r] \longleftarrow$ partition$(E, f)$
3: $[R_2, g_2, U_c] \longleftarrow QR$ factorization of subproblem $(E_2, f_1, G_c)$
4: $[R_1, g_1] \longleftarrow$ appendbcols$(R_2, g_2, U_c)$
5: $[\tilde{R}, \tilde{g}] \longleftarrow$ appendbrows$(R_1, G_r, g_1, f_r)$
6: $x_{\text{LSE}} \longleftarrow$ backsub$(\tilde{R}(1:n, 1:n), \tilde{g}(1:n))$
```

Here, Algorithm 5 for a solution of LSE problem (1) calls upon the partition process, Algorithms 2, 3, 4 and MATLAB command *backsub* for back-substitution procedure, respectively.

4 Error analysis

In this section, we will study the backward stability of our proposed Algorithm 5. The mainstay in our presented algorithm for the solution of LSE problem is the updating procedure. Therefore, our main concern is to study the error analysis of the updating steps. For others, such as the effect of using the weighting factor, finding the QR factorization and for the back-substitution procedure, we refer the reader to [28, 31]. Here, we recall some important results without giving their proofs and refer the reader to [28].

We will consider the following standard model of floating point arithmetic in order to analyze the rounding errors effects in the computations:

$$fl(x \text{ op } y) = (x \text{ op } y)(1 + \eta), \qquad |\eta| \le \epsilon_M, \qquad \text{op} = +, -, *, /, \tag{18}$$

where ϵ_M is the unit roundoff. The value of ϵ_M is of order 10^{-8} in single precision computer arithmetic, while in double precision it is of the order 10^{-16}. Moreover, for addition and subtraction in the absence of guard integers, we can write model (18) as follows:

$$fl(x \pm y) = x(1 + \eta_1) \pm y(1 + \eta_2), \qquad |\eta_i| \le \epsilon_M, \quad i = 1, 2.$$

Lemma 4.1 ([28]) *If* $|\eta_i| \le \epsilon_M, \delta_i = \pm 1$ *for* $i = 1, 2, \ldots, n$ *and* $n\epsilon_M < 1$, *then*

$$\prod_{i=1}^{n} (1 + \eta_i)^{\delta_i} = 1 + \phi_n,$$

where

$$|\phi_n| \le \frac{n\epsilon_M}{1 - n\epsilon_M} = \gamma_n.$$

Here, we will use the constants γ_n and $\tilde{\gamma}_{cn}$ for convenience as adopted in [28] where these are defined by

$$\gamma_n = \frac{n\epsilon_M}{1 - n\epsilon_M}, \quad \text{assuming } n\epsilon_M < 1, \tag{19}$$

and

$$\tilde{\gamma}_{cn} = \frac{nc\epsilon_M}{1 - nc\epsilon_M}, \quad \text{assuming } nc\epsilon_M < 1, \tag{20}$$

where c is a small positive integer.

Lemma 4.2 ([28]) *Let $x, y \in \mathcal{R}^n$ and consider the inner product $x^T y$. Then*

$$fl(x^T y) = x^T(y + \triangle y), \qquad |\triangle y| \le \gamma_n |y|.$$

Lemma 4.3 ([28]) *Let ϕ_k and γ_k be defined for any positive integer k. Then for positive integers j and k the following relations hold*:

$$(1 + \phi_k)(1 + \phi_j) = 1 + \phi_{k+j}, \tag{21}$$

$$\frac{(1 + \phi_k)}{(1 + \phi_j)} = \begin{cases} 1 + \phi_{k+j}, & j \le k, \\ 1 + \phi_{k+2j}, & j > k, \end{cases} \tag{22}$$

$$j\gamma_k \le \gamma_{jk}, \tag{23}$$

$$k\gamma_k + \epsilon_M \le \gamma_{k+1}, \tag{24}$$

$$\gamma_k + \gamma_j + \gamma_k\gamma_j \le \gamma_{k+j}. \tag{25}$$

Lemma 4.4 ([28]) *Considering the construction of $\tau \in \mathcal{R}$ and $v \in \mathcal{R}^n$ given in Section* 2, *then the computed $\tilde{\tau}$ and $\tilde{v}$ in floating point arithmetic satisfy*

$$\tilde{v}(2:n) = v(2:n)$$

and

$$\tilde{\tau} = \tau(1 + \tilde{\phi}_n) \quad \textit{and} \quad \tilde{v}_1 = v_1(1 + \tilde{\phi}_n),$$

where $\tilde{\phi}_n \le \tilde{\gamma}_n$.

