QR Factorization and

## Singular Value Decomposition

$\cos 323$

## Last time

- Solving non-linear least squares
- Newton, Gauss-Newton, Levenberg-Marquardt methods
- Intro to logistic regresion
- Dealing with outliers and bad data:
- Robust regression, least absolute deviation, and iteratively re-weighted least-squares
- Practical considerations
- Solving with Excel and Matlab


## Today

- How do we solve least-squares...
- without incurring condition-squaring effect of normal equations ( $A^{\top} A x=A^{\top} b$ )
- when A is singular, "fat", or otherwise poorly-specified?
- QR Factorization
- Householder method
- Singular Value Decomposition
- Total least squares
- Practical notes


## Review: Condition Number

- Cond(A) is function of $A$
- $\operatorname{Cond}(A)>=1$, bigger is bad
- Measures how change in input is propogated to change in output

$$
\frac{\|\Delta x\|}{\|x\|} \leq \operatorname{cond}(A) \frac{\|\Delta A\|}{\|A\|}
$$

- E.g., if cond $(A)=451$ then can lose log(451)= 2.65 digits of accuracy in $x$, compared to precision of $A$


## Normal Equations are Bad

$$
\frac{\|\Delta x\|}{\|x\|} \leq \operatorname{cond}(A) \frac{\|\Delta A\|}{\|A\|}
$$

- Normal equations involves solving $\mathrm{A}^{\top} \mathrm{Ax}=\mathrm{A}^{\top} \mathrm{b}$
- $\operatorname{cond}\left(\mathrm{A}^{\top} \mathrm{A}\right)=[\operatorname{cond}(\mathrm{A})]^{2}$
- E.g., if $\operatorname{cond}(A)=451$ then can $\operatorname{lose} \log \left(451^{2}\right)=5.3$ digits of accuracy, compared to precision of $A$


## QR Decomposition

## What if we didn't have to use $A^{T} A$ ?

- Suppose we are "lucky":

$$
\left[\begin{array}{cccc}
\# & \# & \cdots & \# \\
0 & \# & & \# \\
0 & 0 & \ddots & \vdots \\
0 & \cdots & 0 & \# \\
0 & \cdots & \cdots & 0 \\
\vdots & & & \vdots \\
0 & 0 & \cdots & 0
\end{array}\right] x \cong\left[\begin{array}{l}
\# \\
\# \\
\# \\
\# \\
\# \\
\# \\
\#
\end{array}\right] \quad\left[\begin{array}{l}
R \\
O
\end{array}\right] x=b
$$

- Upper triangular matrices are nice!


## How to make A upper-triangular?

- Gaussian elimination?
- Applying elimination yields MAx = Mb
- Want to find x s.t. minimizes $\|\mathrm{Mb}-\mathrm{MAx}\|_{2}$
- Problem: $\|\mathrm{Mv}\|_{2}$ != $\|\mathrm{v}\|_{2}$ (i.e., M might "stretch" a vector v)
- Another problem: M may stretch different vectors differently
- i.e., M does not preserve Euclidean norm
- i.e., $x$ that minimizes ||Mb-MAx|| may not be same $x$ that minimizes $A x=b$


## QR Factorization

- Can't usually find R such $A=\left[\begin{array}{l}R \\ O\end{array}\right]$
- Can find Q , R such that $A=Q\left[\begin{array}{l}R \\ O\end{array}\right]$, so $\left[\begin{array}{l}R \\ O\end{array}\right] x=Q^{T} b$
- If Q orthogonal, doesn't change leastsquares solution
$-Q^{\top} Q=I$, columns of $Q$ are orthonormal
- i.e., $Q$ preserves Euclidean norm: $\|Q v\|_{2}=\|v\|_{2}$


