QR Factorization

QR Factorization

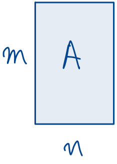
The QR Factorization

- $A\vec{x} = \vec{b}$ (sq. linear system) via LU factorization
- $A\tilde{x}$ "=" $\vec{b}$ (LLS prob; overdet.) via QR factorization

The following matrix factorization plays a role in solving linear least squares problems similar to that of LU factorization in solving linear systems.

Theorem 1

Let $A \in \mathbb{R}^{m\times n}$ where $m \geqslant n$. Then $A = QR$ where $Q \in \mathbb{R}^{m\times m}$ is orthogonal and $R \in \mathbb{R}^{m\times n}$ is upper triangular.

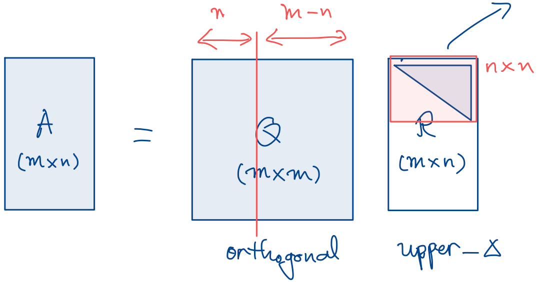


$m \geqslant n$

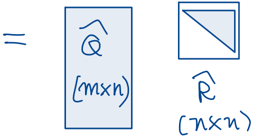
$$\underbrace{\left[\begin{array}{c|c|c|c} \mathbf{a}_1 & \mathbf{a}_2 & \cdots & \mathbf{a}_n \end{array}\right]}_{A} = \underbrace{\left[\begin{array}{c|c|c|c} \mathbf{q}_1 & \mathbf{q}_2 & \cdots & \mathbf{q}_m \end{array}\right]}_{Q} \underbrace{\begin{bmatrix} r_{11} & r_{12} & \cdots & r_{1n} \\ 0 & r_{22} & \cdots & r_{2n} \\ \vdots & & \ddots & \vdots \\ 0 & 0 & \cdots & r_{nn} \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & & \vdots \\ 0 & 0 & \cdots & 0 \end{bmatrix}}_{R}$$

Cartoon view of QR factorization

"unshaded part all zeros"



Thick QR or Full QR



Thin QR or Compressed QR

<u>Recall</u> Q is an orthogonal matrix if

① Q is a square matrix; $\quad Q \in \mathbb{R}^{m\times m}$

② <u>Co</u>lumns of Q are <u>orthonormal</u>. $\left(\underbrace{Q^T Q = I}_{m\times m} \right)$

$\hat{Q} \in \mathbb{R}^{m\times n}$ is not an orthogonal matrix.

Nonetheless, the columns of $\hat{Q}$ are <u>orthonormal</u>!

Consequently,

$$\underbrace{\hat{Q}^T}_{n\times m} \underbrace{\hat{Q}}_{m\times n} = \underbrace{I}_{n\times n}$$

Thick VS Thin QR Factorization

- Here is the QR factorization again.

$$A = \underbrace{\left[\begin{array}{c|c|c|c} & & & \\ \mathbf{q}_1 & \mathbf{q}_2 & \cdots & \mathbf{q}_m \\ & & & \end{array}\right]}_{Q} \underbrace{\begin{bmatrix} r_{11} & r_{12} & \cdots & r_{1n} \\ 0 & r_{22} & \cdots & r_{2n} \\ \vdots & & \ddots & \vdots \\ 0 & 0 & \cdots & r_{nn} \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & & \vdots \\ 0 & 0 & \cdots & 0 \end{bmatrix}}_{R} \qquad \text{(thick)}$$

- When m is much larger than n, it is much more efficient to use the *thin* or *compressed* QR factorization.

$$A = \underbrace{\left[\begin{array}{c|c|c|c} & & & \\ \mathbf{q}_1 & \mathbf{q}_2 & \cdots & \mathbf{q}_n \\ & & & \end{array}\right]}_{\hat{Q}} \underbrace{\begin{bmatrix} r_{11} & r_{12} & \cdots & r_{1n} \\ 0 & r_{22} & \cdots & r_{2n} \\ \vdots & & \ddots & \vdots \\ 0 & 0 & \cdots & r_{nn} \end{bmatrix}}_{\hat{R}} \qquad \text{(thin)}$$

QR Factorization in MATLAB

Either type of QR factorization is computed by `qr` command.

- Thick/Full QR factorization

```
[Q, R] = qr(A)
```

- Thin/Compressed QR factorization

```
[Q, R] = qr(A, 0)
```

Test the orthogonality of `Q` by calculating the norm of $Q^{\mathrm{T}}Q - I$ where I is the identity matrix with *suitable* dimensions.

```
norm(Q'*Q - eye(m))        % full QR
norm(Q'*Q - eye(n))        % thin QR
```

Least Squares and QR Factorization

Moore-Penrose Pseudoinverse

Let $A \in \mathbb{R}^{m \times n}$ with $m \geqslant n$ and suppose that columns of A are linearly independent.

- The least square problem $A\mathbf{x}$ "=" $\mathbf{b}$ is equivalent to the normal equation $A^{\mathrm{T}}A\mathbf{x} = A^{\mathrm{T}}\mathbf{b}$, which is a square matrix equation.

- The solution can be written as

$$\mathbf{x} = \underbrace{\left(A^{\mathrm{T}}A\right)^{-1} A^{\mathrm{T}}}_{\text{pseudo inverse}} \underbrace{\mathbf{b}}_{\text{RHS}}.$$

- The matrix

$$A^{\dagger} := A^{+} = \left(A^{\mathrm{T}}A\right)^{-1} A^{\mathrm{T}} \in \mathbb{R}^{n \times m},$$

is called the **(Moore-Penrose) pseudoinverse**.