Here, we represent the Householder transformation as $I - vv^T$, which requires $\|v\|_2 = \sqrt{2}$. Therefore, by redefining $v = \sqrt{\tau}v$ and $\tau = 1$ using Lemma 4.4, we have

$$\tilde{v} = v + \triangle v, \qquad |\triangle v| \le \tilde{\gamma}_m v \quad \text{for } v \in \mathcal{R}^m, \|v\|_2 = \sqrt{2}. \tag{26}$$

Lemma 4.5 ([29]) *Considering the computation of $y = \tilde{H}b = (I - \tilde{v}\tilde{v}^T)b = b - \tilde{v}(\tilde{v}^T b)$ for $b \in \mathcal{R}^m$ and $\tilde{v} \in \mathcal{R}^m$ satisfies* (26). *Then the computed $\tilde{y}$ satisfies*

$$\tilde{y} = (H + \triangle H)b, \qquad \|\triangle H\|_F \le \tilde{\gamma}_m,$$

where $H = I - vv^T$ and $\|\cdot\|_F$ denotes the Frobenius norm.

Lemma 4.6 ([28]) *Let $H_k = I - v_k v_k^T \in \mathcal{R}^{m\times m}$ be the Householder matrix and we define the sequence of transformations*

$$X_{k+1} = H_k X_k, \quad k = 1 : r,$$

where $X_1 = X \in \mathcal{R}^{m\times n}$. Furthermore, it is assumed that these transformations are performed using computed Householder vectors $\tilde{v}_k \approx v_k$ and satisfy Lemma 4.4. Then we have

$$\tilde{X}_{r+1} = Q^T(X + \triangle X), \tag{27}$$

where $Q^T = H_r, \ldots, H_1$, and

$$\|\triangle x_j\|_2 \le r\tilde{\gamma}_m \|x_j\|_2, \quad j = 1 : n.$$

Theorem 4.7 ([28]) *Let $\tilde{R} \in \mathcal{R}^{m\times n}$ be the computed upper trapezoidal QR factor of $X \in \mathcal{R}^{m\times n}$ ($m \ge n$) obtained via Householder QR algorithm by Lemma 4.6. Then there exists an orthogonal matrix $Q \in \mathcal{R}^{m\times m}$ such that*

$$X + \triangle X = Q\tilde{R},$$

where

$$\|\triangle x_j\|_2 \le \tilde{\gamma}_{mn} \|x_j\|_2, \quad j = 1 : n. \tag{28}$$

Lemma 4.8 ([28]) *Let $X \in \mathcal{R}^{m\times n}$ and $H = I - \tau v v^T \in R^{m\times m}$ be the Householder matrix. Also, assuming that the computation of HX is performed using computed $\tilde{\tau}$ and $\tilde{v}$ such that it satisfies Lemma 4.4. Then, from Theorem 4.7, we have*

$$fl(HX) = H(X + \triangle X), \qquad \|\triangle X\|_F \le \gamma_{cm}\|X\|_F. \tag{29}$$

4.1 Backward error analysis of proposed algorithm

To appreciate the backward stability of our proposed Algorithm 5, we first need to carry out the error analysis of Algorithms 3 and 4. For this purpose, we present the following.

