Welcome to Ma221! (Feb 24)

Normal Equations:

Solve $A^TAx = A^Tb$

cost = $m \cdot n^2 (A^TA) + \frac{n^3}{3}$ (Cholesky)

words moved: know how to minimize both steps

QR: $A = QR$, $A^{m\times n}$, $Q^{m\times n}$, $R^{n\times n}$

Q: orthonormal columns

minimize $\|Ax-b\|_2$ $\quad x = R^{-1}Q^Tb$

(in Matlab: $x = A\backslash b$ will do this)

proof 1: $A = QR = {}_m\left[\overset{n}{Q}, \overset{m-n}{\hat{Q}}\right]\begin{bmatrix} R \\ 0 \end{bmatrix}$ orthogonal $[Q,\hat{Q}]$

$$\|Ax-b\|_2^2 = \left\| \begin{bmatrix} Q^T \\ \hat{Q}^T \end{bmatrix} (Ax-b) \right\|_2^2$$

$$= \left\| \begin{bmatrix} Q^T \\ \hat{Q}^T \end{bmatrix} (QRx-b) \right\|_2^2$$

$$= \left\| \begin{matrix} Q^T(QRx-b) \\ \hat{Q}^T(QRx-b) \end{matrix} \right\|_2^2$$

$$= \left\| \begin{matrix} Rx - Q^Tb \\ -\hat{Q}^Tb \end{matrix} \right\|_2^2$$

$$= \|Rx - Q^Tb\|_2^2 + \|\hat{Q}^Tb\|_2^2$$

$$\geq \|\hat{Q}^Tb\|_2^2, \quad = \text{if} \quad x = R^{-1}Q^Tb$$

proof 2: plug $A = QR$ into NE

$$x = (A^T A)^{-1} A^T b$$

$$= (R^T \underbrace{Q^T Q}_{I} R)^{-1} R^T Q^T b$$

$$= (R^T R)^{-1} R^T Q^T b$$

$$= R^{-1} \underbrace{R^{-T} R^T}_{I} Q^T b$$

$$= R^{-1} Q^T b$$

Algorithms for $A = QR$

Classical + Modified Gram-Schmidt

CGS + MGS

Equate columns of A and QR

$$A(:,i) = \sum_{j=1}^{i} Q(:,j) \cdot R(j,i)$$

since columns of Q orthogonal

$$(Q(:,j))^T A(:,i) = R(j,i)$$

for $i = 1$ to n

  $tmp = A(:,i)$

  for $j = 1 : i-1$

2m flops $R(j,i) = Q(:,j)^T \cdot A(:,i)$ ... CGS

" $R(j,i) = Q(:,j)^T \cdot tmp$ ... MGS

" $tmp = tmp - R(j,i) \cdot Q(:,j)$

endfor

" $R(i,i) = \|tmp\|_2$

m flops $Q(:,i) = tmp / R(i,i)$

$\#\text{flops} = 2mn^2 + O(mn)$

$\sim 2\times$ NE if $m \gg n$

Householder – stable

MGS – less stable

CGS – even less stable

2 Metrics of Backward Stability

want accurate decomposition $A = QR$

$A + E = QR$, $\|E\| = O(\varepsilon)\cdot\|A\|$

$$\frac{\|A - QR\|}{\|A\|} = \frac{\|E\|}{\|A\|} = O(\varepsilon)$$

also want Q close to orthonormal

$$\|Q^TQ - I\| = O(\varepsilon)$$

MGS2 = MGS twice $\quad A = QR = (Q_1 R_1) R = Q_1 (R_1 R)$

CGS2 = CGS twice

Recall SVD: (U orthogonal)

$$A^{m\times n} = U^{m\times m} \cdot \Sigma \; V^{T\; n\times n}$$

$$= \begin{matrix} n & m-n \\ \left[U_1 \middle| U_2 \right] & \begin{matrix} \hat{\Sigma} \\ 0 \end{matrix} \end{matrix} V^T$$

$$\arg\min_x \|Ax-b\|_2 = V \hat{\Sigma}^{-1} U_1^T b$$

Moore Penrose Pseudoinverse

$$A^+ = V \hat{\Sigma}^{-1} U_1^T$$

$$= (A^T A)^{-1} A^T$$

Perturbation Theory for LS

How much can $x = \arg\min_x \|Ax-b\|$ change if A and b change?

