A Generalized SVD Analysis of Some Weighting Methods for Equality Constrained Least Squares

Charles Van Loan
Department of Computer Science
Cornell University
Ithaca, New York, 14853, USA

Abstract

The method of weighting is a useful way to solve least squares problems that have linear equality constraints. New error bounds for the method are derived using the generalized singular value decomposition. The analysis clarifies when the weighting approach is successful and suggests modifications when it is not.

1. Introduction

The problem we consider is how to find a vector $x \in \mathbb{R}^n$ that solves the equality constrained problem

\[
\text{min} \quad ||Ax - b||_2 \\
A x = d
\]

where $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$, $B \in \mathbb{R}^{p \times n}$ and $d \in \mathbb{R}^p$. We will assume that $\text{rank}(B) = p$ and that the nullspaces of the two matrices satisfy $N(A) \cap N(B) = \{0\}$. These conditions ensure that (LSB) has a unique solution which we designate by $x_{LSB}$.

Important settings where this problem arises include constrained surface fitting, penalty function methods in nonlinear optimization, and geodetic least squares adjustment.

Several methods for solving the LS problem are discussed in Lawson...
and Hanson [7, Chapters 20–22]. In one approach QR factorizations are used to compute the projections of $x_{LS}$ onto $N(B)^\perp$ and $N(B)$:

(a) $B^T = \begin{bmatrix} Q_2 & Q_2 \end{bmatrix} \begin{bmatrix} R_B & 0 \\ 0 & R_B \end{bmatrix}$ (Q-R)

(b) $R_B y_1 = d : x_1 := Q_2 y_1$

(1.1)

(c) $A = [U_1, U_2] \begin{bmatrix} 0 \\ 0 \end{bmatrix} \begin{bmatrix} U_1 \end{bmatrix} n-p$ (Q-R)

(d) $R_2 y_2 = U_1^T (b - Ax_1) : y_{LS} = x_1 + Q_2 y_2$

This algorithm is easy to implement using the LINPACK routines. (It is a MATLAB "5-liner".)

Unfortunately, (1.1) is not a viable method for solving the large sparse LS problem because the matrix $A_2$ will generally be dense. In this context the method of weighting is of interest. The idea behind this approach is simply to compute the solution $x(u)$ to the unconstrained problem

(1.2) $\min_{x \in B^n} \left\| \begin{bmatrix} A \end{bmatrix} x - \begin{bmatrix} d \end{bmatrix} \right\|_2$

for a large value of $u \in \mathbb{R}$. It is widely known that $x(b) = x_{LS}$ as $u \to \infty$. Thus, existing software for sparse LS problems can “in principle” be used to generate an approximation to $x_{LS}$ of arbitrary quality. However, several issues associated with the method of weighting demand our attention. At what rate do the quantities $x(b)$ and $d - Ax(b)$ converge? Are there practical ways to estimate the accuracy of $x(b)$? How can we cope with the numerical problems that can be expected to arise when $u$ is extremely large? We are prompted to discuss these issues because of the increasing importance of the sparse LS problem. But our analytic and algorithmic developments will also be of interest to solvers of small LS problems since the simplicity of the weighting method makes it extremely attractive and popular.
Our discussion is structured as follows. First, we analyze the properties of $X$ and $Z$ using the generalized singular value decomposition. The limitations of the theory are then made obvious by reviewing the numerical difficulties associated with large $\mu$. Next, we propose two techniques that can be used both to improve $x(u)$ and to estimate its error. One technique involves extrapolation and the other iterative improvement. We conclude with some remarks about the practical implementation of our ideas.

2. A GSVD Analysis of $x(u)$

The generalized singular value decomposition (GSVD) of $A$ and $B$ is useful as a tool for analyzing the method of weighting. This decomposition is as follows:

**Theorem 2.1**

If $A \in \mathbb{R}^{m \times n}$ and $B \in \mathbb{R}^{p \times n}$ satisfy $N(A) \cap N(B) = \{0\}$ then there exist

$$U = [u_1, \ldots, u_n] \in \mathbb{R}^{m \times n} \quad \text{(orthogonal)}$$

$$V = [v_1, \ldots, v_p] \in \mathbb{R}^{n \times p} \quad \text{(orthogonal)}$$

$$X = [x_1, \ldots, x_n] \in \mathbb{R}^{n \times n} \quad \text{(nonsingular)}$$

such that

$$U^T A V = D_A = \text{diag}(\alpha_1, \ldots, \alpha_p)$$

and

$$V^T B X = D_B = \text{diag}(\beta_1, \ldots, \beta_p).$$

Without loss of generality we may assume

$$||X||_2 = 1 \quad \text{and} \quad ||X^T||_2 = \frac{\alpha_1}{\alpha_n}$$

where $\alpha_1$ and $\alpha_n$ are the largest and smallest singular values of $A$.
Proof

Let

\[
\begin{bmatrix}
A \\
B
\end{bmatrix} = \begin{bmatrix}
C_1 \\
C_2
\end{bmatrix} \text{diag}(c_1, \ldots, c_n) S^T
\]

be the SVD of \([A, B]\) with \(C_1^TC_1 + C_2^TC_2 = I_n\), \(c_1, \ldots, c_n \geq 0\), and \(S^TS = I_n\).