Theorem 4.9 *The computed factor $\tilde{R}$ in Algorithm 4 satisfies*

$$\tilde{R} = Q^T\left[\begin{pmatrix} R_1 \\ G_r \end{pmatrix} + e_r\right] \quad \text{and} \quad \|e_r\|_F \le \tilde{\gamma}_{n(r+1)} \left\|\begin{pmatrix} R_1 \\ G_r \end{pmatrix}\right\|_F,$$

where $G_r \in \mathcal{R}^{r\times n}$ is the appended block of rows to the R_1 factor and $Q = H_1 H_2 \cdots H_n$.

Proof Let the Householder matrix H_j have zeros on the jth column of the matrix $\left(\begin{smallmatrix} \tilde{r}_{jj} \\ G_{rj} \end{smallmatrix}\right)$. Then using Lemma 4.6, we have

$$H_1\left[\begin{pmatrix} R_1 \\ G_r \end{pmatrix} + e_1\right] = \begin{pmatrix} \tilde{R}_{1r} \\ \tilde{G}_{1r} \end{pmatrix}, \tag{30}$$

where

$$\|e_1\|_F \leq \tilde{\gamma}_{r+1} \left\| \begin{pmatrix} R_1 \\ G_r \end{pmatrix} \right\|_F .$$

Similarly,

$$\begin{aligned} H_2 \left[\begin{pmatrix} \tilde{R}_{1r} \\ \tilde{G}_{1r} \end{pmatrix} + e_2 \right] = \begin{pmatrix} \tilde{R}_{2r} \\ \tilde{G}_{2r} \end{pmatrix} \\ = H_2 H_1 \left[\begin{pmatrix} R_1 \\ G_r \end{pmatrix} + e_r^{(2)} \right], \end{aligned}$$

and

$$\|e_2\|_F \leq \tilde{\gamma}_{r+1} \left\| \begin{pmatrix} \tilde{R}_{1r} \\ \tilde{G}_{1r} \end{pmatrix} \right\|_F ,$$

where

$$\begin{aligned} \left\|e_r^{(2)}\right\|_F &= \|e_1 + e_2\|_F, \\ \left\|e_r^{(2)}\right\|_F &\leq \|e_1\|_F + \|e_2\|_F, \\ &\leq \tilde{\gamma}_{r+1} \left\| \begin{pmatrix} R_1 \\ G_r \end{pmatrix} \right\|_F + \tilde{\gamma}_{r+1} \left\| \begin{pmatrix} \hat{R}_{1r} \\ \hat{G}_{1r} \end{pmatrix} \right\|_F . \end{aligned}$$

From equation (30), we have

$$\begin{aligned} \left\| \begin{pmatrix} \hat{R}_{1r} \\ \hat{G}_{1r} \end{pmatrix} \right\|_F &\leq \left\| \begin{pmatrix} R_1 \\ G_r \end{pmatrix} \right\|_F + e_1 \\ &\leq \left\| \begin{pmatrix} R_1 \\ G_r \end{pmatrix} \right\|_F + \tilde{\gamma}_{r+1} \left\| \begin{pmatrix} R_1 \\ G_r \end{pmatrix} \right\|_F . \end{aligned}$$

This implies that

$$\begin{aligned} \|e_r^{(2)}\|_F &\leq \tilde{\gamma}_{r+1} \left\| \begin{pmatrix} R_1 \\ G_r \end{pmatrix} \right\|_F + \tilde{\gamma}_{r+1}(1 + \tilde{\gamma}_{r+1}) \left\| \begin{pmatrix} R_1 \\ G_r \end{pmatrix} \right\|_F \\ &\leq \left(\tilde{\gamma}_{r+1} + \tilde{\gamma}_{r+1}(1 + \tilde{\gamma}_{r+1})\right) \left\| \begin{pmatrix} R_1 \\ G_r \end{pmatrix} \right\|_F \\ &\leq 2\tilde{\gamma}_{r+1} \left\| \begin{pmatrix} R_1 \\ G_r \end{pmatrix} \right\|_F , \end{aligned}$$

where we ignored $(\tilde{\gamma}_{r+1})^2$ as $\tilde{\gamma}_{r+1}$ is very small.

