10. The QR factorization

- solving the normal equations
- the QR factorization
- orthogonal matrices
- modified Gram-Schmidt algorithm
- Cholesky factorization versus QR factorization

Least-squares methods

least-squares problem

$$\text{minimize} \quad \|Ax - b\|^2 \qquad (A \in \mathbf{R}^{m\times n},\ m \geq n,\ \mathbf{rank}(A) = n)$$

normal equations

$$A^TAx = A^Tb$$

- method 1: solve the normal equations using the Cholesky factorization
- method 2: use the QR factorization

method 2 has better numerical properties; method 1 is faster

Least-squares method 1: Cholesky factorization

$$A^TAx = A^Tb$$

n equations in n variables, A^TA is symmetric positive definite

algorithm:

1. calculate $C = A^TA$ (C is *symmetric:* $\frac{1}{2}n(n+1)(2m-1) \approx mn^2$ flops)
2. Cholesky factorization $C = LL^T$ ($(1/3)n^3$ flops)
3. calculate $d = A^Tb$ ($2mn$ flops)
4. solve $Lz = d$ by forward substitution (n^2 flops)
5. solve $L^Tx = z$ by backward substitution (n^2 flops)

total for large m, n: $mn^2 + (1/3)n^3$ flops

example

$$A = \begin{bmatrix} 3 & -6 \\ 4 & -8 \\ 0 & 1 \end{bmatrix}, \qquad b = \begin{bmatrix} -1 \\ 7 \\ 2 \end{bmatrix}$$

1. calculate $A^TA = \begin{bmatrix} 25 & -50 \\ -50 & 101 \end{bmatrix}$ and $A^Tb = \begin{bmatrix} 25 \\ -48 \end{bmatrix}$
2. Cholesky factorization: $A^TA = \begin{bmatrix} 5 & 0 \\ -10 & 1 \end{bmatrix} \begin{bmatrix} 5 & -10 \\ 0 & 1 \end{bmatrix}$
3. forward substitution: solve $\begin{bmatrix} 5 & 0 \\ -10 & 1 \end{bmatrix} \begin{bmatrix} z_1 \\ z_2 \end{bmatrix} = \begin{bmatrix} 25 \\ -48 \end{bmatrix}$
   $z_1 = 5$, $z_2 = 2$
4. backward substitution: solve $\begin{bmatrix} 5 & -10 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 5 \\ 2 \end{bmatrix}$
   $x_1 = 5$, $x_2 = 2$

The QR factorization

if $A \in \mathbf{R}^{m\times n}$ with $m \geq n$ and $\mathbf{rank}\, A = n$ then it can be factored as

$$A = QR$$

- $R \in \mathbf{R}^{n\times n}$ is upper triangular with $r_{ii} > 0$
- $Q \in \mathbf{R}^{m\times n}$ satisfies $Q^TQ = I$ (Q is an *orthogonal matrix*)

can be computed in $2mn^2$ flops (more later)

Least-squares method 2: QR factorization

rewrite normal equations $A^TAx = A^Tb$ using QR factorization $A = QR$:

$$\begin{aligned} A^TAx &= A^Tb \\ R^TQ^TQRx &= R^TQ^Tb \\ R^TRx &= R^TQ^Tb \qquad (Q^TQ = I) \\ Rx &= Q^Tb \qquad (R \text{ nonsingular}) \end{aligned}$$

algorithm

1. QR factorization of A: $A = QR$ ($2mn^2$ flops)
2. form $d = Q^Tb$ ($2mn$ flops)
3. solve $Rx = d$ by backward substitution (n^2 flops)

total for large m, n: $2mn^2$ flops

example

$$A = \begin{bmatrix} 3 & -6 \\ 4 & -8 \\ 0 & 1 \end{bmatrix}, \qquad b = \begin{bmatrix} -1 \\ 7 \\ 2 \end{bmatrix}$$

1. QR factorization: $A = QR$ with

$$Q = \begin{bmatrix} 3/5 & 0 \\ 4/5 & 0 \\ 0 & 1 \end{bmatrix}, \qquad R = \begin{bmatrix} 5 & -10 \\ 0 & 1 \end{bmatrix}$$

2. calculate $d = Q^T b = \begin{bmatrix} 5 \\ 2 \end{bmatrix}$

3. backward substitution: solve $\begin{bmatrix} 5 & -10 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 5 \\ 2 \end{bmatrix}$