Let

\[
\begin{bmatrix}
C_1 \\
C_2
\end{bmatrix} = \begin{bmatrix}
U & 0 \\
0 & V
\end{bmatrix} \begin{bmatrix} T
C_1 \\
W
\end{bmatrix}
\]

be the C-S decomposition of \([C_1, C_2]\) where \(U \in \mathbb{R}^{n \times m}, V \in \mathbb{R}^{n \times p}, W \in \mathbb{R}^{p \times n}\) are orthogonal and

\[
C = \text{diag}(c_1, \ldots, c_n) \in \mathbb{R}^{m \times n} \quad c_j \geq 0
\]

and

\[
S = \text{diag}(s_1, \ldots, s_p) \in \mathbb{R}^{n \times p} \quad 0 \leq s_j \leq 1
\]
satisfy \(C^TC + S^TS = I_n\). This decomposition is discussed in Stewart [9] who also presents an effective algorithm for computing it in [10].

The theorem follows by setting \(A_1 = c_1 C, A_2 = S, B = c_2 S, X^{-1} = \begin{bmatrix} W^T \text{diag}(c_1) S^T \end{bmatrix} \). Note that \(c_n > 0\) because \(N(A) \cap N(B) = \{0\} \mathbb{N}\).

A number of elementary consequences of the GVD are repeatedly used in the sequel. These are summarized in the following result:

**Corollary 2.2**

Suppose the GVD is computed as indicated by the proof of Theorem 2.1. If \(\text{rank}(N) = p\) and

\[
v_i = \frac{v_i}{\delta_i} \quad i = 1, \ldots, p
\]

then
\( \alpha_1^2 + \beta_1^2 = \gamma_1^2 \quad i = 1, \ldots, p \)

(b) \( \alpha_2 \geq \ldots \geq \alpha_q \geq \alpha_{q+1} = \ldots = \alpha_p = 0 \) where \( q = \dim(\text{null } B(A) \mathbb{K}) \)

(c) \( q = 1, \ldots, q \leq n \)

(d) \( 0 \leq \beta_1 \leq \beta_2 \leq \ldots \leq \beta_p \)

(e) \( u_1, \ldots, u_p \in \mathbb{K} \)

(f) \( AX = A \alpha \beta u_1 \quad i = 1, \ldots, n \)

(g) \( RX = B \alpha \beta v_1 \quad i = 1, \ldots, p \).

**Proof**

Contentions (a) and (c) follow from \( D^T A = D^T B = S \) while (d) and (e) are true because \( s_1 \leq \ldots \leq s_p \) and \( p = \text{rank}(B) \). The equations \( AX = D \alpha \) and \( BX = D \beta \) establish (f) and (g). Finally, if \( \alpha_1 > \alpha_{q+1} = \ldots = \alpha_p = 0 \) then it follows from \( X^T X = X^T D^T A D X \) that the first \( q \) columns of \( X^T X \) span the subspace \( \text{null}(A^T) \cap \text{null}(A) = (\text{null}(A))^\perp \). This proves (b).

The \( u_i \) are called the generalized singular values of \((A, B)\). We mention that a more general version of the GSVD is given in [11].

The GSVD can be used to diagonalize the LSE problem. In particular, by setting

\[
\tilde{b} = u^T b = [u_1^T b, \ldots, u_p^T b]^T
\]
\[
\tilde{d} = v^T d = [v_1^T d, \ldots, v_p^T d]^T
\]
\[
y = x^T \gamma = [y_1, \ldots, y_p]^T
\]

we obtain

\[
(\text{LSE}') \quad \min \| D X \gamma - \tilde{b} \|_2
\]

\[
\gamma \in \mathbb{K}^p
\]

It is not hard to show that
solves this problem. Since \( x_{p+1} = \ldots = x_n = 0 \), we have

\[
X_{\text{LS}} = X Y_{\text{LS}} = \begin{pmatrix}
\frac{T}{d} \begin{pmatrix}
1 & \ldots & 1
\end{pmatrix} \varepsilon_1 & \ldots & \frac{T}{d} \begin{pmatrix}
1 & \ldots & 1
\end{pmatrix} \varepsilon_p & \frac{T}{d} \begin{pmatrix}
1 & \ldots & 1
\end{pmatrix} \varepsilon_{p+1} & \ldots & \frac{T}{d} \begin{pmatrix}
1 & \ldots & 1
\end{pmatrix} \varepsilon_n
\end{pmatrix}
\]