When $m=n$, same as $x = A^{-1}b$

so expect $\kappa(A)$ to appear

Another source of ill-conditioning

$$A = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \; b = \begin{bmatrix} 0 \\ b_2 \end{bmatrix} \quad x = [1,0] \begin{bmatrix} 0 \\ b_2 \end{bmatrix} = 0$$

$$A = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \; b = \begin{bmatrix} b_1 \\ b_2 \end{bmatrix} \quad x = [1,0] \begin{bmatrix} b_1 \\ b_2 \end{bmatrix} = b_1$$

so large relative change in x

In general, ill-conditioned if

b (nearly) orthogonal to span(A)

$$x+e = \arg\min \|(A+\delta A)(x+e) - (b+\delta b)\|_2$$

vs $$x = \arg\min \|Ax - b\|_2$$

$$e = (x+e) - x = ((A+\delta A)^T(A+\delta A))^{-1}(A+\delta A)^T(b+\delta b) - (A^TA)^{-1}A^Tb$$

Do Taylor expansion, use

$$(I-X)^{-1} = I + X + O(\|X\|^2)$$

when $\|x\| \ll 1$

and only keep terms with one small factor δA, δb

Def: $\varepsilon = \max\left(\frac{\|\delta A\|}{\|A\|}, \frac{\|\delta b\|}{\|b\|}\right)$

$$\frac{\|e\|}{\|x\|} \leq \varepsilon\left(2\cdot K(A)\cdot\frac{1}{\cos\theta} + \tan\theta\, K^2(A)\right) + O(\varepsilon^2)$$

$\theta = \text{angle}(b, Ax)$

$$\sin\theta = \frac{\|Ax-b\|_2}{\|b\|_2}$$

$$\theta = 0 \Rightarrow \frac{\|e\|}{\|x\|} \leq \varepsilon\,(2\cdot K(A))$$

condition number for LS large if

(1) $K(A)$ large

(2) θ near $\pi/2$ $\Rightarrow$ $\frac{1}{\cos\theta} \sim \tan(\theta)$ large

(3) error like $K^2(A)$ when θ not near 0

Stable Algorithms for QR

MGS and CGS not stable

need Householder factorization

so $\|Q^TQ - I\| = O(\epsilon)$ and $\|A - QR\| = O(\epsilon) \cdot \|A\|$

Idea for stability: represent Q as a product $Q = Q_1 \cdot Q_2 \cdots Q_n$ of "simple" orthogonal matrices, chosen to make progress towards A turning into R ie. zeroing out entries of A. Since each Q_i orthogonal, so is Q

2 kinds of "Simple" Q_i's

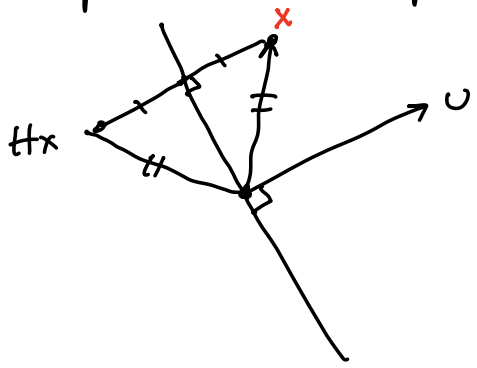
Householder Transformation (reflections)

Givens rotations

Householder reflection: $H = I - 2uu^T$ $\|u\|_2 = 1$

$$HH^T = (I - 2uu^T)(I - 2uu^T)$$
$$= I - 4uu^T + 4u\underbrace{u^Tu}_{1}u^T = I$$

Reflection: Hx is reflection of x in plane orthogonal to v



Given x we want to choose v so Hx has zeros in certain locations

$$Hx = \begin{bmatrix} c \\ 0 \\ \vdots \\ 0 \end{bmatrix} = c \cdot e_1 \quad \text{for some } c$$

$$\|Hx\|_2 = |c| = \|x\|_2 \Rightarrow c = \pm \|x\|_2$$

$$Hx = (I - 2vv^T)x = x - 2v(v^T x) = ce_1$$

$$v = \frac{x - ce_1}{2v^T x}$$

denominator scalar, choose it so $\|v\|_2 = 1$

$$y = x - ce_1 = x \pm \|x\|_2 e_1$$

$$v = y / \|y\|_2$$

$$v = \text{House}(x)$$

How to pick sign: choose sign to avoid cancellation which would cause numerical issues

$$y = \begin{bmatrix} x_1 + \text{sign}(x_1)\cdot\|x\|_2 \\ x_2 \\ x_3 \\ \vdots \\ x_n \end{bmatrix}$$

$$\begin{bmatrix} I^3 & 0 \\ 0 & H^{2\times 2} \end{bmatrix} \begin{bmatrix} I^2 & 0 \\ 0 & H^{3\times 3} \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & H^{4\times 4} \end{bmatrix} H^{5\times 5} \begin{bmatrix} x & x & x & x \\ x & x & x & x \\ x & x & x & x \\ x & x & x & x \\ x & x & x & x \end{bmatrix} = R$$