$A\vec{x} = \vec{b}$ (sq. linear)

$$\Rightarrow \vec{x} = \underbrace{A^{-1}}_{\text{inverse}} \underbrace{\vec{b}}_{\text{RHS}}$$

Moore-Penrose Pseudoinverse (cont')

- MATLAB's backslash is mathematically equivalent to left-multiplication by the inverse or pseudoinverse of a matrix.
- MATLAB's `pinv` calculates the pseudoinverse, but it is rarely used in practice, just as `inv`.
- A^+ can be calculated by using the thin QR factorization $A = \widehat{Q}\widehat{R}$.

$$\boxed{A^+ = \widehat{R}^{-1}\widehat{Q}^{\mathrm{T}}.}$$

It can be done using the thick QR factorization as seen on p. 1624 of the text.

Least Squares Using QR Factorization

We now reveal the connection between QR factorization and the LLS approximation.

Substitute the thin factorization $A = \hat{Q}\hat{R}$ into the normal equation $A^{\mathrm{T}}A\mathbf{x} = A^{\mathrm{T}}\mathbf{b}$ and simplify.

$$(\hat{Q}\hat{R})^T(\hat{Q}\hat{R})\vec{x} = (\hat{Q}\hat{R})^T\vec{b}$$

$$\cancel{\hat{R}^T}\underbrace{\hat{Q}^T\hat{Q}}_{= I}\hat{R}\vec{x} = \cancel{\hat{R}^T}\hat{Q}^T\vec{b}$$

$$\hat{R}\vec{x} = \hat{Q}^T\vec{b}$$

$$\vec{x} = \underbrace{\hat{R}^{-1}\hat{Q}^T}_{\text{pseudo inverse}}\underbrace{\vec{b}}_{\text{RHS}}$$

To solve LLS using QR

1. Compute thin QR $A = \hat{Q}\hat{R}$
2. Compute $\vec{z} = \hat{Q}^T\vec{b}$.
3. Solve $\underbrace{\hat{R}\vec{x} = \vec{z}}_{\text{upper-}\Delta\text{ system}}$ using back. subs. routine.

Least Squares Using QR Factorization (cont')

Summary: Algorithm for LLS Approximation

If A has rank n, the normal equation $A^{\mathrm{T}}A\mathbf{x} = A^{\mathrm{T}}\mathbf{b}$ is consistent and is equivalent to $\widehat{R}\mathbf{x} = \widehat{Q}^{\mathrm{T}}\mathbf{b}$.

1. Factor $A = \widehat{Q}\widehat{R}$.
2. Let $\mathbf{z} = \widehat{Q}^{\mathrm{T}}\mathbf{b}$.
3. Solve $\widehat{R}\mathbf{x} = \mathbf{z}$ for $\mathbf{x}$ using backward substitution.

Least Squares Using QR Factorization (cont')

```
function x = lsqrfact(A,b)
% LSQRFACT x = lsqrfact(A,b)
% Solve linear least squares by QR factorization
% Input:
%   A    coefficient matrix (m-by-n, m>n)
%   b    right-hand side (m-by-1)
% Output:
%   x    minimizer of || b - Ax || (2-norm)
    [Q,R] = qr(A,0);                 % thin QR fact.
    z = Q'*b;
    x = backsub(R,z);
end
```

Appendix: More on Pseudoinverse and Normal Equation

Analytical Properties of Pseudoinverse

The matrix $A^{\mathrm{T}}A$ appearing in the definition of A^{+} satisfies the following properties.

Theorem 2

For any $A \in \mathbb{R}^{m \times n}$ with $m \geqslant n$, the following are true:

1. *$A^{\mathrm{T}}A$ is symmetric.*
2. *$A^{\mathrm{T}}A$ is singular if and only if rank$(A) < n$.*
3. *If $A^{\mathrm{T}}A$ is nonsingular, then it is positive definite.*

A symmetric positive definite (SPD) matrix S such as $A^{\mathrm{T}}A$ permits so-called the **Cholesky factorization**

$$S = R^{\mathrm{T}}R$$

where R is an upper triangular matrix.

Least Squares Using Cholesky Factorization

One can solve the LLS problem $A\mathbf{x}$ "=" $\mathbf{b}$ by solving the normal equation $A^{\mathrm{T}}A\mathbf{x} = A^{\mathrm{T}}\mathbf{b}$ directly as below.

1. Compute $N = A^{\mathrm{T}}A$.
2. Compute $\mathbf{z} = A^{\mathrm{T}}\mathbf{b}$.
3. Solve the square linear system $N\mathbf{x} = \mathbf{z}$ for $\mathbf{x}$.

Step 3 is done using `chol` which implements the Cholesky factorization.

MATLAB Implementarion.

```
N = A'*A;
z = A'*b;
R = chol(N);
w = forelim(R',z);     % solve R'w = z
x = backsub(R,w);      % solve R x = w
```

Conditioning of Normal Equations

- Recall that the condition number of solving a square linear system $A\mathbf{x} = \mathbf{b}$ is $\kappa(A) = \|A\| \left\|A^{-1}\right\|$.
- Provided that the residual norm at the least square solution is relatively small, the conditioning of LLS problem is similar:

$$\kappa(A) = \|A\| \left\|A^{+}\right\| .$$

- If A is rank-deficient (columns are linearly dependent), then $\kappa(A) = \infty$.
- If an LLS problem is solved solving the normal equation, it can be shown that the condition number is

$$\kappa(A^{\mathrm{T}}A) = \kappa(A)^2.$$