If we define

\[
c_i = \begin{pmatrix}
1 & \ldots & 1
\end{pmatrix} \varepsilon_i d_i = \\
1 & \ldots & 1
\]

\[
c_{i+1} = \begin{pmatrix}
1 & \ldots & 1
\end{pmatrix} \varepsilon_{i+1} d_{i+1} = \\
1 & \ldots & 1
\]

and

\[
x_{i+1} = \begin{pmatrix}
1 & \ldots & 1
\end{pmatrix} \varepsilon_{i+1} d_{i+1} = \\
1 & \ldots & 1
\]

then it is easy to show that the constrained minimum residual is given by

\[
X_{\text{LS}} = X Y_{\text{LS}} = X_{\text{LS}} + \begin{pmatrix}
1 & \ldots & 1
\end{pmatrix} \cdot \begin{pmatrix}
\cdots
\end{pmatrix}
\]

Note that \( X_{\text{LS}} = b - A \hat{x}_{\text{LS}} \) where

\[
X_{\text{LS}} = \begin{pmatrix}
\begin{pmatrix}
1 & \ldots & 1
\end{pmatrix} \varepsilon_d & \begin{pmatrix}
1 & \ldots & 1
\end{pmatrix} \varepsilon_{i+1}
\end{pmatrix}
\]

is the solution of minimum 2-norm to the unconstrained 1D problem

\[
\min \| b - A \hat{x}_{\text{LS}} \|_2.
\]

Observe that

\[
x_{i+1} = \begin{pmatrix}
1 & \ldots & 1
\end{pmatrix} \varepsilon_{i+1} d_{i+1} = \\
1 & \ldots & 1
\]

measures how much the residual increases as a result of the constraints.

It is clear from (2.2) that \( X_{\text{LS}} \) is sensitive to perturbation whenever \( c_k = 0 \) or \( \lambda = \sqrt{\frac{1}{\lambda_k}} \) is small. Because it "displays" these critical quantities, it is advisable to solve ill-conditioned LSE problems via the GSVD. The sensitivity of the LSE problem is investigated in [5] and [12].

The GSVD is also convenient for analyzing \( x | y \). Since this vector is usual (1.2) it must satisfy the normal equations

\[
(A^T A + \varepsilon^T \varepsilon B) x(y) = A^T b + \varepsilon^T \varepsilon d.
\]
Under (2.1) this transforms to
\[ (D^2 + y^2 D)\bar{y}(\mu) = D^2 \mu + y^2 D \mu \]
where \( \bar{y}(\mu) = \chi y(\mu) \). It is easy to deduce the solution to this diagonal system and that
\[ x(\mu) = \sum_{i=1}^{n} \frac{\mu_i x_i}{\mu_i^2 + y^2} \frac{y_i^2}{\mu_i^2 + y^2} x_i + \sum_{i=1}^{n} \frac{y_i}{\mu_i^2 + y^2} x_i. \]
By subtracting (2.2) from this equation we find
\[ x(\mu) = x(b) - x_{\text{LSE}} = \sum_{i=1}^{n} \frac{\mu_i}{\mu_i^2 + y^2} \left( \frac{\mu_i}{\mu_i^2 + y^2} \right) x_i. \]
Note that the error is confined to \( \text{spec} \{ x_1, \ldots, x_n \} \) and that it tends to zero as \( n \to \infty \). An alternative proof of the latter fact may be found in [7, Chapter 22].

In section 4 we establish the following inequalities:
\[ \| x(\mu) - x_{\text{LSE}} \|_2 \leq \frac{\Delta}{2\mu_1} \sqrt{1 + \mu_1^2} \]
\[ \| x - x(\bar{\mu}) \|_2 \leq \frac{\Delta}{2\mu} \]
\[ 0 \leq \| x_{\text{LSE}} \|_2 - \| x - x(\bar{\mu}) \|_2 \leq \frac{\Delta\mu_1}{2\mu} \]
These results suggest that a large weight might be required if either \( \mu_1 \) is large or \( n \) is small.

3. Difficulties Associated With a Large Weight

It is widely appreciated that care must be exercised when Householder matrices are used to compute the QR factorization of a matrix whose rows vary greatly in norm, e.g., the matrix in (1.3). Powell and Reid [8] examined this problem in conjunction with the Businger-Sorenson algorithms [4] and advise incorporation of row interchanges, such as in Gaussian elimination. Specifically, they recommend that the k-th
column be searched and its largest entry pivoted to the \((k,k)\) position before the \(k\)-th Householder matrix is applied.

Note that near-domination of the pivot elements will result if the Businger-Golub algorithm is applied to our heavily weighted matrix in (1.2) but not if we apply it to the mathematically equivalent problem

\[
\min \left\| \begin{pmatrix} a \\ b \end{pmatrix} x - \begin{pmatrix} c \\ d \end{pmatrix} \right\|_2 \quad \nu \gg 1.
\]

To approximate the difference between the "B-over-A" and the "A-over-B" formulations consider the LSE problem

\[
A = \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}, \quad b = \begin{pmatrix} 1 \\ 1 \end{pmatrix}, \quad B = (1,-1), \quad d = (2).\]

This problem is well-conditioned and has exact solution \(x_{\text{LSR}} = \frac{1}{29}(19, -19)^T\). In the following table we record the error \(\| x(\nu) - x_{\text{LSR}} \|_2\) for both approaches as a function of \(\nu\):