## Goal of QR



## Reformulating Least Squares using QR

$$
\begin{aligned}
\|r\|_{2}^{2} & =\|b-A x\|_{2}^{2} \\
& \left.=\left\|b-Q\left[\begin{array}{l}
R \\
O
\end{array}\right] x\right\|_{2}^{2}=\| Q^{T} b-Q^{T} Q\left[\begin{array}{l}
R \\
O
\end{array}\right]\right]^{2} \|_{2}^{2} \quad \text { because } A=Q\left[\begin{array}{l}
R \\
O
\end{array}\right] \\
& \left.=\left\|Q^{T} b-\left[\begin{array}{l}
R \\
O
\end{array}\right] x\right\|_{2}^{2} \quad \text { because } \mathrm{Q} \text { is orthogonal (QTQ}=1\right) \\
& =\left\|c_{1}-R x+c_{2}\right\|_{2}^{2} \text { if we call } Q^{T} b=\left[\begin{array}{l}
c_{1} \\
c_{2}
\end{array}\right] \\
& =\left\|c_{1}-R x\right\|_{2}^{2}+\left\|c_{2}\right\|_{2}^{2} \\
& =\left\|c_{2}\right\|_{2}^{2} \text { if we choose } \mathrm{x} \text { such that } \mathrm{R} \mathbf{x}=\mathrm{c}_{1}
\end{aligned}
$$

# Householder Method for Computing QR Decomposition 

## Orthogonalization for Factorization

$$
A=Q\left[\begin{array}{l}
R \\
O
\end{array}\right]
$$

- Rough idea:
- For each i-th column of A, "zero out" rows i+1 and lower
- Accomplish this by multiplying A with an orthogonal matrix $\mathrm{H}_{\mathrm{i}}$
- Equivalently, apply an orthogonal transformation to the i-th column (e.g., rotation, reflection)
$-Q$ becomes product $H_{1}{ }^{*} \ldots{ }^{*} H_{n}, R$ contains zero-ed out columns


## Householder Transformation

- Accomplishes the critical sub-step of factorization:
- Given any vector (e.g., a column of A), reflect it so that its last $p$ elements become 0 .
- Reflection preserves length (Euclidean norm)



## Computing Householder

- if $a$ is the $k$-th column:

$$
\begin{aligned}
& a=\left[\begin{array}{l}
a_{1} \\
a_{2}
\end{array}\right] \\
& v=\left[\begin{array}{l}
0 \\
a_{2}
\end{array}\right]-\alpha e_{k} \text { where } \alpha=-\operatorname{sign}\left(a_{k}\right)\left\|a_{2}\right\|_{2}
\end{aligned}
$$

apply H to a and columns to the right :

$$
H u=u-\left(2 \frac{v^{T} u}{v^{T} v}\right) v(* \text { with some shortcuts - see } \mathrm{p} 124)
$$

Exercise: Show H is orthogonal $\left(\mathrm{H}^{\top} \mathrm{H}=\mathrm{I}\right)$

## Outcome of Householder

$$
\begin{aligned}
& H_{n} \ldots H_{1} A=\left[\begin{array}{l}
R \\
O
\end{array}\right] \\
& \text { where } Q^{T}=H_{n} \ldots H_{1} \\
& \text { so } Q=H_{1} \ldots H_{n} \\
& \text { so } \mathrm{A}=\mathrm{Q}\left[\begin{array}{l}
R \\
O
\end{array}\right]
\end{aligned}
$$

## Review: Least Squares using QR

$$
\begin{aligned}
\|r\|_{2}^{2} & =\|b-A x\|_{2}^{2} \\
& =\left\|b-Q\left[\begin{array}{l}
R \\
O
\end{array}\right]\right\|^{2}\left\|_{2}^{2}=\right\| Q^{T} b-Q^{T} Q\left[\begin{array}{l}
R \\
O
\end{array}\right] \|_{2}^{2} \quad \text { because } A=Q\left[\begin{array}{l}
R \\
O
\end{array}\right] \\
& \left.=\left\|Q^{T} b-\left[\begin{array}{l}
R \\
O
\end{array}\right] x\right\|_{2}^{2} \quad \text { because } \mathrm{Q} \text { is orthogonal (QTQ}=1\right) \\
& =\left\|c_{1}-R x+c_{2}\right\|_{2}^{2} \text { if we call } Q^{T} b=\left[\begin{array}{l}
c_{1} \\
c_{2}
\end{array}\right] \\
& =\left\|c_{1}-R x\right\|_{2}^{2}+\left\|c_{2}\right\|_{2}^{2} \\
& =\left\|c_{2}\right\|_{2}^{2} \text { if we choose } \mathrm{x} \text { such that } \mathrm{R} \mathbf{x}=\mathrm{c}_{1}
\end{aligned}
$$

