Lecture # 7
Modified Gram–Schmidt and Normal Equations in Least Squares Solutions

To compute

\[
X = Q_1 R = (x_1, \ldots, x_n) \\
Q_1 = (q_1, \ldots, q_n), \quad R = (r_{jk})
\]

we use the Modified Gram–Schmidt (MGS) Algorithm

\[
r_{11} = \|x_1\|_2; \quad q_1 = x_1/r_{11}; \\
\text{for } k = 2: n \\
\quad s_k = x_k; \\
\quad \text{for } j = 1: k - 1 \\
\quad \quad r_{jk} = q_j^T s_k; \\
\quad \quad s_k = s_k - r_{jk} q_j; \\
\quad \text{end}; \\
\quad r_{kk} = \|s_k\|_2; \\
\text{end};
\]

To solve for \(y_{LS}\) from

\[
\|b - Xy_{LS}\|_2 = \min_{y \in \mathbb{R}^n} \|b - Xy\|_2
\]
treat \(b\) as an extra column of \(X\) (you can do this in Householder, too)!

\[
r = b; \\
\text{for } j = 1: n \\
\quad c_j = q_j^T r; \\
\quad r = r - c_j q_j; \\
\text{end}; \\
\quad r_{LS} = r; \quad \% \text{You do not need to code this line} \\
c = (c_1, \ldots, c_n)^T; \\
\text{Solve } Ry_{LS} = c;
\]

The solution \(y_{LS}\) and the residual \(r_{LS}\) are as good as Householder.
But, DO NOT DO THIS

\[
\begin{align*}
\mathbf{c} &= \mathbf{Q}_1^T \mathbf{b} \\
\mathbf{r}_{LS} &= \mathbf{b} - \mathbf{Q}_1 \mathbf{c} \\
\mathbf{R} \mathbf{y}_{LS} &= \mathbf{c}
\end{align*}
\]

It yields a poor solution.

The first procedure obtains a good solution because it is related to Householder Q–R on

\[
\begin{bmatrix}
\mathbf{0} \\
\mathbf{X}
\end{bmatrix}
= \hat{\mathbf{Q}} \begin{bmatrix}
\mathbf{R} \\
\mathbf{0}
\end{bmatrix}.
\]

The matrix \(\hat{\mathbf{Q}}\) is given by

\[
\hat{\mathbf{Q}} = \hat{\mathbf{H}}_1 \cdots \hat{\mathbf{H}}_n
\]

where

\[
\begin{align*}
\hat{\mathbf{H}}_k &= \mathbf{I} - \mathbf{w}_k \mathbf{w}_k^T, \\
\|\mathbf{w}_k\|_2 &= \sqrt{2} \\
\mathbf{w}_k &= \begin{pmatrix}
-\mathbf{e}_k \\
\mathbf{q}_k
\end{pmatrix}.
\end{align*}
\]

Here \(\mathbf{q}_k\) is the same vector, both mathematically and numerically, as is produced by modified Gram–Schmidt.

The first procedure uses this structure to solve

\[
\| \begin{bmatrix}
\mathbf{0} \\
\mathbf{b}
\end{bmatrix} - \begin{bmatrix}
\mathbf{0} \\
\mathbf{X}
\end{bmatrix} \mathbf{y}_{LS} \|_2 = \min_{\mathbf{y} \in \mathbb{R}^n} \| \begin{bmatrix}
\mathbf{0} \\
\mathbf{b}
\end{bmatrix} - \begin{bmatrix}
\mathbf{0} \\
\mathbf{X}
\end{bmatrix} \mathbf{y} \|_2.
\]

This connection was discovered by Charles Sheffield (by accident) in the 1960’s.

**Cholesky Factorization of the Normal Equations**

Note that the residual

\[
\mathbf{r}_{LS} = \mathbf{b} - \mathbf{X} \mathbf{y}_{LS}
\]

satisfies

\[
\mathbf{X}^T \mathbf{r}_{LS} = 0
\]

yields

\[
\mathbf{X}^T \mathbf{r}_{LS} = \mathbf{X}^T (\mathbf{b} - \mathbf{X} \mathbf{y}_{LS}) = 0
\]

which reorganizes to

\[
\mathbf{X}^T \mathbf{X} \mathbf{y}_{LS} = \mathbf{X}^T \mathbf{b}. \tag{1}
\]
Equation (1) is called the normal equations. The matrix $A = X^T X$ is clearly symmetric, and also positive definite. The latter means that

$$y^T Ay > 0, \quad y \neq 0.$$  

That follows from

$$y^T Ay = y^T X^T X y = \|Xy\|_2^2 > 0$$

for all $y \neq 0$ since $\text{rank}(X) = n$.

Thus we can solve (1) using the Cholesky factorization of $A$, given by

$$A = X^T X = R^T R$$

where $R \in \mathbb{R}^{n \times n}$ is upper triangular with positive diagonal elements. We then solve

$$R^T R y_{LS} = X^T b$$

using forward and back substitution.