Continuing in the same fashion till the nth Householder reflection, we have

$$\tilde{R} = H_n H_{n-1} \cdots H_1 \left[\begin{pmatrix} R_1 \\ G_r \end{pmatrix} + e_r \right],$$

where

$$\|e_r\|_F = \|e_r^{(n)}\|_F \le n\tilde{\gamma}_{r+1} \left\| \begin{pmatrix} R_1 \\ G_r \end{pmatrix} \right\|_F,$$

or, by using equation (23) of Lemma 4.3, we can write

$$\|e_r\|_F \le \tilde{\gamma}_{n(r+1)} \left\| \begin{pmatrix} R_1 \\ G_r \end{pmatrix} \right\|_F,$$

which is the required result. □

Theorem 4.10 *The backward error for the computed factor R_1 in Algorithm* 3 *is given by*

$$R_1 = \tilde{Q}^T\big[(R_2, U_c) + \tilde{e}_c\big],$$

where $U_c \in \mathcal{R}^{(m+p)\times c}$ is the appended block of columns to the right-end of R_2 factor, $\|\tilde{e}_c\|_F \le \tilde{\gamma}_{(m+p)c}\|U_c\|_F$ and $\tilde{Q} = H_{n_2+1}\cdots H_{n_2+c}$.

Proof To prove the required result, we proceed similar to the proof of Theorem 4.9, and obtain

$$\|e_c^2\|_F \le 2\tilde{\gamma}_{c1}\big\|\big[(R_2, U_c) + \tilde{e}_c\big]\big\|_F.$$

This implies that the error in the whole process of appending the block of columns to the R_2 factor is given by

$$R_1 = H_{n_2+c}\cdots H_{n_2+1}\big[(R_2, U_c) + \tilde{e}_c\big],$$

where

$$\|e_c\|_F \le \tilde{\gamma}_{(m+p)c}\|U_c\|_F.$$ □

Theorem 4.11 *Let the LSE problem* (1) *satisfy the conditions given in* (2). *Then Algorithm* 5 *solves the LSE problem with computed $\tilde{Q} = H_1\cdots H_n$ and $\tilde{R}$ which satisfies*

$$\|I - \tilde{Q}^T\tilde{Q}\|_F \le \sqrt{n}\tilde{\gamma}_{(m+p)n} \tag{31}$$

and

$$\|E - \tilde{Q}\tilde{R}\|_F \le \sqrt{n}\tilde{\gamma}_{(m+p)n}\|E\|_F. \tag{32}$$

Proof To study the backward error analysis of Algorithm 5, we consider the reduced subproblem matrix $E_2 \in \mathcal{R}^{m_2\times n_2}$ from (13) and let $\hat{e}_2$ be its backward error in the computed QR factorization. Then from Lemma 4.8, we have

$$Q_2^T(E_2 + \hat{e}_2) = R_2 \quad \text{and} \quad \|\hat{e}_2\|_F \le \tilde{\gamma}_{m_2n_2}\|E_2\|_F.$$

As in our proposed Algorithm 5, we appended a block of columns $U_c = Q_2^T G_c$ to the R_2 factor, then by using Theorem 4.10, we have

$$R_1 = H_{n_2+1} \cdots H_{n_2+c}\big[(R_2, U_c) + e_c\big],$$

where $\|e_c\|_F \leq \tilde{\gamma}_{m_2c}\|U_c\|_F$.