   $x_1 = 5$, $x_2 = 2$

Orthogonal matrices

$Q = [q_1 \; q_2 \; \cdots \; q_n] \in \mathbf{R}^{m \times n}$ $(m \geq n)$ is *orthogonal* if $Q^T Q = I$

$$Q^T Q = \begin{bmatrix} q_1^T q_1 & q_1^T q_2 & \cdots & q_1^T q_n \\ q_2^T q_1 & q_2^T q_2 & \cdots & q_2^T q_n \\ \vdots & \vdots & \ddots & \vdots \\ q_n^T q_1 & q_n^T q_2 & \cdots & q_n^T q_n \end{bmatrix}$$

properties

- the columns q_i have unit norm: $q_i^T q_i = 1$ for $i = 1, \ldots, n$
- the columns are mutually orthogonal: $q_i^T q_j = 0$ for $i \neq j$
- $\mathbf{rank}\, Q = n$, *i.e.*, the columns of Q are linearly independent

$$Qx = 0 \quad \Longrightarrow \quad Q^T Q x = 0 \quad \Longrightarrow \quad x = 0$$

- if Q is **square** $(m = n)$, then Q is nonsingular and $Q^{-1} = Q^T$

examples of orthogonal matrices

- permutation matrices, *e.g.*, $Q = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

- $Q = \begin{bmatrix} \cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta \end{bmatrix}$

- $Q = \begin{bmatrix} \cos\theta & -\sin\theta \\ 0 & 0 \\ \sin\theta & \cos\theta \end{bmatrix}$

- $Q = I - 2uu^T$ where $u \in \mathbf{R}^n$ with $\|u\| = 1$

$$Q^TQ = (I - 2uu^T)(I - 2uu^T) = I - 2uu^T - 2uu^T + 4uu^Tuu^T = I$$

Computing the QR factorization

given $A \in \mathbf{R}^{m\times n}$ with $\mathbf{rank}\, A = n$

partition $A = QR$ as

$$\begin{bmatrix} a_1 & A_2 \end{bmatrix} = \begin{bmatrix} q_1 & Q_2 \end{bmatrix} \begin{bmatrix} r_{11} & R_{12} \\ 0 & R_{22} \end{bmatrix}$$

- $a_1 \in \mathbf{R}^m$, $A_2 \in \mathbf{R}^{m\times(n-1)}$
- $q_1 \in \mathbf{R}^m$, $Q_2 \in \mathbf{R}^{m\times(n-1)}$ satisfy

$$\begin{bmatrix} q_1^T \\ Q_2^T \end{bmatrix} \begin{bmatrix} q_1 & Q_2 \end{bmatrix} = \begin{bmatrix} q_1^Tq_1 & q_1^TQ_2 \\ Q_2^Tq_1 & Q_2^TQ_2 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & I \end{bmatrix},$$

i.e.,

$$q_1^Tq_1 = 1, \qquad Q_2^TQ_2 = I, \qquad q_1^TQ_2 = 0$$

- $r_{11} \in \mathbf{R}$, $R_{12} \in \mathbf{R}^{1\times(n-1)}$, $R_{22} \in \mathbf{R}^{(n-1)\times(n-1)}$ is upper triangular

recursive algorithm (*'modified Gram-Schmidt algorithm'*)

$$\begin{bmatrix} a_1 & A_2 \end{bmatrix} = \begin{bmatrix} q_1 & Q_2 \end{bmatrix} \begin{bmatrix} r_{11} & R_{12} \\ 0 & R_{22} \end{bmatrix} = \begin{bmatrix} q_1 r_{11} & q_1 R_{12} + Q_2 R_{22} \end{bmatrix}$$

1. determine q_1 and r_{11}:

$$r_{11} = \|a_1\|, \qquad q_1 = (1/r_{11}) a_1$$

2. R_{12} follows from $q_1^T A_2 = q_1^T (q_1 R_{12} + Q_2 R_{22}) = R_{12}$:

$$R_{12} = q_1^T A_2$$

3. Q_2 and R_{22} follow from

$$A_2 - q_1 R_{12} = Q_2 R_{22},$$

   $i.e.$, the QR factorization of an $m \times (n-1)$ matrix

cost: $2mn^2$ flops (no proof)

proof that the algorithm works for $A \in \mathbf{R}^{m \times n}$ with rank n

- step 1: $a_1 \neq 0$ because $\mathbf{rank}\, A = n$
- step 3: $A_2 - q_1 R_{12}$ has full rank (rank $n-1$):

