# Welcome to Ma221! Lec 13, Fall 24

## Stability of applying orthogonal transformations

Summary: Any Algorithm that just multiplies by orthog. matrices is backward stable

$Q'$ "nearly orthogonal" $\|Q'-Q\| = O(\varepsilon)$

$Q^T Q = I$

(*) $fl(Q'\cdot A) = Q\cdot A + G \qquad \|G\| = O(\varepsilon)\|A\|$

$= Q(A + Q^T G) = Q(A+G')$

$\|G'\|_2 = \|G\|_2$

Multiplication by $Q'$ is backward stable = exact transform of $A+G'$

Eg: $Q' = I - 2uu^T \quad \|u\|_2 = 1$

in practice $|\|u\|_2 - 1| = O(\varepsilon)$

What if I multiply by many $Q_i'$?

$fl(Q_3'(Q_2'(Q_1'A)))$

$= fl(Q_3'(Q_2'(Q_1A+G_1)))$

$= fl(Q_3'(Q_2(Q_1A+G_1)+G_2))$

$$= fl\left(Q_3\left(Q_2\left(Q_1 A + G_1\right) + G_2\right) + G_3\right)$$

$$= Q_3 Q_2 Q_1 A + \underbrace{Q_3 G_2 + Q_3 \cdot Q_2 \cdot G_1 + G_3}_{G}$$

$$\|G\| \le \|G_2\| + \|G_1\| + \|G_3\| = O(\varepsilon) \cdot \|A\|$$

$$= Q_3 Q_2 Q_1 (A + \underbrace{(Q_3 Q_2 Q_1)^T G}_{G'})$$

$$\|G'\| = \|G\| = O(\varepsilon)\|A\|$$

$\Rightarrow$ Perturbation for LS applies to solving using $A = QR$ via Householder

One more fast but sometimes unstable alg for QR

Cholesky QR:

Factor $A^T A = LL^T = R^T R \quad (R = L^T)$

Form $Q = A R^{-1}$

Can fail if $A$ ill-conditioned but used if $A$ known to be well-conditioned

---

Dealing with (nearly) low rank matrices

Motivations: Real data often low rank (nearly redundant)

1. take precautions for inaccurate LS

② Use it to solve LS faster or compress data

Use LS as example of compression, other uses too

Solving a LS problem when A rank deficient

Thm: $A^{m\times n}$, $m \ge n$ rank $r < n$

$$A = U\Sigma V^T = \begin{array}{c} \\ m \end{array}\begin{array}{c} \begin{array}{ccc} r & n-r & m-n \end{array} \\ \left[U_1, U_2, U_3\right] \end{array} \begin{array}{c} \\ r \\ n-r \\ m-n \end{array}\begin{array}{c} \begin{array}{cc} r & n-r \end{array} \\ \begin{bmatrix} \Sigma_1 & 0 \\ 0 & \Sigma_2 \\ 0 & 0 \end{bmatrix} \end{array} \begin{array}{c} \\ n \end{array}\begin{array}{c} \begin{array}{cc} r & n-r \end{array} \\ \left[V_1, V_2\right]^T \end{array}$$

$\Sigma_1$: full rank

$\Sigma_2 = 0$ (later: tiny)

The set of vectors minimizing $\|Ax - b\|_2$ is

$$\{x = V_1 \Sigma_1^{-1} U_1^T b + \underbrace{V_2 y_2}_{\text{any vector in null\_space}(A)}, \text{ any } y_2 \in \mathbb{R}^{n-r}\}$$

Unique $x$ minimizing both $\|Ax-b\|_2$ and $\|x\|_2$ is gotten from $y_2 = 0$

$$\Rightarrow x = \underbrace{V_1 \Sigma_1^{-1} U_1^T}\, b$$

Def: $A^+ = V_1 \Sigma_1^{-1} U_1^T$ is Moore-Penrose pseudoinverse of $A$ (includes full rank case $r=n$)

in practice, $\Sigma_2$ contains all singular values $<$ some user defined threshold