Therefore, by simplifying

$$\|\hat{e}_1\|_F = \big\|Q_1^T Q_2^T \hat{e}_2 + Q_1^T e_c\big\|_F,$$

where $Q_1^T = H_{n_2+1} \cdots H_{n_2+c}$ and $\hat{e}_2$ is the error of computing the QR factorization, we obtain

$$\begin{aligned}
\|\hat{e}_1\|_F &= \big\|Q_1^T Q_2^T \hat{e}_2 + Q_1^T e_c\big\|_F \\
&\leq \big\|Q_1^T\big\|_F \big\|Q_2^T\big\|_F \|\hat{e}_2\|_F + \big\|Q_1^T\big\|_F \|e_c\|_F \\
&\leq \|\hat{e}_2\|_F + \|e_c\|_F \\
&\leq \tilde{\gamma}_{m_2n_2}\|E_2\|_F + \tilde{\gamma}_{m_2c}\|U_c\|_F \\
&\leq [\tilde{\gamma}_{m_2n_2} + \tilde{\gamma}_{m_2c}]\max\big(\|E_2\|_F, \|U_c\|_F\big).
\end{aligned}$$

Hence, we have

$$\|\hat{e}_1\|_F \leq [\tilde{\gamma}_{m_2n_2} + \tilde{\gamma}_{m_2c}]\max\big(\|E_2\|_F, \|U_c\|_F\big),$$

which is the total error at this stage of appending the block of columns and its updating. Furthermore, we appended the block of rows G_r to the computed factor R_1 in our algorithm and, by using Theorem 4.9, we obtain

$$\tilde{R} = H_n \cdots H_1 \left[\begin{pmatrix} R_1 \\ G_r \end{pmatrix} + e_r\right],$$

where

$$\|e_r\|_F \leq \tilde{\gamma}_{(r+1)n} \left\|\begin{pmatrix} R_1 \\ G_r \end{pmatrix}\right\|_F,$$

is the error for appending the block of rows G_r.

Hence, the total error e for the whole updating procedure in Algorithm 5 can be written as

$$\begin{aligned}
\|e\|_F &\leq \|e_r\|_F + \|\hat{e}_1\|_F \\
&\leq \tilde{\gamma}_{(r+1)n} \left\|\begin{pmatrix} R_1 \\ G_r \end{pmatrix}\right\|_F + (\tilde{\gamma}_{m_2n_2} + \tilde{\gamma}_{m_2c})\max\big(\|E_2\|_F, \|U_c\|_F\big) \\
&\leq (\tilde{\gamma}_{(r+1)n} + \tilde{\gamma}_{m_2n_2} + \tilde{\gamma}_{m_2c})\max\left(\|E_2\|_F, \|U_c\|_F, \left\|\begin{pmatrix} R_1 \\ G_r \end{pmatrix}\right\|_F\right).
\end{aligned} \tag{33}$$

Now, if the orthogonal factor Q is to be formed explicitly, then the deviation from normality in our updating procedure can be examined. For this, we consider $E = I$, then from Lemma 4.8, we have

$$\tilde{Q}_2 = Q_2^T(I_{m_2} + \zeta),$$

where ζ is the error in the computed factor $\tilde{Q}_2$ given as

$$\|\zeta\|_F \le \tilde{\gamma}_{m_2 n_2}\|I_{m_2}\|_F = \sqrt{n_2}\tilde{\gamma}_{m_2 n_2},$$

where $\|I_{m_2}\|_F = \sqrt{n_2}$. In a similar manner, the computed factor $\tilde{Q}_1$ after appending columns U_c is given by

$$\tilde{Q}_1 = Q_1^T(I_{m_2} + \zeta_c),$$

where

$$\|\zeta_c\|_F \le \sqrt{c}\tilde{\gamma}_{m_2 c}.$$

Therefore, the total error in $\tilde{Q}$ is given as

$$\tilde{Q}_2\tilde{Q}_1 = Q_2^T(I_{m_2} + \zeta_1),$$

where

$$\|\zeta_1\|_F \le \|\zeta_2\|_F + \|\zeta_c\|_F \le \sqrt{n_2}\tilde{\gamma}_{m_2 n_2} + \sqrt{c}\tilde{\gamma}_{m_2 c}.$$

Similarly, the error in the computed factor $\tilde{Q}$ after appending block of rows is given as

$$\tilde{Q} = Q^T(I_r + \zeta_r),$$

where

$$\|\zeta_r\|_F \le \sqrt{n}\tilde{\gamma}_{(r+1)n}.$$

So, the total error ζ in Q during the whole updating procedure is given by

$$\begin{aligned}\|\zeta\|_F &= \|\zeta_1\|_F + \|\zeta_r\|_F \\ &\le \sqrt{n}\tilde{\gamma}_{(r+1)n} + \sqrt{n_2}\tilde{\gamma}_{m_2 n_2} + \sqrt{c}\tilde{\gamma}_{m_2 c},\end{aligned} \tag{34}$$

where $m_2 < m + p$, $n_2 < n$ and $c < n$.