$$A_2 - q_1 R_{12} = A_2 - (1/r_{11}) a_1 R_{12}$$

  hence if $(A_2 - q_1 R_{12})x = 0$, then

$$\begin{bmatrix} a_1 & A_2 \end{bmatrix} \begin{bmatrix} -R_{12} x / r_{11} \\ x \end{bmatrix} = 0$$

  but this implies $x = 0$ because $\mathbf{rank}(A) = n$

- therefore the algorithm works for an $m \times n$ matrix with rank n, if it works for an $m \times (n-1)$ matrix with rank $n-1$
- obviously it works for an $m \times 1$ matrix with rank 1; so by induction it works for all $m \times n$ matrices with rank n

example

$$A = \begin{bmatrix} 9 & 0 & 26 \\ 12 & 0 & -7 \\ 0 & 4 & 4 \\ 0 & -3 & -3 \end{bmatrix}$$

we want to factor A as

$$A = \begin{bmatrix} a_1 & a_2 & a_3 \end{bmatrix} = \begin{bmatrix} q_1 & q_2 & q_3 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ 0 & r_{22} & r_{23} \\ 0 & 0 & r_{33} \end{bmatrix}$$

$$= \begin{bmatrix} q_1 r_{11} & q_1 r_{12} + q_2 r_{22} & q_1 r_{13} + q_2 r_{23} + q_3 r_{33} \end{bmatrix}$$

with

$$\begin{array}{lll} q_1^T q_1 = 1, & q_2^T q_2 = 1, & q_3^T q_3 = 1 \\ q_1^T q_2 = 0, & q_1^T q_3 = 0, & q_2^T q_3 = 0 \end{array}$$

and $r_{11} > 0$, $r_{22} > 0$, $r_{33} > 0$

- determine first column of Q, first row of R
  - $a_1 = q_1 r_{11}$ with $\|q_1\| = 1$

$$r_{11} = \|a_1\| = 15, \qquad q_1 = (1/r_{11}) a_1 = \begin{bmatrix} 3/5 \\ 4/5 \\ 0 \\ 0 \end{bmatrix}$$

  - inner product of q_1 with a_2 and a_3:

$$\begin{aligned} q_1^T a_2 &= q_1^T (q_1 r_{12} + q_2 r_{22}) = r_{12} \\ q_1^T a_3 &= q_1^T (q_1 r_{13} + q_2 r_{23} + q_3 r_{33}) = r_{13} \end{aligned}$$

    therefore, $r_{12} = q_1^T a_2 = 0$, $r_{13} = q_1^T a_3 = 10$

$$A = \begin{bmatrix} 9 & 0 & 26 \\ 12 & 0 & -7 \\ 0 & 4 & 4 \\ 0 & -3 & -3 \end{bmatrix} = \begin{bmatrix} 3/5 & q_{12} & q_{13} \\ 4/5 & q_{22} & q_{23} \\ 0 & q_{32} & q_{33} \\ 0 & q_{42} & q_{43} \end{bmatrix} \begin{bmatrix} 15 & 0 & 10 \\ 0 & r_{22} & r_{23} \\ 0 & 0 & r_{33} \end{bmatrix}$$

- determine 2nd column of Q, 2nd row or R

$$\begin{bmatrix} 0 & 26 \\ 0 & -7 \\ 4 & 4 \\ -3 & -3 \end{bmatrix} - \begin{bmatrix} 3/5 \\ 4/5 \\ 0 \\ 0 \end{bmatrix} \begin{bmatrix} 0 & 10 \end{bmatrix} = \begin{bmatrix} q_2 & q_3 \end{bmatrix} \begin{bmatrix} r_{22} & r_{23} \\ 0 & r_{33} \end{bmatrix}$$

$i.e.$, the QR factorization of $\begin{bmatrix} 0 & 20 \\ 0 & -15 \\ 4 & 4 \\ -3 & -3 \end{bmatrix} = \begin{bmatrix} q_2 r_{22} & q_2 r_{23} + q_3 r_{33} \end{bmatrix}$