product of 4 orthogonal matrices $\Rightarrow$ orthogonal

A

QR decomposition of $A^{m\times n}$ $\quad m \geq n$

for $i = 1$ to $\min(m-1, n)$ ... only need to do last column if $m > n$

$$v(i) = \text{House}(A(i{:}m, i))$$

$$A(i{:}m, i{:}n) = \left(I - 2v(i)\,v(i)^T\right) A(i{:}m, i{:}n)$$

$$= A(i{:}m, i{:}n) - 2v(i)\cdot\left(v(i)^T A(i{:}m, i{:}n)\right)$$

vector-matrix multiply

rank-1 update

never form $H = I - 2uu^T$

or multiply them to get Q

Only need to store $u(i)$ vectors

∴ Same trick as GE: store $u(i)$ vectors below diagonal in "zeroed out" entries

A overwritten by Q (implicitly) and R

$$\text{Cost} = \sum_{i=1}^{n} 4(m-i+1)(n-i+1) = 2n^2m - \frac{2}{3}n^3$$

$m \gg n$: dominated by $2n^2m$

$m = n$: $\frac{4}{3}n^2$, twice GE

Implicit representation of Q:

$$Q_n \cdots Q_2 Q_1 A = R$$

$$A = Q_1^T Q_2^T \cdots Q_n^T R$$
$$= Q_1 Q_2 \cdots Q_n R$$
$$= QR$$

Solve LS problem: $x = R^{-1} Q^T b$

for $i = 1$ to n

$$b = Q_i b = (I - 2u(i)u(i)^T) b$$
$$= b + (-2u(i)^T b) \cdot u(i)$$

(dotproduct; saxpy)

$x = R^{-1} b$ by substitution

cost = $O(m \cdot n)$ much less than QR

Optimizing QR

So far: simple BLAS 2 version:

Same tricks as for LU and Cholesky

Goal: lower bound on

#words moved = $\Omega\left(\frac{\#\text{flops}}{\sqrt{\text{cache size}}}\right)$

(1) do QR on left part of matrix
(could be block of b columns,
or left half if recursive)

(2) update right part using
factorization of left part

(3) Do QR on right part

Need to do (2) using matmul

Need to multiply by $Q = Q_b \cdots Q_1$
using a few matmuls

Thm (see Q3.17 for details)
if $Q_i = I - 2 v_i v_i^T$ then

$Q_b \cdots Q_1 = Q = I - \underset{m \times b}{V} \; \underset{b \times b}{T} \; \underset{b \times m}{V^T}$

$Y = [v_1 \ldots v_b]$

T can be computed from v_i

One more case to optimize

"Tall skinny" A $m \gg n$, $n^2 \leq$ cache size $= M$

Lower bound $= \Theta\left(\frac{\#flops}{\sqrt{M}}\right) = \Theta\left(\frac{mn^2}{\sqrt{M}}\right)$

$\leq \Theta\left(\frac{mn^2}{n}\right) = \Theta(m \cdot n) = \Theta(\text{size of input})$

oops: can't move fewer words than size of input $= m \cdot n$

Different algorithm to hit lower bound Ω(input size), called TSQR "Tall Skinny" QR

suppose we can fit $\frac{1}{3}$ of A into cache

$$A = \begin{bmatrix} A_1 \\ A_2 \\ A_3 \end{bmatrix} \begin{matrix} m/3 \\ m/3 \\ m/3 \end{matrix}$$

... read A_1 into cache, do QR

$$= \begin{bmatrix} Q_1 R_1 \\ A_2 \\ A_3 \end{bmatrix} = \begin{bmatrix} Q_1 & & \\ & I & \\ & & I \end{bmatrix} \begin{bmatrix} R_1 \\ A_2 \\ A_3 \end{bmatrix} = \hat{Q}_1 \begin{bmatrix} R_1 \\ A_2 \\ A_3 \end{bmatrix}$$

... read A_2 into cache, do QR on $\begin{bmatrix} R_1 \\ A_2 \end{bmatrix}$

$= \hat{Q}_1 \begin{bmatrix} \begin{bmatrix} R_1 \\ A_2 \end{bmatrix} \\ A_3 \end{bmatrix} = \hat{Q}_1 \begin{bmatrix} Q_2 R_2 \\ A_3 \end{bmatrix} = \hat{Q}_1 \begin{bmatrix} Q_2 \\ I \end{bmatrix} \begin{bmatrix} R_2 \\ A_3 \end{bmatrix}$

$= \hat{Q}_1 \hat{Q}_2 \begin{bmatrix} R_2 \\ A_3 \end{bmatrix}$

... read A_3 into cache

do QR on $\begin{bmatrix} R_2 \\ A_3 \end{bmatrix}$

$= \hat{Q}_1 \hat{Q}_2 Q_3 R_3$ done!