<table>
<thead>
<tr>
<th>method</th>
<th>(10^1)</th>
<th>(10^2)</th>
<th>(10^3)</th>
<th>(10^4)</th>
<th>(10^5)</th>
<th>(10^6)</th>
<th>(10^7)</th>
<th>(10^8)</th>
<th>(10^9)</th>
</tr>
</thead>
<tbody>
<tr>
<td>B-over-A</td>
<td>(10^{-2})</td>
<td>(10^{-1})</td>
<td>(10^{-2})</td>
<td>(10^{-3})</td>
<td>(10^{-4})</td>
<td>(10^{-5})</td>
<td>(10^{-6})</td>
<td>(10^{-7})</td>
<td>(10^{-8})</td>
</tr>
<tr>
<td>A-over-B</td>
<td>(10^{-3})</td>
<td>(10^{-2})</td>
<td>(10^{-3})</td>
<td>(10^{-4})</td>
<td>(10^{-5})</td>
<td>(10^{-6})</td>
<td>(10^{-7})</td>
<td>(10^{-8})</td>
<td>(10^{-9})</td>
</tr>
</tbody>
</table>

These computations were performed using VAX double precision arithmetic (macheps \(= 10^{-17}\)) in the MATLAB environment.

The divergent of the two methods in the vicinity of \(\nu = \text{macheps}^{1/2}\) is fairly typical, even for ill-conditioned examples. However, for ill-conditioned LSE problems, the "optimum" weight for the B-over-A approach is usually several orders of magnitude greater than (macheps)\(^{1/2}\). Thus it is preferable (although not always critical) to solve (1.2) instead of (3.1). In the remainder of this paper, we will always use the B-over-A formulation (1.2).

However, it is not always easy to arrange the rows of the unconstrained problem so that the constraint equations come first. The minimization of fill-in may dictate some other row ordering in the case when \(A\) and \(B\) are large and sparse.
Column ordering is also important in the method of weighting. At the
suggestion of the referee, we considered the example

\[
A = \begin{bmatrix}
1 & 1 & 1 \\
1 & 3 & 1 \\
1 & -1 & 1
\end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix}
1 \\
2 \\
4
\end{bmatrix}
\]

\[
B = \begin{bmatrix}
1 & 1 & 1 \\
1 & 1 & -1
\end{bmatrix}, \quad \mathbf{d} = \begin{bmatrix}
7 \\
4
\end{bmatrix}
\]

This example is well conditioned and has exact solution \( \mathbf{x}_{\text{LR}} = \begin{bmatrix} 146 \\ -2 \\ 13 \end{bmatrix} \). Let \( \mathbf{R}(\mathbf{u}) = (\mathbf{R}_{11}(\mathbf{u})) \) be the 3-by-3 upper triangular matrix

\( \begin{bmatrix} 3 & 1 & 1 \\
1 & 1 & -1
\end{bmatrix} \)

obtained by computing the Q-R decomposition of \( \mathbf{A} \). It is clear that

\( \mathbf{R}_{11}(\mathbf{u})/\mathbf{R}_{22}(\mathbf{u}) \) becomes very large as \( \mathbf{u} \to 0 \). This occurs because the

first two columns of \( \mathbf{B} \) are dependent. On the other hand, if we apply

Q-R with column pivoting to \( \mathbf{B} \) then \( \mathbf{R}_{11}(\mathbf{u}) \) and \( \mathbf{R}_{22}(\mathbf{u}) \) have the same order of magnitude as \( \mathbf{u} \to 0 \). The decision to column pivot is impor-

tant as the following table of relative errors in \( \mathbf{x}(\mathbf{u}) \) indicates:

<table>
<thead>
<tr>
<th>Column Pivoting</th>
<th>( 10^1 )</th>
<th>( 10^2 )</th>
<th>( 10^3 )</th>
<th>( 10^4 )</th>
<th>( 10^5 )</th>
<th>( 10^6 )</th>
<th>( 10^7 )</th>
<th>( 10^8 )</th>
<th>( 10^9 )</th>
<th>( 10^{10} )</th>
<th>( 10^{11} )</th>
<th>( 10^{12} )</th>
<th>( 10^{13} )</th>
<th>( 10^{14} )</th>
<th>( 10^{15} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>No</td>
<td>( 10^{-4} )</td>
<td>( 10^{-7} )</td>
<td>( 10^{-11} )</td>
<td>( 10^{-14} )</td>
<td>( 10^{-16} )</td>
<td>( 10^{-16} )</td>
<td>( 10^{-16} )</td>
<td>( 10^{-16} )</td>
<td>( 10^{-16} )</td>
<td>( 10^{-16} )</td>
<td>( 10^{-16} )</td>
<td>( 10^{-16} )</td>
<td>( 10^{-16} )</td>
<td>( 10^{-16} )</td>
<td>( 10^{-16} )</td>
</tr>
<tr>
<td>Yes</td>
<td>( 10^{-2} )</td>
<td>( 10^{-2} )</td>
<td>( 10^{-7} )</td>
<td>( 10^{-7} )</td>
<td>( 10^{-7} )</td>
<td>( 10^{-7} )</td>
<td>( 10^{-7} )</td>
<td>( 10^{-7} )</td>
<td>( 10^{-7} )</td>
<td>( 10^{-7} )</td>
<td>( 10^{-7} )</td>
<td>( 10^{-7} )</td>
<td>( 10^{-7} )</td>
<td>( 10^{-7} )</td>
<td>( 10^{-7} )</td>
</tr>
</tbody>
</table>

We infer that the weighting method can be unstable if (1.2) is solved

using the Q-R decomposition without pivoting.