Therefore, the error measure in $\tilde{Q}$ which shows the amount by which it has been deviated from normality is given by

$$\left\|I - \tilde{Q}^T\tilde{Q}\right\|_F \le \sqrt{n}(\tilde{\gamma}_{(r+1)n} + \tilde{\gamma}_{m_2 n_2} + \tilde{\gamma}_{m_2 c}). \tag{35}$$

Table 1 Description of test problems

Problem	Size of (A)	$\kappa(A)$	$\|A\|_F$	Size of (B)	$\kappa(B)$	$\|B\|_F$
1.	10×8	1.3667e+02	2.0006e+02	6×8	7.4200e+01	1.0216e+02
2.	100×90	2.9303e+03	2.1395e+03	90×90	3.3687e+03	1.3735e+03
3.	800×700	6.2106e+03	1.6872e+04	600×700	1.6164e+03	9.9000e+03
4.	1,000×500	1.1602e+03	1.5943e+04	500×500	1.2883e+05	7.6360e+03
5.	2,000×1,000	1.6727e+03	3.1884e+04	1,000×1,000	1.7430e+06	1.5272e+04

From equation (20), we have

$$\tilde{\gamma}_{(r+1)n} + \tilde{\gamma}_{m_2 n_2} + \tilde{\gamma}_{m_2 c} \approx \tilde{\gamma}_{(m+p+1)n} = \tilde{\gamma}_{(m+p)n}, \tag{36}$$

and using it in equation (35), we get the required result (31).

Also, we have

$$\begin{aligned} \|E - \tilde{Q}\tilde{R}\|_F &= \big\|(E - Q\tilde{R}) + \big((Q - \tilde{Q})\tilde{R}\big)\big\|_F \\ &\leq \sqrt{n}(\tilde{\gamma}_{(r+1)n} + \tilde{\gamma}_{m_2 n_2} + \tilde{\gamma}_{m_2 c}) \max\left(\|E_2\|_F, \|E_c\|_F, \left\|\begin{pmatrix} R_1 \\ G_r \end{pmatrix}\right\|_F\right). \end{aligned} \tag{37}$$

As $\|X\|_F = \|QR\|_F = \|R\|_F$, therefore, we can write

$$\max\left(\|E_2\|_F, \|U_c\|_F, \left\|\begin{pmatrix} R_1 \\ G_r \end{pmatrix}\right\|_F\right) = \left\|\begin{pmatrix} R_1 \\ G_r \end{pmatrix}\right\|_F = \|E\|_F. \tag{38}$$

Hence, applying expressions (36) and (38) to (37), we obtain the required equation (32) which shows how large is the error in the computed QR factorization. □

5 Numerical experiments

This section is devoted to some numerical experiments which illustrate the applicability and accuracy of Algorithm 5. The problem matrices A and B and its corresponding right-hand side vectors b and d are generated randomly using the MATLAB built-in commands rand('twister') and rand. These commands generate pseudorandom numbers from a standard uniform distribution in the open interval $(0,1)$. The full description of test matrices are given in Table 1, where we denote the Frobenius norm with $\|\cdot\|_F$ and the condition number by $\kappa(\cdot)$. For accuracy of the solution, we consider the actual solution such that $x = \text{rand}(n,1)$ and denote the result obtained from Algorithm 5 by x_{LSE} and that of direct QR Householder factorization with column pivoting by x_p. We obtain the relative errors between the solutions given in Table 2. Moreover, the solution x_{LSE} satisfy the constrained system effectively. The description of the matrix E, the size of the reduced subproblem (SP), value of the weighted factor ω, the relative errors err $= \|x - x_{\text{LSE}}\|_2/\|x\|_2$ and err1 $= \|x - x_p\|_2/\|x\|_2$ are provided in Table 2. We also carry out the backward error tests of Algorithm 5 numerically for our considered problems and provide the results in Table 3, which agrees with our theoretical results.