## Using Householder

- Iteratively compute $\mathrm{H}_{1}, \mathrm{H}_{2}, \ldots \mathrm{H}_{\mathrm{n}}$ and apply to A to get R
- also apply to b to get

$$
Q^{T} b=\left[\begin{array}{l}
c_{1} \\
c_{2}
\end{array}\right]
$$

- Solve for $\mathrm{Rx}=\mathrm{c}_{1}$ using back-substitution


## Alternative Orthogonalization Methods

- Givens:
- Don't reflect; rotate instead
- Introduces zeroes into A one at a time
- More complicated implementation than Householder
- Useful when matrix is sparse
- Gram-Schmidt
- Iteratively express each new column vector as a linear combination of previous columns, plus some (normalized) orthogonal component
- Conceptually nice, but suffers from subtractive cancellation


## Singular Value Decomposition



## Motivation \#1

- Diagonal matrices are even nicer than triangular ones:

$$
\left[\begin{array}{cccc}
\# & 0 & 0 & 0 \\
0 & \# & 0 & 0 \\
0 & 0 & \ddots & 0 \\
0 & \cdots & 0 & \# \\
0 & \cdots & \cdots & 0 \\
\vdots & & & \vdots \\
0 & 0 & \cdots & 0
\end{array}\right] x \cong\left[\begin{array}{c}
\# \\
\# \\
\# \\
\# \\
\# \\
\# \\
\#
\end{array}\right]
$$

## Motivation \#2

- What if you have fewer data points than parameters in your function?
- i.e., A is "fat"
- Intuitively, can't do standard least squares
- Recall that solution takes the form $A^{\top} A x=A^{\top} b$
- When $A$ has more columns than rows, $A^{\top} A$ is singular: can't take its inverse, etc.


## Motivation \#3

- What if your data poorly constrains the function?
- Example: fitting to $y=a x^{2}+b x+c$


## Underconstrained Least Squares

- Problem: if problem very close to singular, roundoff error can have a huge effect
- Even on "well-determined" values!
- Can detect this:
- Uncertainty proportional to covariance $C=\left(A^{\top} A\right)^{-1}$
- In other words, unstable if $A^{\top} A$ has small values
- More precisely, care if $x^{\top}\left(A^{\top} A\right) x$ is small for any $x$
- Idea: if part of solution unstable, set answer to 0
- Avoid corrupting good parts of answer


## Singular Value Decomposition (SVD)

- Handy mathematical technique that has application to many problems
- Given any $m \times n$ matrix A, algorithm to find matrices $\mathbf{U}, \mathbf{V}$, and $\mathbf{W}$ such that
$\mathbf{A}=\mathbf{U} \mathbf{W} \mathbf{V}^{\top}$
$\mathbf{U}$ is $m \times n$ and orthonormal
$\mathbf{W}$ is $n \times n$ and diagonal
$\mathbf{V}$ is $n \times n$ and orthonormal


## SVD



- Treat as black box: code widely available In Matlab: $[\mathrm{U}, \mathrm{w}, \mathrm{v}]=\operatorname{svd}(\mathrm{A}, \mathrm{O})$


## SVD

- The $w_{i}$ are called the singular values of $\mathbf{A}$
- If $\mathbf{A}$ is singular, some of the $w_{i}$ will be 0
- In general $\operatorname{rank}(\mathbf{A})=$ number of nonzero $w_{i}$
- SVD is mostly unique (up to permutation of singular values, or if some $w_{i}$ are equal)


## SVD and Inverses

- Why is SVD so useful?
- Application \#1: inverses
- $\mathbf{A}^{-1}=\left(\mathbf{V}^{\top}\right)^{-1} \mathbf{W}^{-1} \mathbf{U}^{-1}=\mathbf{V} \mathbf{W}^{-1} \mathbf{U}^{\top}$
- Using fact that inverse = transpose for orthogonal matrices
- Since $\mathbf{W}$ is diagonal, $\mathbf{W}^{-1}$ also diagonal with reciprocals of entries of $\mathbf{W}$