4. An Extrapolation Procedure

As we have seen, the importance of row and column ordering in (1.2)

increases with increasing \( \mu \). Hence, it is potentially interesting to see how the method might be used with "safe" weights since the
ordering of rows and columns is often inconvenient.

Suppose $x(\mu)$ and $x(\nu\mu)$ have been computed for some $\mu > 0$ and $\nu > 1$. Using (2.10) it can be shown that if

$$
(4.1) \quad x(\nu\mu) = x(\nu) + \frac{1}{\gamma^2 - 1} \left[ x(\nu) - x(\nu^2) \right]
$$

then

$$
x(\nu\mu) = x_{LSE} + \frac{\mu_1^2}{\gamma_1^2 \gamma_2^2} \cdot \frac{\mu_1}{\mu_2 (\nu \mu)} \cdot \frac{\beta_1}{\beta_2} \cdot x_1
$$

Thus, the error expansion for $x(\nu\mu)$ is the same as that for $x(\nu)$ except the coefficient of $x_1$ is diminished by the factor $\mu_1^2 / (\gamma_1^2 \gamma_2^2)$.

The calculation (4.1) is merely the first extrapolation in the following Richardson scheme:

For $j=0, 1, \ldots$

$$
x^{(j)}(\nu) = x(\nu^{j-\nu})
$$

For $k=1, 2, \ldots$

$$
x^{(k)}(\nu) = x^{(k)}(\nu^{j-\nu}) + \frac{1}{\gamma^2 - 1} \left[ x^{(k-1)}(\nu^{j-\nu}) - x^{(k-1)}(\nu^{j-\nu - 1}) \right]
$$

A messy but straightforward induction argument based on (2.10) can be used to show that

$$
x^{(k)}(\nu) = x_{LSE} + \frac{\mu_1}{\gamma_1^2 \gamma_2^2} \cdot \frac{\mu_1}{\mu_2 (\nu)^2} \cdot \frac{\beta_1}{\beta_2} \cdot x_1
$$

where

$$
(4.4) \quad \left( k \right)_j = \frac{\mu_1}{\beta_1} \cdot \frac{\mu_1}{\mu_2 (\nu)^2} \cdot \frac{1}{\gamma^2 - 1} \prod_{\ell=0}^{j-1} \left( \frac{\mu_1}{\mu_2} + \frac{\beta_1}{\beta_2} \gamma^2 \right)
$$
Other quantities of interest associated with \( x^{(k)}(y^j u) \) include the B-residual

\[
(4.5) \quad s^{(k)}(y^j u) = d - B x^{(k)}(y^j u) = - \frac{q}{v^2} \delta \varphi^{(k)}(j) v^1
\]

and the A-residual

\[
(4.6) \quad r^{(k)}(y^j u) = b - Ax^{(k)}(y^j u) = r_{LSE} - \frac{q}{v^1} \varphi^{(k)}(j) u^1
\]

Using these formulae we have

**Theorem 4.1**

If \( 0 = \frac{u^2}{v^2} + \frac{u^2}{v^2} + \frac{\gamma^2(k)}{2v^2} \) and \( x^{(k)}(y^j u) \) is generated via (4.2), then

(a) \( \| x^{(k)}(y^j u) - x_{LSE} \|_2 \leq \frac{\Delta \sqrt{1 + \frac{u^2}{v^2}}}{2v^2} \gamma k \)

(b) \( \| d - B x^{(k)}(y^j u) \|_2 \leq \frac{\Delta}{2v^1} \gamma k \)

(c) \( 0 \leq \| x_{LSE} \|_2 - \| b - Ax^{(k)}(y^j u) \|_2 \leq \frac{u^1 \Delta}{2v^1 \gamma^2} \gamma k \).

**Proof**

Using (4.4) and the inequality

\[
\frac{u^1}{\gamma^2} \leq \frac{1}{2v^1 \gamma^2}
\]

it follows that \( x^{(k)}(y^j u) \leq \frac{2}{v^1} \| \delta \varphi^{(k)}(j) \|_2 \gamma k \). Since \( \| X \|_2 = 1 \)

we have by taking norms in (4.3) and using (2.7) that

\[
\| x^{(k)}(y^j u) \|_2 \leq \frac{\Delta}{2v^1} \gamma k
\]

Inequality (a) follows since \( S_1^{-2} = (1 + \frac{2}{v^2}) \frac{1}{v^2} \). To prove (b) and the
Upper bound in (c), take norms in (4.5) and (4.6) respectively and use the orthonormality of \( u_1 \) and \( v_1 \).