6 Conclusion

The solution of linear least squares problems with equality constraints is studied by updated techniques based on QR factorization. We updated only the R factor of the QR

Table 2 Results comparison

Problem	Size of (E)	γ	Size of SP	err1	err
1.	16×8	8.9564e+15	3×3	1.3222e–14	1.4585e–15
2.	190×90	7.1258e+15	3×3	1.2628e–13	5.5294e–14
3.	1,400×700	7.7993e+15	3×3	1.2821e–12	4.2522e–13
4.	1,500×500	9.5551e+15	3×3	1.9377e–12	1.3559e–12
5.	3,000×1,000	9.5549e+15	3×3	1.0828e–10	8.5181e–12

Table 3 Backward error analysis results

Problem	$\frac{\|E-\tilde{Q}\tilde{R}\|_F}{\|E\|_F}$	$\|I-\tilde{Q}^T\tilde{Q}\|_F$
1.	4.4202e–16	1.3174e–15
2.	4.7858e–16	9.0854e–15
3.	1.0450e–15	4.9428e–14
4.	9.0230e–16	3.8711e–14
5.	9.9304e–16	6.4026e–14

factorization of the small subproblem in order to obtain the solution of our considered problem. Numerical experiments are provided which illustrated the accuracy of the presented algorithm. We also showed that the algorithm is backward stable. The presented approach is suitable for dense problems and also applicable where *QR* factorization of a problem matrix is available and we are interested in the solution after adding new data to the original problem. In the future, it will be of interest to study the updating techniques for sparse data problems and for those where the linear least squares problem is fixed and the constraint system is changing frequently.

Competing interests
The authors declare that they have no competing interests.

Authors' contributions
All authors contributed equally to this work. All authors read and approved the final manuscript.

Publisher's Note

Received: 5 July 2017 Accepted: 16 October 2017 Published online: 13 November 2017

References
1. Auton, JR, Van Blaricum, ML: Investigation of procedures for automatic resonance extraction from noisy transient electromagnetics data. Final Report, Contract N00014-80-C-029, Office of Naval Research Arlington, VA 22217 (1981)
2. Barlow, JL, Handy, SL: The direct solution of weighted and equality constrained least-squares problems. SIAM J. Sci. Stat. Comput. **9**, 704-716 (1988)
3. Lawson, CL, Hanson, RJ: Solving Least Squares Problems. SIAM, Philadelphia (1995)
4. Eldén, L: Perturbation theory for the least squares problem with linear equality constraints. SIAM J. Numer. Anal. **17**, 338-350 (1980)
5. Leringe, O, Wedin, PÅ: A comparison between different methods to compute a vector x which minimizes $\|Ax-b\|_2$ when $Gx=h$. Tech. Report, Department of computer science, Lund University (1970)
6. Van Loan, CF: A generalized SVD analysis of some weighting methods for equality constrained least squares. In: Kågström, B, Ruhe, A (eds.) Matrix Pencils. Lecture Notes in Mathematics, vol. 973, pp. 245-262. Springer, Heidelberg (1983)
7. Van Loan, CF: On the method of weighting for equality-constrained least-squares problems. SIAM J. Numer. Anal. **22**, 851-864 (1985)
8. Wei, M: Algebraic properties of the rank-deficient equality-constrained and weighted least squares problems. Linear Algebra Appl. **161**, 27-43 (1992)