## SVD and the Pseudoinverse

- $\mathbf{A}^{-1}=\left(\mathbf{V}^{\top}\right)^{-1} \mathbf{W}^{-1} \mathbf{U}^{-1}=\mathbf{V} \mathbf{W}^{-1} \mathbf{U}^{\top}$
- This fails when some $w_{i}$ are 0
- It's supposed to fail - singular matrix
- Happens when rectangular $A$ is rank deficient
- Pseudoinverse: if $w_{i}=0$, set $1 / w_{i}$ to 0 (!)
- "Closest" matrix to inverse
- Defined for all (even non-square, singular, etc.) matrices
- Equal to ( $\left.\mathbf{A}^{\top} \mathbf{A}\right)^{-1} \mathbf{A}^{\top}$ if $\mathbf{A}^{\top} \mathbf{A}$ invertible


## SVD and Condition Number

- Singular values used to compute Euclidean (spectral) norm for a matrix:

$$
\operatorname{cond}(A)=\frac{\sigma_{\max }(A)}{\sigma_{\min }(A)}
$$

## SVD and Least Squares

- Solving $\mathbf{A x}=\mathbf{b}$ by least squares:
- $A^{\top} A x=A^{\top} b \rightarrow x=\left(A^{\top} A\right)^{-1} A^{\top} b$
- Replace with $\mathrm{A}^{+}: ~ x=A^{+} b$
- Compute pseudoinverse using SVD
- Lets you see if data is singular (<n nonzero singular values)
- Even if not singular, condition number tells you how stable the solution will be
- Set $1 / w_{i}$ to 0 if $w_{i}$ is small (even if not exactly 0 )


## SVD and Matrix Similarity

- One common definition for the norm of a matrix is the Frobenius norm:

$$
\|\mathbf{A}\|_{\mathrm{F}}=\sum_{i} \sum_{j} a_{i j}^{2}
$$

- Frobenius norm can be computed from SVD

$$
\|\mathbf{A}\|_{\mathrm{F}}=\sum_{i} w_{i}^{2}
$$

- Euclidean (spectral) norm can also be computed:

$$
\|\mathbf{A}\|_{2}=\{\max |\lambda|: \lambda \in \sigma(\mathbf{A})\}
$$

- So changes to a matrix can be evaluated by looking at changes to singular values


## SVD and Matrix Similarity

- Suppose you want to find best rank-k approximation to $\mathbf{A}$
- Answer: set all but the largest $k$ singular values to zero
- Can form compact representation by eliminating columns of $\mathbf{U}$ and $\mathbf{V}$ corresponding to zeroed $w_{i}$


## SVD and Eigenvectors

- Let $\mathbf{A}=\mathbf{U W V}^{\top}$, and let $x_{i}$ be $i^{\text {th }}$ column of $\mathbf{V}$

$$
\begin{gathered}
\text { - Consider } \mathbf{A}^{\mathrm{T}} \mathbf{A} x_{i}: \\
\mathbf{A}^{\mathrm{T}} \mathbf{A} x_{i}=\mathbf{V W}^{\mathrm{T}} \mathbf{U}^{\mathrm{T}} \mathbf{U W} \mathbf{V}^{\mathrm{T}} x_{i}=\mathbf{V} \mathbf{W}^{2} \mathbf{V}^{\mathrm{T}} x_{i}=\mathbf{V} \mathbf{w}^{2}\left(\begin{array}{c}
0 \\
\vdots \\
1 \\
\vdots \\
0
\end{array}\right)=\mathbf{V}\left(\begin{array}{c}
0 \\
\vdots \\
w_{i}^{2} \\
\vdots \\
0
\end{array}\right)=w_{i}^{2} x_{i} .
\end{gathered}
$$

- So elements of $\mathbf{W}$ are sqrt(eigenvalues) and columns of $\mathbf{V}$ are eigenvectors of $\mathbf{A}^{\top} \mathbf{A}$


## Total Least Squares

- One final least squares application
- Fitting a line: vertical vs. perpendicular error


## Total Least Squares

- Distance from point to line:

$$
d_{i}=\binom{x_{i}}{y_{i}} \cdot \vec{n}-a
$$

where n is normal vector to line, a is a constant

- Minimize:

$$
\chi^{2}=\sum_{i} d_{i}^{2}=\sum_{i}\left[\binom{x_{i}}{y_{i}} \cdot \vec{n}-a\right]^{2}
$$