  - first column is $q_2 r_{22}$ where $\|q_2\| = 1$, hence

$$r_{22} = 5, \qquad q_2 = \begin{bmatrix} 0 \\ 0 \\ 4/5 \\ -3/5 \end{bmatrix}$$

  - inner product of q_2 with 2nd column gives r_{23}

$$q_2^T \begin{bmatrix} 20 \\ -15 \\ 4 \\ -3 \end{bmatrix} = q_2^T (q_2 r_{23} + q_3 r_{33}) = r_{23}$$

therefore, $r_{23} = 5$

QR factorization so far:

$$A = \begin{bmatrix} 9 & 0 & 26 \\ 12 & 0 & -7 \\ 0 & 4 & 4 \\ 0 & -3 & -3 \end{bmatrix} = \begin{bmatrix} 3/5 & 0 & q_{13} \\ 4/5 & 0 & q_{23} \\ 0 & 4/5 & q_{33} \\ 0 & -3/5 & q_{43} \end{bmatrix} \begin{bmatrix} 15 & 0 & 10 \\ 0 & 5 & 5 \\ 0 & 0 & r_{33} \end{bmatrix}$$

- determine 3rd column of Q, 3rd row of R

$$\begin{bmatrix} 26 \\ -7 \\ 4 \\ -3 \end{bmatrix} - \begin{bmatrix} 3/5 & 0 \\ 4/5 & 0 \\ 0 & 4/5 \\ 0 & -3/5 \end{bmatrix} \begin{bmatrix} 10 \\ 5 \end{bmatrix} = q_3 r_{33}$$

$$\begin{bmatrix} 20 \\ -15 \\ 0 \\ 0 \end{bmatrix} = q_3 r_{33}$$

with $\|q_3\| = 1$, hence

$$r_{33} = 25, \qquad q_3 = \begin{bmatrix} 4/5 \\ -3/5 \\ 0 \\ 0 \end{bmatrix}$$

in summary,

$$A = \begin{bmatrix} 9 & 0 & 26 \\ 12 & 0 & -7 \\ 0 & 4 & 4 \\ 0 & -3 & -3 \end{bmatrix} = \begin{bmatrix} 3/5 & 0 & 4/5 \\ 4/5 & 0 & -3/5 \\ 0 & 4/5 & 0 \\ 0 & -3/5 & 0 \end{bmatrix} \begin{bmatrix} 15 & 0 & 10 \\ 0 & 5 & 5 \\ 0 & 0 & 25 \end{bmatrix}$$

$$= QR$$

Cholesky factorization versus QR factorization

example: minimize $\|Ax - b\|^2$ with

$$A = \begin{bmatrix} 1 & -1 \\ 0 & 10^{-5} \\ 0 & 0 \end{bmatrix}, \qquad b = \begin{bmatrix} 0 \\ 10^{-5} \\ 1 \end{bmatrix}$$

solution:

normal equations $A^T Ax = A^T b$:

$$\begin{bmatrix} 1 & -1 \\ -1 & 1 + 10^{-10} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 10^{-10} \end{bmatrix}$$

solution: $x_1 = 1$, $x_2 = 1$

let us compare both methods, rounding intermediate results to 8 significant decimal digits

method 1 (Cholesky factorization)

$A^T A$ and $A^T b$ rounded to 8 digits:

$$A^T A = \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}, \qquad A^T b = \begin{bmatrix} 0 \\ 10^{-10} \end{bmatrix}$$

no solution (singular matrix)

method 2 (QR factorization): factor $A = QR$ and solve $Rx = Q^T b$

$$Q = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \end{bmatrix}, \qquad R = \begin{bmatrix} 1 & -1 \\ 0 & 10^{-5} \end{bmatrix}, \qquad Q^T b = \begin{bmatrix} 0 \\ 10^{-5} \end{bmatrix}$$

rounding does not change any values

solution of $Rx = Q^T b$ is $x_1 = 1$, $x_2 = 1$

conclusion:

- for this example, Cholesky factorization method fails due to rounding errors; QR factorization method gives the exact solution
- from numerical analysis: Cholesky factorization method can be very inaccurate if $\kappa(A^T A)$ is high
- numerical stability of QR factorization method is better

Summary

cost for dense A

- method 1 (Cholesky factorization): $mn^2 + (1/3)n^3$ flops
- method 2 (QR factorization): $2mn^2$ flops
- method 1 is always faster (twice as fast if $m \gg n$)

cost for large sparse A

- method 1: we can form $A^T A$ fast, and use a sparse Cholesky factorization (cost $\ll mn^2 + (1/3)n^3$)
- method 2: no good methods for sparse QR factorization
- method 1 is much more efficient

numerical stability: method 2 is more accurate

in practice: preferred method is method 2; method 1 is used when A is large and sparse