9. Stewart, GW: On the weighting method for least squares problems with linear equality constraints. BIT Numer. Math. **37**, 961-967 (1997)
10. Anderson, E, Bai, Z, Dongarra, J: Generalized *QR* factorization and its applications. Linear Algebra Appl. **162**, 243-271 (1992)
11. Björck, Å: Numerical Methods for Least Squares Problems. SIAM, Philadelphia (1996)

12. Golub, GH, Van Loan, CF: Matrix Computations. Johns Hopkins University Press, Baltimore (1996)
13. Stewart, GW: Matrix Algorithms. SIAM, Philadelphia (1998)
14. Alexander, ST, Pan, CT, Plemmons, RJ: Analysis of a recursive least squares hyperbolic rotation algorithm for signal processing. Linear Algebra Appl. **98**, 3-40 (1988)
15. Chambers, JM: Regression updating. J. Am. Stat. Assoc. **66**, 744-748 (1971)
16. Haley, SB, Current, KW: Response change in linearized circuits and systems: computational algorithms and applications. Proc. IEEE **73**, 5-24 (1985)
17. Haley, SB: Solution of modified matrix equations. SIAM J. Numer. Anal. **24**, 946-951 (1987)
18. Lai, SH, Vemuri, BC: Sherman-Morrison-Woodbury-formula-based algorithms for the surface smoothing problem. Linear Algebra Appl. **265**, 203-229 (1997)
19. Björck, Å, Eldén, L, Park, H: Accurate downdating of least squares solutions. SIAM J. Matrix Anal. Appl. **15**, 549-568 (1994)
20. Daniel, JW, Gragg, WB, Kaufman, L, Stewart, GW: Reorthogonalization and stable algorithms for updating the Gram-Schmidt *QR* factorization. Math. Comput. **30**, 772-795 (1976)
21. Gill, PE, Golub, GH, Murray, W, Saunders, M: Methods for modifying matrix factorizations. Math. Comput. **28**, 505-535 (1974)
22. Reichel, L, Gragg, WB: Algorithm 686: FORTRAN subroutines for updating the *QR* decomposition. ACM Trans. Math. Softw. **16**, 369-377 (1990)
23. Hammarling, S, Lucas, C: Updating the *QR* factorization and the least squares problem. Tech. Report, The University of Manchester (2008). http://www.manchester.ac.uk/mims/eprints
24. Yousaf, M: Repeated updating as a solution tool for linear least squares problems. Dissertation, University of Essex (2010)
25. Andrew, R, Dingle, N: Implementing *QR* factorization updating algorithms on GPUs. Parallel Comput. **40**, 161-172 (2014)
26. Zhdanov, AI, Gogoleva, SY: Solving least squares problems with equality constraints based on augmented regularized normal equations. Appl. Math. E-Notes **15**, 218-224 (2015)
27. Zeb, S, Yousaf, M: Repeated *QR* updating algorithm for solution of equality constrained linear least squares problems. Punjab Univ. J. Math. **49**, 51-61 (2017)
28. Higham, NJ: Accuracy and Stability of Numerical Algorithms. SIAM, Philadelphia (2002)
29. Wilkinson, JH: Error analysis of transformations based on the use of matrices of the form $I - 2ww^H$. In: Rall, LB (ed.) Error in Digital Computation, vol. 2, pp. 77-101. Wiley, New York (1965)
30. Parlett, BN: Analysis of algorithms for reflections in bisectors. SIAM Rev. **13**, 197-208 (1971)
31. Cox, AJ: Stability of algorithms for solving weighted and constrained least squares problems. Dissertation, University of Manchester (1997)