For review, the Cholesky algorithm is given by (a different algorithm from that given in class, using the MATLAB ability to expand matrices).

function R=my_chol(A)
R = sqrt(A(1, 1));
for k = 2: n
    Solve $R'v = A(1 : k - 1, k)$;
    rkk = sqrt(A(k, k) - v^T v);
    R = [R \ v; zeros(1, k - 1) rkk];
end:
end: my_chol

It is no coincidence that we labeled this upper triangular matrix $R$.

For Q–R factorization, we have

$$X = Q \begin{pmatrix} R \\ 0 \end{pmatrix} = Q_1 R,$$

where $Q$ is orthogonal and $Q_1$ is left orthogonal. Note that

$$A = X^T X = \begin{pmatrix} R^T \\ 0 \end{pmatrix} Q^T Q \begin{pmatrix} R \\ 0 \end{pmatrix} = R^T Q_1^T Q_1 R$$

$$= R^T R.$$  

Since the $R$ from Cholesky factorization and that from Q–R factorization are both unique, they must be the same $R$! (Mathematically, of course). Below is an example (I changed it slightly from the one in class).
Example 1 Let $X$ be the Läuchli matrix

$$X = \begin{pmatrix}
1 & 1 & 1 & 1 \\
0 & \delta & 0 & 0 \\
0 & 0 & \delta & 0 \\
0 & 0 & 0 & \delta
\end{pmatrix}$$

where $\delta = \sqrt{\text{eps}}$, $\text{eps} = 2^{-52}$. Householder Q–R factorization yields

$$Q_H = \begin{pmatrix}
-1 & 1.05367e-08 & 6.08337e-09 & 4.30159e-09 \\
-1.49012e-08 & -0.707107 & -0.408248 & -0.288675 \\
-0 & 0.707107 & -0.408248 & -0.288675 \\
-0 & 0 & 0.816497 & -0.288675
\end{pmatrix},$$

$$R_H = \begin{pmatrix}
-1 & -1 & -1 & -1 \\
0 & 2.10734e-08 & 1.05367e-08 & 1.05367e-08 \\
0 & 0 & 1.82501e-08 & 6.08337e-09 \\
0 & 0 & 0 & 1.72064e-08
\end{pmatrix}.$$  

Cholesky factorization of the normal equations gives us

$$R = \begin{pmatrix}
1 & 1 & 1 & 1 \\
0 & 1.4901e-08 & 1 & 1 \\
0 & 0 & 1.4901e-08 & 0 \\
0 & 0 & 0 & 1.4901e-08
\end{pmatrix}$$

We gave this problem the right hand side

$$b = X \begin{pmatrix}
1 \\
1 \\
1 \\
1
\end{pmatrix} + \delta * \begin{pmatrix}
-7.4506e-09 \\
0.5 \\
0.5 \\
0.5
\end{pmatrix},$$

which should yield

$$y_{LS} = \begin{pmatrix}
1 \\
1 \\
1 \\
1
\end{pmatrix}, \quad r_{LS} = \delta * \begin{pmatrix}
-7.4506e-09 \\
0.5 \\
0.5 \\
0.5
\end{pmatrix}.$$
Householder Q–R factorization yields the solution $\hat{y}_{LS}$ and residual $\mathbf{r}_{LS}$ such that

$$\|\hat{y}_{LS} - y_{LS}\|_2/\|y_{LS}\|_2 = 2.8305e-16, \quad \|\mathbf{r}_{LS} - \mathbf{r}_{LS}\|_2/\|\mathbf{r}_{LS}\|_2 = 5.5511e-16$$

Normal equations gets the solution

$$\hat{y}_{LS} = \begin{pmatrix} 4 \\ 0 \\ 0 \end{pmatrix}, \quad \mathbf{r}_{LS} = \begin{pmatrix} 0 \\ -3.7253e-08 \\ 2.2352e-08 \end{pmatrix}$$

Neither $\hat{y}_{LS}$ nor $\mathbf{r}_{LS}$ has any correct digits.

If we take this example further, say, for the Läuchli matrix

$$X = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1e-10 & 0 & 0 & 0 \\ 0 & 1e-10 & 0 & 0 \\ 0 & 0 & 1e-10 & 0 \\ 0 & 0 & 0 & 1e-10 \end{pmatrix}$$

The Householder and Gram–Schmidt Q–R decompositions of $X$ are fine in floating point arithmetic, but when we compute the normal equations in MATLAB we get

$$A = \text{f}l(X^T X) = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{pmatrix}$$

rounded from

$$X^T X = \begin{pmatrix} 1 + 1e-20 & 1 & 1 & 1 \\ 1 & 1 + 1e-20 & 1 & 1 \\ 1 & 1 & 1 + 1e-20 & 1 \\ 1 & 1 & 1 & 1 + 1e-20 \end{pmatrix}.$$ 

Thus the normal equations algorithm simply fails.