To establish the lower bound in (c), note from (4.6) and (2.6) that
\[
\tau^{(k)} (\gamma^2 \omega) = \tau_{\text{LS},k} - \sum_{i=1}^q (\xi_i - \xi^{(k)}_i(j)) v_i.
\]
Since \( \tau_{\text{LS},k} \in \text{span} \{ u_1, \ldots, u_q \} \), it follows that
\[
|\tau^{(k)} (\gamma^2 \omega)|_2^2 = |\tau_{\text{LS},k}|_2^2 - \sum_{i=1}^q |(\xi_i - \xi^{(k)}_i(j))|^2.
\]
The desired bound follows from (2.7) and the easily verified inequality
\[
\sigma^2 \leq (\xi_i - \xi^{(k)}_i(j))^2, \quad \text{for all } i = 1, \ldots, q.
\]
Note that \(|\tau^{(k)} (\gamma^2 \omega)|_2^2\) increases monotonically to \(|\tau_{\text{LS},k}|_2^2\) as \( \omega \) increases.

The main observation to make from the theorem is that the error, \( A \)-residual, and \( B \)-residual improve by a factor of 8 with each extrapolation. For example, in a small problem with \( \nu_4 = 10^3 \), we set \( \nu = 100 \), \( \nu = 2 \) and found the following relative errors in \( x^{(k)} (\gamma^2 \omega) \):

<table>
<thead>
<tr>
<th>( j \times \nu )</th>
<th>( k=0 )</th>
<th>( k=1 )</th>
<th>( k=2 )</th>
<th>( k=3 )</th>
<th>( k=4 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>10^0</td>
<td>( 10^{-1} )</td>
<td>( 10^{-2} )</td>
<td>( 10^{-3} )</td>
<td>( 10^{-4} )</td>
</tr>
<tr>
<td>200</td>
<td>10^0</td>
<td>( 10^{-1} )</td>
<td>( 10^{-2} )</td>
<td>( 10^{-3} )</td>
<td>( 10^{-4} )</td>
</tr>
<tr>
<td>400</td>
<td>10^0</td>
<td>( 10^{-2} )</td>
<td>( 10^{-3} )</td>
<td>( 10^{-4} )</td>
<td>( 10^{-5} )</td>
</tr>
<tr>
<td>800</td>
<td>10^0</td>
<td>( 10^{-2} )</td>
<td>( 10^{-3} )</td>
<td>( 10^{-4} )</td>
<td>( 10^{-5} )</td>
</tr>
<tr>
<td>1600</td>
<td>10^0</td>
<td>( 10^{-3} )</td>
<td>( 10^{-4} )</td>
<td>( 10^{-5} )</td>
<td>( 10^{-6} )</td>
</tr>
<tr>
<td>3200</td>
<td>10^0</td>
<td>( 10^{-3} )</td>
<td>( 10^{-4} )</td>
<td>( 10^{-5} )</td>
<td>( 10^{-6} )</td>
</tr>
</tbody>
</table>

In another example \( \nu_4 = 10^3 \) we found the relative errors diminishing as follows:

<table>
<thead>
<tr>
<th>( j \times \nu )</th>
<th>( k=0 )</th>
<th>( k=1 )</th>
<th>( k=2 )</th>
<th>( k=3 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>10^4</td>
<td>10^0</td>
<td>10^{-1}</td>
<td>10^{-2}</td>
<td>10^{-3}</td>
</tr>
<tr>
<td>10^6</td>
<td>10^{-1}</td>
<td>10^{-2}</td>
<td>10^{-3}</td>
<td>10^{-4}</td>
</tr>
<tr>
<td>10^8</td>
<td>10^{-5}</td>
<td>10^{-6}</td>
<td>10^{-7}</td>
<td>10^{-8}</td>
</tr>
<tr>
<td>10^{10}</td>
<td>10^{-9}</td>
<td>10^{-11}</td>
<td>10^{-13}</td>
<td>10^{-15}</td>
</tr>
</tbody>
</table>

The corresponding entries for the relative \( B \)-residual
\[
\left| \mathbf{x}^{(k)} (\gamma^2 \omega)|/|\mathbf{x}|_2 |/|\mathbf{u}_2|_2\right|_2
\]
were smaller by a factor of \( 10^4 \).
Based on numerous small examples run with MATLAB we conclude

(a) The extrapolation cannot "take hold" until \( j^3 \approx v^4 \).

(b) In ill-conditioned examples, there is no significant reduction in the error beyond the second or third extrapolation.

(c) There is no point in extrapolating if \( \gamma^j > \gamma^{j+1} \) when the unconstrained LS problem is in A-over-n form.

(d) Once the extrapolation "settle down", \( \|x^{(k)}(v^j \gamma^k) - x^{(k-1)}(v^j \gamma^k)\| \) gives good estimates of the error in \( x^{(k-1)}(v^{j+1}) \).