## Total Least Squares

- First, let's pretend we know n, solve for a

$$
\begin{aligned}
& \chi^{2}=\sum_{i}\left[\binom{x_{i}}{y_{i}} \cdot \vec{n}-a\right]^{2} \\
& a=\frac{1}{m} \sum_{i}\binom{x_{i}}{y_{i}} \cdot \vec{n}
\end{aligned}
$$

- Then

$$
d_{i}=\binom{x_{i}}{y_{i}} \cdot \vec{n}-a=\binom{x_{i}-\frac{\Sigma x_{i}}{m}}{y_{i}-\frac{\Sigma y_{i}}{m}} \cdot \vec{n}
$$

## Total Least Squares

- So, let's define

$$
\binom{\widetilde{x}_{i}}{\tilde{y}_{i}}=\binom{x_{i}-\frac{\Sigma x_{i}}{m}}{y_{i}-\frac{\Sigma v_{i}}{m}}
$$

and minimize

## Total Least Squares

- Write as linear system

$$
\left(\begin{array}{cc}
\widetilde{x}_{1} & \widetilde{y}_{1} \\
\widetilde{x}_{2} & \widetilde{y}_{2} \\
\widetilde{x}_{3} & \widetilde{y}_{3}
\end{array}\right)\left(\begin{array}{l}
n_{x} \\
\\
\\
n_{y}
\end{array}\right)=\overrightarrow{0}
$$

- Have An=0
- Problem: lots of $n$ are solutions, including $n=0$
- Standard least squares will, in fact, return $n=0$


## Constrained Optimization

- Solution: constrain n to be unit length
- So, try to minimize $|A n|^{2}$ subject to $|n|^{2}=1$

$$
\|\mathbf{A} \vec{n}\|^{2}=(\mathbf{A} \vec{n})^{\mathrm{T}}(\mathbf{A} \vec{n})=\vec{n}^{\mathrm{T}} \mathbf{A}^{\mathrm{T}} \mathbf{A} \vec{n}
$$

- Expand in eigenvectors $e_{i}$ of $A^{\top} A$ :

$$
\begin{gathered}
\vec{n}=\mu_{1} \mathbf{e}_{1}+\mu_{2} \mathbf{e}_{2} \\
\vec{n}^{\mathrm{T}}\left(\mathbf{A}^{\mathrm{T}} \mathbf{A}\right) \hat{n}=\lambda_{1} \mu_{1}^{2}+\lambda_{2} \mu_{2}^{2} \\
\|\vec{n}\|^{2}=\mu_{1}^{2}+\mu_{2}^{2}
\end{gathered}
$$

where the $\lambda_{i}$ are eigenvalues of $\mathrm{A}^{\top} \mathrm{A}$

## Constrained Optimization

- To minimize $\lambda_{1} \mu_{1}^{2}+\lambda_{2} \mu_{2}^{2}$ subject to $\mu_{1}^{2}+\mu_{2}^{2}=1$ set $\mu_{\text {min }}=1$, all other $\mu_{\mathrm{i}}=0$
- That is, $n$ is eigenvector of $A^{\top} A$ with the smallest corresponding eigenvalue


## Comparison of Least Squares Methods

- Normal equations ( $\mathrm{A}^{\top} \mathrm{Ax}$
$=A^{\top} b$ )
- $O\left(m n^{2}\right)$ (using Cholesky)
$-\operatorname{cond}\left(\mathrm{A}^{\top} \mathrm{A}\right)=[\operatorname{cond}(\mathrm{A})]^{2}$
- Cholesky fails if cond(A)~1/sqrt(machine epsilon)
- Householder
- Usually best orthogonalization method
$-\mathrm{O}\left(\mathrm{mn}^{2}-\mathrm{n}^{3} / 3\right)$ operations
- Relative error is best possible for least squares
- Breaks if cond(A) ~ 1/ (machine eps)
- SVD
- Expensive: $m n^{2}+n^{3}$ with bad constant factor
- Can handle rankdeficiency, near-singularity
- Handy for many different things


## Matlab functions

- qr: explicit QR factorization
- svd
- Alb: ('\’ operator)
- Performs least-squares if $A$ is m-by-n
- Uses QR decomposition
- pinv: pseudoinverse
- rank: Uses SVD to compute rank of a matrix

