More on QR

- There are other ways to calculate QR factors
  - Householder transformations
  - Givens rotations
- We discussed what Matlab calls the 'economy' QR factorization, where Q is mxn orthogonal, R is nxn upper triangular.
- More generally, it is written as

$$\mathrm{A} = \mathrm{QR} = \begin{bmatrix} Q_1 & Q_2 \end{bmatrix} \begin{bmatrix} R_1 \\ 0 \end{bmatrix} = Q_1 R_1,$$

where Q is mxm, R is mxn (while Q_1 is mxn and R_1 is nxn).

Matlab's `qr` function

- `[Q,R] = qr(A),` where A is m-by-n, produces an m-by-n upper triangular matrix R and an m-by-m orthogonal matrix Q so that A = Q*R.

- `[Q,R] = qr(A,0)` produces the "economy size" decomposition. If m>n, only the first n columns of Q and the first n rows of R are computed. If m<=n, this is the same as [Q,R] = qr(A).

- If A is sparse: `R = qr(A)` computes a "Q-less qr decomposition" and returns the upper triangular factor R. Note that `R = chol(A'*A).` Since Q is often nearly full, even when A is sparse, this is preferred to `[Q,R] = qr(A).`

Matlab's `qr` function

- `[Q,R,E] = qr(A)` produces orthogonal Q, upper triangular R and a permutation matrix E so that A*E = Q*R. The column permutation E is chosen so that ABS(DIAG(R)) is decreasing. Note [Q,R,e] = qr(A,'vector') produces the permutations in vector e instead of matrix E.
- Other variations are available.

Suppose rank(A) = k < n

- Permute columns, using "largest" columns first
- $Q^T AP = \begin{bmatrix} \tilde{R} & S \\ 0 & 0 \end{bmatrix}$, where
- Q is mxm orthogonal, P is nxn permutation (for columns), $\tilde{R}$ is kxk triangular with $\widetilde{|r_{11}|} \geq \widetilde{|r_{22}|} \geq \cdots \widetilde{|r_{kk}|}$, S is kx(n-k).
- solution not unique → Multiple solutions, including
  - Let $\tilde{c} = (Q^T b)_{1:k}$
  - Solve $\tilde{R}z = \tilde{c}$
  - Then $x = P \begin{bmatrix} z \\ 0 \end{bmatrix}$