(e) Most time is spent in computing the Q-R decompositions for each \( x(v^j \gamma^k) \). The extrapolates are "free".

5. The Iterative Improvement of \( x(u) \)

A routine Lagrange multiplier argument can be used to show that \( x_{\text{LS}} \) and \( x_{\text{RSE}} \) satisfy

\[
\begin{pmatrix}
    b \\
    c \\
    0
\end{pmatrix}
= \begin{pmatrix}
    I & 0 & A \\
    0 & 0 & B \\
    A^2 & B^2 & 0
\end{pmatrix}
\begin{pmatrix}
    x_{\text{LS}} \\
    x_{\text{RSE}}
\end{pmatrix}
\]

Here, \( x_{\text{RSE}} \) is the corresponding Lagrange multiplier.

The unconstrained LS problem (1.2) can also be posed as a linear equation problem:

\[
\begin{pmatrix}
    b \\
    c \\
    0
\end{pmatrix}
= \begin{pmatrix}
    I & 0 & A \\
    0 & -A^{-2} & B \\
    A^2 & B^2 & 0
\end{pmatrix}
\begin{pmatrix}
    b - Ax(u) \\
    \gamma^2 (d - \gamma x(u)) \\
    x(u)
\end{pmatrix}
\]

(The last row in this equation is just the normal equation system (2.8).)

This suggests the following iteration for improving \( x(u) \):

\[
\begin{pmatrix}
    b \\
    c \\
    0
\end{pmatrix}
= \begin{pmatrix}
    I & 0 & A \\
    0 & -A^{-2} & B \\
    A^2 & B^2 & 0
\end{pmatrix}
\begin{pmatrix}
    b - Ax(u) \\
    \gamma^2 (d - \gamma x(u)) \\
    x(u)
\end{pmatrix}
\]
Based on numerous small examples run with MATLAB we conclude

(a) The extrapolation cannot "take hold" until $y_j = u_j$.

(b) In ill-conditioned examples, there is no significant reduction in the error beyond the second or third extrapolation.

(c) There is no point in extrapolating if $y_j > (\text{naehope})^{-1/2}$ when the unconstrained LS problem is in $A^{-1}$ form.

(d) Once the extrapolates "settle down", $||x^{(k)}(y_j) - x^{(k-1)}(y_j)||_2$ gives good estimates of the error in $x^{(k-1)}(y_j)$.

(e) Most time is spent in computing the Q-R decompositions for each $x(y_j)$. The extrapolates are "free".

5. The Iterative Improvement of $x(u)$

A routine Lagrange multiplier argument can be used to show that $x_{\text{LSE}}$ and $r_{\text{LSE}} = b - Ax_{\text{LSE}}$ satisfy

\[
\begin{bmatrix}
\hat{b} \\
\hat{d} \\
0
\end{bmatrix} =
\begin{bmatrix}
1 & 0 & A \\
0 & 0 & b \\
A^T & 0 & 0
\end{bmatrix}
\begin{bmatrix}
r_{\text{LSE}} \\
x_{\text{LSE}} \\
x_{\text{LSE}}
\end{bmatrix}
\]

Here, $\lambda_{\text{LSE}}$ is the corresponding Lagrange multiplier.

The unconstrained LS problem (1.7) can also be posed as a linear equation problem:

\[
\begin{bmatrix}
\hat{b} \\
\hat{d} \\
0
\end{bmatrix} =
\begin{bmatrix}
1 & 0 & A \\
0 & 0 & b \\
A^T & 0 & 0
\end{bmatrix}
\begin{bmatrix}
\hat{b} - Ax(u) \\
\lambda^2(d - Bx(u)) \\
x(u)
\end{bmatrix}
\]

(The last row in this equation is just the normal equation system (2.8).)

This suggests the following iteration for improving $x(u)$:
In this scheme, if the matrix that defines the corrections $\Delta x, \Delta y$, and $\Delta z$ is replaced by the matrix of (5.1), then we obtain the iterative improvement scheme of Bjorck and Bjorck [2]. These authors were interested in how to improve the computed solution $x_{\text{dx}}$ obtained via (1.1). We, however, are only interested in getting a solution approximately as good as $x_{\text{dx}}$. Consequently, we need not resort to multiple precision computation of the residuals $\delta_j$.

The iteration (5.3) undergoes considerable simplification when we observe that $\delta_1$ and $\delta_3$ are always zero. (Use induction.) In light of this observation, it follows that $\Delta x$ satisfies

$$||\begin{bmatrix} x_k \\ y_k \end{bmatrix} - \begin{bmatrix} \Delta x \\
\Delta y 
\end{bmatrix}||_2 \leq \min_k ||\begin{bmatrix} x_k \\ y_k \end{bmatrix} - \begin{bmatrix} \Delta x(k) \\
\Delta y(k) \end{bmatrix}||_2$$

where $\delta_2 = d - Bx_k$. Thus, (5.3) collapses to

$$x(k+1) = x(k) + \Delta x(k)$$

For $k = 0, 1, \ldots$

Compute the solution $\Delta x(k)$ to

$$\min_k ||\begin{bmatrix} x_k \\ y_k \end{bmatrix} - \begin{bmatrix} \Delta x(k) \\
\Delta y(k) \end{bmatrix}||_2$$

$$x(k+1) = x(k) + \Delta x(k)$$

Note that only one Q-R factorization, that of $A_k$, is needed to execute (5.4).
The convergence of the method is established in the following result:

**Theorem 4.1**

In the notation of section 2, the vectors \( x(u,k) \) generated by (5.4) satisfy

\[
x(u,k) = x_{\text{LEG}} + a(u,k)
\]

where

\[
a(u,k) = \sum_{i=1}^{p} \frac{\nabla f_i}{\nabla f_i^T \nabla f_i} u_i k x_i
\]

with \( \eta_i = \eta_i^2 / (\mu_i^2 + \nu_i^2) \).

**Proof**

We use induction observing from (2.10) that the theorem is true for \( k = 0 \). Since \( a^{(k)} = -\Delta e(u,k) \), we have by induction that

\[
\delta^{(k)} = \frac{q}{\sum_{i=1}^{p} \eta_i} \frac{\mu_i}{\Sigma^{1/2}} \frac{\nu_i}{\Sigma^{1/2}} \chi_i
\]

Now \( (A^T y B)^{-1} B \) is \( (\nu_i + \mu_i - \mu_i u_i y B)^{-1} B \) and so

\[
a^{(k)} = \frac{2}{\nu_i + \mu_i - \mu_i u_i y B})^{-1} B \delta^{(k)}
\]

\[
= \frac{2}{\nu_i + \mu_i - \mu_i u_i y B})^{-1} B \delta^{(k)}
\]

The theorem follows directly from the equation \( a^{(u,k+1)} = a^{(u,k)} + \delta^{(k)} \).

Arguments very similar to those used in the proof of Theorem 4.1 can be used to establish the following inequalities:

\[
||x(u,k) - x_{\text{LEG}}||_2 \leq \sqrt{\frac{1}{2 \nu_i} \frac{\nu_i}{\eta_i} \delta_i^k}
\]

\[
||d - Bx(u,k)||_2 \leq \sqrt{\frac{1}{2 \nu_i} \frac{\nu_i}{\eta_i} \delta_i^k}
\]
$$0 \leq \| x_{n,k} - B - Ax(n,k) \|_2 \leq \frac{\alpha n}{2\mu} \mu^k.$$  

These bounds closely correspond to those for the extrapolation method.

To examine the effectiveness of the iteration, we applied it to a problem in which \( \mu = 5000 \). The relative error of \( x(n,k) \) is tabulated in the following table:

<table>
<thead>
<tr>
<th>( \mu )</th>
<th>( k=0 )</th>
<th>( k=1 )</th>
<th>( k=2 )</th>
<th>( k=3 )</th>
<th>( k=4 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( 10^3 )</td>
<td>10^0</td>
<td>10^0</td>
<td>10^0</td>
<td>10^0</td>
<td>10^0</td>
</tr>
<tr>
<td>( 10^4 )</td>
<td>10^{-1}</td>
<td>10^{-2}</td>
<td>10^{-3}</td>
<td>10^{-4}</td>
<td>10^{-4}</td>
</tr>
<tr>
<td>( 10^5 )</td>
<td>10^{-9}</td>
<td>10^{-14}</td>
<td>10^{-14}</td>
<td>10^{-14}</td>
<td>10^{-14}</td>
</tr>
</tbody>
</table>

Note that the iteration cannot substantially improve the accuracy of \( x(n,0) \) unless \( \mu^k / (\mu^k + \mu^2) \) is somewhat less than unity.

If the A-over-B formulation to compute \( x(n,0) \) is used, then improvements in the iterates should not be expected if \( \mu > (\text{macheps})^{-1} \).

In practice, \( \| x(n,k) \|_2 \) is a very good estimate of the error in \( x(n,k) \) once convergence begins to set in.

6. Conclusions

Much more work is needed before the ideas in this paper can take on a practical form. Some topics for future research include the following.

1. A closer examination of ill-conditioning is needed. Does the weighting method merely fail to compute the "unstable" components of \( x_{n,k} \)?

11. It is not necessary to re-compute the Q-R factorisation of \( A \) for each different \( \mu \) when extrapolating. Instead, the Q-R factorisation of \( A \) can be updated \([11]\). How could this idea be implemented with the George-Heath algorithm \( [12] \)?
(iii) What are good values for \( r \) and \( \mu \) when extrapolating? How many extrapolation steps can we "afford"?

(iv) Are there ways to implement (5.4) without having to store the complete QR factorization of \( A^T \)?

It is our intention to continue investigating these questions.

Acknowledgements

I would like to thank Gene Golub, Per-Ake Wedin, and the referee for their constructive comments. In addition, the support of the National Science Foundation and the Swedish Natural Science Research Council is gratefully acknowledged. This paper was produced while I was a visitor at the University of Örebro.

REFERENCES


