Math 515  
Fall, 2014  

Homework 5, due Tuesday, November 4.

In many applications the solution to a least squares (LS) problem needs to be modified as new data is collected. For example, assume an LS problem \( \min ||b - Ax||_2 \) is solved using a QR factorization of the matrix \( A \):

\[
A = QR.
\]

Suppose an additional equation is added to the system:

\[
\tilde{A} = \begin{bmatrix} A & w^T \end{bmatrix} \quad \text{and} \quad \tilde{b} = \begin{bmatrix} b \\ \beta \end{bmatrix}.
\]

To solve the least squares problem \( \min ||\tilde{b} - \tilde{A}\tilde{x}||_2 \) we must compute a QR factorization of \( \tilde{A} \). Rather than start from scratch, however, we would like to use information already known to us, namely \( A = QR \).

**Problem 1.** Suppose we compute a QR factorization of the matrix

\[
\begin{bmatrix} R \\ w^T \end{bmatrix}
\]

How can this be used to compute the QR factorization of \( \tilde{A} \)?

The above problem suggests that an efficient algorithm to compute the QR factorization of \( \tilde{A} \) depends on finding an efficient scheme for computing the QR factorization of \( \begin{bmatrix} R \\ w^T \end{bmatrix} \).

**Problem 2.** Develop an algorithm that uses Givens rotations to efficiently update \( R \) to \( \tilde{R} \) and \( Q^Tb \) to \( \tilde{Q}^T\tilde{b} \), where \( A = QR \) and \( \tilde{A} = \tilde{Q}\tilde{R} \). Your algorithm should require only \( O(n^2) \) flops; verify this by counting the total flops needed by your algorithm.
**Problem 3.** Now implement the algorithm from Problem 2. In particular:

(a) Write a function

\[ [c, s] = \text{givens}(a, b) \]

that constructs a Givens rotation such that

\[
\begin{bmatrix}
  c & s \\
- s & c \\
\end{bmatrix}
\begin{bmatrix}
a \\
b \\
\end{bmatrix}
= \begin{bmatrix}
r \\
0 \\
\end{bmatrix}.
\]

(b) Write a function

\[ [\text{Rnew}, d\text{new}] = \text{RowUpdate}(R, d, w, \beta) \]

that updates \( R \) and \( d = QTb \) to \( \text{Rnew} = \tilde{R} \) and \( d\text{new} = \tilde{d} = \tilde{Q}^T\tilde{b} \).

**Problem 4.** Test your codes with the following data:

\[
R = \begin{bmatrix}
1 & 2 & 3 & 4 & 5 \\
0 & 6 & 7 & 8 & 9 \\
0 & 0 & 10 & 11 & 12 \\
0 & 0 & 13 & 14 & 15 \\
0 & 0 & 0 & 0 & 15 \\
\end{bmatrix}, \quad d = QTb = \begin{bmatrix}
15 \\
30 \\
33 \\
27 \\
15 \\
\end{bmatrix}
\]

\[
w^T = \begin{bmatrix}
3 & -1 & 0 & 1 & 5 \\
\end{bmatrix}, \quad \beta = 8.
\]

Compute the corresponding least squares solutions using your codes. Do your results make sense?

**Problem 5.** Compare the time it takes to update a sequence of LS problems as new rows of data are given using the method outlined in this homework with using MATLAB’s built-in \texttt{qr} function each time. For example, you could

- Use \texttt{rand} to create an initial \( 500 \times 500 \) matrix \( A \)
- Use \texttt{qr} to get the initial \( QR \) factorization of \( A \)
- Start a loop where each time through the loop you create a new random row vector \( w^T \) and a new scalar \( \beta \). Compute the \( QR \) factorization of \( \tilde{A} \) using the method from this assignment with the new \( w^T \) and \( \beta \). Run the loop 500 times, so that in the end you’ve computed the \( QR \) factorization of a \( 1000 \times 500 \) matrix.
- Repeat the above loop, but this time explicitly form the new matrix \( \tilde{A} \) and use \texttt{qr} to compute its \( QR \) factorization.