Proof: $\|Ax-b\|_2 = \|U\Sigma V^T x - b\|_2$

$= \|\Sigma V^T x - U^T b\|_2$ since $UU^T = I$

$= \|\Sigma y - U^T b\|_2$ where $y = V^T x$

$\|x\|_2 = \|y\|_2$ so ok to minimize either one

$$= \left\| \begin{bmatrix} \Sigma_1 y_1 - U_1^T b \\ -U_2^T b \\ -U_3^T b \end{bmatrix} \right\|_2 \quad \text{where } y = \begin{array}{c} \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} \end{array} \begin{array}{l} r \\ n-r \end{array}$$

minimized by $y_1 = \Sigma_1^{-1} U_1^T b$

$\|y_1\|_2^2 + \|y_2\|_2^2 = \|y\|_2^2$ minimized by $y_2 = 0$

$$x = Vy = [V_1, V_2]\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = V_1 y_1 + V_2 y_2$$

$$= V_1 \Sigma_1^{-1} U_1^T b \qquad \text{QED}$$

---

Solving LS when $A$ nearly rank deficient using truncated SVD

$A$ sing $\Rightarrow \kappa(A) = \dfrac{\sigma_{max}}{\sigma_{min}} = \infty$

$$\underset{x}{\operatorname{argmin}} \left\| \begin{bmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} - \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} \right\|_2 = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

$\|x\|_2$ min.

$0 < e \ll 1$ $\quad \operatorname{argmin}_x \left\| \begin{bmatrix} 1 & 0 \\ 0 & e \\ 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} - \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} \right\|_2 = \begin{bmatrix} 1 \\ 1/e \end{bmatrix}$

what does a solution mean if it can change discontinuously?

Often $A$ not known exactly just up to some tolerance $\|A - A'\|_2 < tol$, What to do?

Def: Truncated SVD

if $A = U \Sigma V^T$

$$A(tol) = U\,\Sigma(tol)\,V^T$$

$$\Sigma(tol) = \operatorname{diag}(\sigma_1(tol), \sigma_2(tol), \dots \sigma_n(tol))$$

$$\sigma_i(tol) = \begin{cases} \sigma_i & \text{if } \sigma_i \geq tol \\ 0 & \text{if } \sigma_i < tol \end{cases}$$

$A(tol)$ = lowest rank matrix within distance tol of $A$

Solve LS using $A(tol)$: reduces condition number from $\kappa(A) = \dfrac{\sigma_{max}}{\sigma_{min}}$

to $\kappa(A_{TOL}) = \dfrac{\sigma_{max}}{tol}$

Replacing $A$ by "easier problem" less sensitive, but larger residual; process called regularization, several mechanisms

Lemma: $x_1 = \underset{x}{\operatorname{argmin}} \| A(tol)x - b_1 \|_2$

$x_2 = \underset{x}{\operatorname{argmin}} \| A(tol)x - b_2 \|_2$

Then $\|x_1 - x_2\|_2 \le \dfrac{\|b_1 - b_2\|_2}{tol}$

proof: $\|x_1 - x_2\|_2 = \| (A(tol))^+ (b_1 - b_2) \|$

$= \| V (\Sigma(tol))^+ U^T (b_1 - b_2) \|_2$

$= \| (\Sigma(tol))^+ U^T (b_1 - b_2) \|_2$

$\le \| (\Sigma(tol))^+ \|_2 \cdot \| b_1 - b_2 \|_2$

$\le \dfrac{1}{\sigma_k} \| b_1 - b_2 \|_2$ $\quad$ $\sigma_k$ smallest $\ge tol$)

$\le \dfrac{1}{tol} \| b_1 - b_2 \|_2$

How does $A(tol)$ depend on $tol$?

piecewise constant, changes when $tol = \sigma_k$

Other advantage of $A(tol)$: compressible

only need $\underset{U_1}{m \cdot k} + \underset{\Sigma(tol)}{k} + \underset{V_1}{n \cdot k}$ words

instead of $m \cdot n$

can be << SVD

Solving (nearly) low rank LS using Tikhonov regularization or ridge regression

Replace $\underset{x}{\operatorname{argmin}} \|Ax-b\|_2^2$

with $\underset{x}{\operatorname{argmin}} \|Ax-b\|_2^2 + \lambda \|x\|_2^2$ , $\lambda>0$

$\lambda$ "penalizes" very large $x$, user parameter

$$\underset{x}{\operatorname{argmin}} \|Ax-b\|_2^2 + \lambda\|x\|_2^2$$

$$= \underset{x}{\operatorname{argmin}} \Big\| \underbrace{\begin{bmatrix} A \\ \lambda^{\frac{1}{2}} I \end{bmatrix}}_{\text{full rank iff } \lambda>0} x - \begin{bmatrix} b \\ 0 \end{bmatrix} \Big\|_2^2$$

$$NE \Rightarrow x = \left( \begin{bmatrix} A \\ \sqrt{\lambda} I \end{bmatrix}^T \begin{bmatrix} A \\ \sqrt{\lambda} I \end{bmatrix} \right)^{-1} \begin{bmatrix} A \\ \sqrt{\lambda} I \end{bmatrix}^T \begin{bmatrix} b \\ 0 \end{bmatrix}$$

$$(*) \qquad = \underbrace{(A^T A + \lambda I)}_{\text{pos def } \forall \lambda>0}^{-1} A^T b$$

How does $\lambda$ change SVD

plug $A = U\Sigma V^T$ into $(*)$

$$x = V(\Sigma(\Sigma^2 + \lambda I)^{-1}) U^T b$$

$$= V \cdot \operatorname{diag}\left( \frac{\sigma_i}{\sigma_i^2 + \lambda} \right) U^T b$$

if $\lambda = 0 \Rightarrow$ usual answer via SVD

$\sigma_i \gg \lambda^{1/2} \Rightarrow \dfrac{\sigma_i}{\sigma_i^2+\lambda} \sim \dfrac{1}{\sigma_i}$ , like SVD

$\sigma_i \le \lambda^{1/2} \Rightarrow \dfrac{\sigma_i}{\sigma_i^2+\lambda} \le \dfrac{1}{2\lambda^{1/2}}$

i.e. $\lambda^{1/2}$ and tol in A(tol)
play similar roles, but
solution of ridge regression is
continuous in $\lambda$

---

Solving low rank LS using QR

QR with column pivoting

Suppose we did $A = QR$ exactly on
$A$: rank$(A) = r < n$, what would $R$ look like?

If leading $r$ columns of $A$ were
linearly independent (true for "almost all"
low rank $A$)

$$R = \begin{array}{c|c} & \begin{array}{cc} r & n-r \end{array} \\ \begin{array}{c} r \\ n-r \end{array} & \begin{bmatrix} R_{11} & R_{12} \\ 0 & 0 \end{bmatrix} \end{array}$$

$R_{11}$ full rank
$R_{22} = 0$

If $A$ nearly low rank, hope that
$\|R_{22}\| < \text{tol}$, set $R_{22} = 0$

Assuming this works, solve LS:

$$A = {}_m\hat{Q}_n \hat{R} = m\begin{bmatrix} \overset{n}{\hat{Q}}, \overset{m-n}{Q'} \end{bmatrix} \begin{bmatrix} R \\ 0 \end{bmatrix}$$

$$m\begin{bmatrix} \overset{n}{\hat{Q}}, \overset{m-n}{Q'} \end{bmatrix} = \begin{bmatrix} \overset{r}{\hat{Q}_1}, \overset{n-r}{Q_2}, \overset{m-n}{Q'} \end{bmatrix} m$$

$$\underset{x}{\operatorname{argmin}} \|Ax-b\|_2$$

$$= \underset{x}{\operatorname{argmin}} \left\| [Q_1, Q_2, Q'] \begin{bmatrix} R \\ 0 \end{bmatrix} x - b \right\|_2$$

$$= \underset{x}{\operatorname{argmin}} \left\| \begin{array}{c|cc} & r & n-r \\ \hline r & R_{11} & R_{12} \\ n-r & 0 & 0 \\ m-n & 0 & 0 \end{array} \; \begin{array}{c|c} r & x_1 \\ n-r & x_2 \end{array} - \begin{bmatrix} Q_1^T b \\ Q_2^T b \\ Q'^T b \end{bmatrix} \right\|_2$$

$$= \underset{x}{\operatorname{argmin}} \left\| \begin{bmatrix} R_{11} x_1 + R_{12} x_2 - Q_1^T b \\ - Q_2^T b \\ - Q'^T b \end{bmatrix} \right\|_2$$

solution: $x_1 = R_{11}^{-1} Q_1^T b - R_{11}^{-1} R_{12} x_2$
for any $x_2$

How to pick $x_2$ to minimize $\|x\|_2$?

Ex: $A = \begin{bmatrix} e & 1 \\ 0 & 0 \end{bmatrix}$ $0 < e \ll 1$ $R_{11} = e$, $R_{12} = 1$

$x = \begin{bmatrix} (b_1 - x_2)/e \\ x_2 \end{bmatrix}$ $e$ tiny $\Rightarrow$ very sensitive to changes in $b_1, x_2$

permute columns of $A$:

$AP = \begin{bmatrix} 1 & e \\ 0 & 0 \end{bmatrix} \Rightarrow x = \begin{bmatrix} b_1 - e x_2 \\ x_2 \end{bmatrix}$

insensitive to changes in $x_2$, $b_1$

What would a perfect R factor look like?

Compare $\begin{array}{cc} & \begin{array}{cc} k & \end{array} \\ k & \begin{bmatrix} R_{11} & R_{12} \\ 0 & R_{22} \end{bmatrix} \end{array}$ to $\begin{array}{cc} & \begin{array}{cc} k & \end{array} \\ k & \begin{bmatrix} \Sigma_1 & 0 \\ 0 & \Sigma_2 \end{bmatrix} \end{array}$

Def: Rank Revealing QR factorization (RRQR for short) is $A \cdot P = QR$

P permutation chosen so that

(1) $R_{22}$ is "small", ideally $\|R_{22}\|_2 = O(\sigma_{k+1})$

$R_{22}$ "contains smallest n-k sing. values"

(2) $R_{11}$ is "large", ideally $\sigma_{min}(R_{11})$ not much smaller than $\sigma_k$

If in addition

(3) $\|R_{11}^{-1} R_{12}\|_2$ not "too large"

then $AP = QR$ called "strong RRQR"

Thm (informal) if (1), (2), (3) hold

$$\sigma_i(A) \geq \sigma_i(R_{11}) \geq \frac{\sigma_i(A)}{\sqrt{1 + \|R_{11}^{-1} R_{12}\|_2^2}}$$

for $1 \leq i \leq k$

$$\sigma_i(A) \leq \sigma_{max}(R_{22}) \sqrt{1 + \|R_{11}^{-1} R_{12}\|_2^2}$$

$i = k+1, \dots, n$

Leading $k$ columns of $A \cdot P$ contain most information about range$(A)$

$$AP = \left[\overset{k}{A_1}, \overset{n-k}{A_2}\right] = QR = \left[\overset{k}{Q_1}, \overset{n-k}{Q_2}\right] \begin{array}{c} k \\ n-k \end{array} \left[\begin{array}{cc} \overset{k}{R_{11}} & \overset{n-k}{R_{12}} \\ 0 & R_{22} \end{array}\right]$$

$$\approx [Q_1, Q_2] \cdot \begin{bmatrix} R_{11} & R_{12} \\ 0 & 0 \end{bmatrix}$$

$$= Q_1 [R_{11} \quad R_{12}]$$

$$= Q_1 R_{11} [I, R_{11}^{-1} R_{12}]$$

$$= A_1 [I, R_{11}^{-1} R_{12}]$$

How to compute $P$?

(matlab demo, see notes)

Algorithm: QR with column pivoting
QRCP for short

Analogous to partial pivoting
often works, like GEPP, can be off by a factor of $2^n$

first step: pick largest column of $A$, move to front take one step of QR

among remaining columns, pick one with largest component orthogonal to first column

repeat

for $i = 1$ to $\min(m-1, n)$ or $R_{22}$ small enough

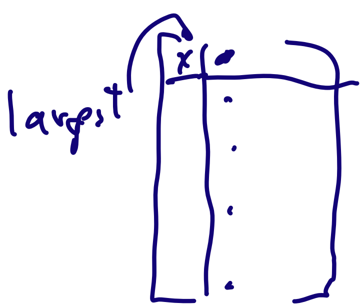
"truncated QR"

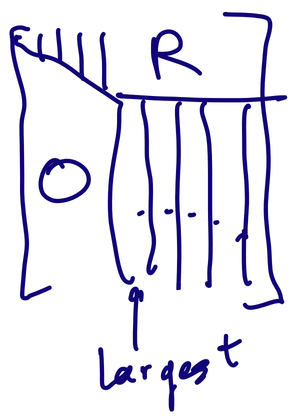
choose largest column (in norm) in trailing matrix

$$j = \operatorname*{argmax}_{j \geq i} \| A(i:m, j) \|_2$$

if $i \neq j$, swap cols $i$ and $j$

multiply $A(i:m, i:n)$ by Householder matrix to zero out $A(i+1:m, i)$




Need to compute column norms carefully to be fast + accurate in LAPACK: geqp3, geqp3rk

Matlab: [Q,R,P] = qr(A)