VE472 Lecture 4

Jing Liu

UM-SJTU Joint Institute

Summer

Theorem 0.1 (Singular Value Decomposition)

Let $\mathbf{A}$ *be a rank* k *matrix of* $m \times n$ *with* $m \geq n$*, then we have* $\sigma_1 \geq \cdots \geq \sigma_k > 0$

$$\mathbf{A} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^{\mathrm{T}}$$

$$= \left[\begin{array}{ccc|ccc} \mathbf{u}_1 & \cdots & \mathbf{u}_k & \mathbf{u}_{k+1} & \cdots & \mathbf{u}_m \end{array}\right] \left[\begin{array}{ccc|c} \sigma_1 & \cdots & 0 & \\ \vdots & \ddots & \vdots & \mathbf{0}_{k\times(n-k)} \\ 0 & \cdots & \sigma_k & \\ \hline & \mathbf{0}_{(m-k)\times k} & & \mathbf{0}_{(m-k)\times(n-k)} \end{array}\right] \left[\begin{array}{c} \mathbf{v}_1^{\mathrm{T}} \\ \vdots \\ \mathbf{v}_k^{\mathrm{T}} \\ \hline \mathbf{v}_{k+1}^{\mathrm{T}} \\ \vdots \\ \mathbf{v}_n^{\mathrm{T}} \end{array}\right]$$

where $\mathbf{U}$ *and* $\mathbf{V}$ *are orthogonal matrices of size* $m \times m$ *and* $n \times n$*, respectively.*

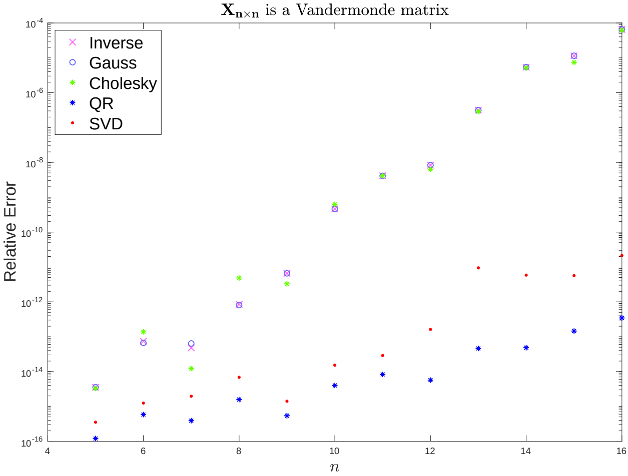
Q: Why is this theorem useful in terms of dealing with a big data matrix $\mathbf{X}$?

- SVD is faster than QR, but a magnitude slower than LU or Cholesky .

```
>> n = 1000; X = rand(n, 100); y = randn(n, 1);
tic;for i = 1:10000
    XX = transpose(X)*X; yy = transpose(X)*y;
    C = chol(XX, 'lower');
    bhat = transpose(C)\(C\yy);
end; toc;
tic;for i = 1:10000
    [Q, R] = qr(X); bhat = R\(transpose(Q)*y);
end; toc;
tic;for i = 1:10000
    [U, S, V] = svd(X, 'econ');
    s = diag(S); s = 1./s;
    bhat = V*diag(s)*transpose(U)*y;
end; toc;
```

```
Elapsed time is 2.521359 seconds.
Elapsed time is 45.891408 seconds.
Elapsed time is 19.551785 seconds.
```

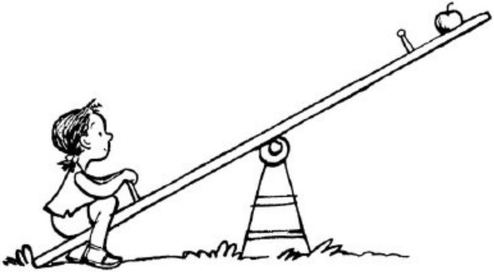
Vandermonde matrix



- Recall the stability of linear model can be studied using the eigenvalues of

$$\mathbf{X}^{\mathrm{T}}\mathbf{X}$$

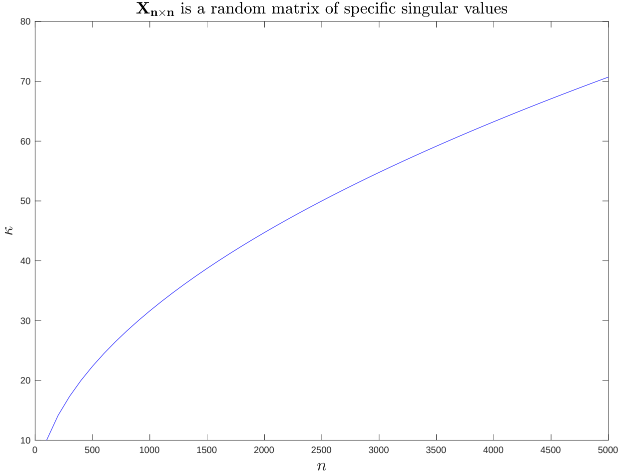
having small eigenvalues indicate columns are nearly linearly dependent.



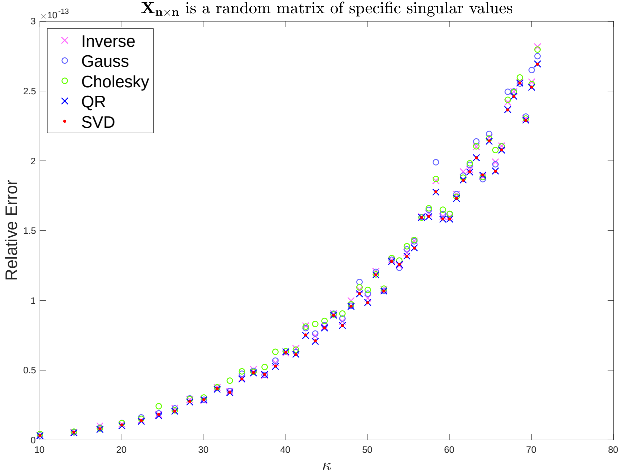
- Using SVD gives us the stability of our model given the data as a byproduct.

$$\kappa = \frac{\max \sigma_i}{\min \sigma_i}$$

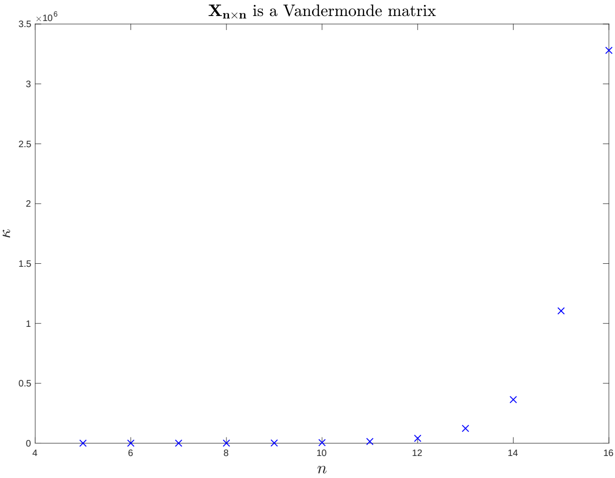
Increasing κ as n increases by construction



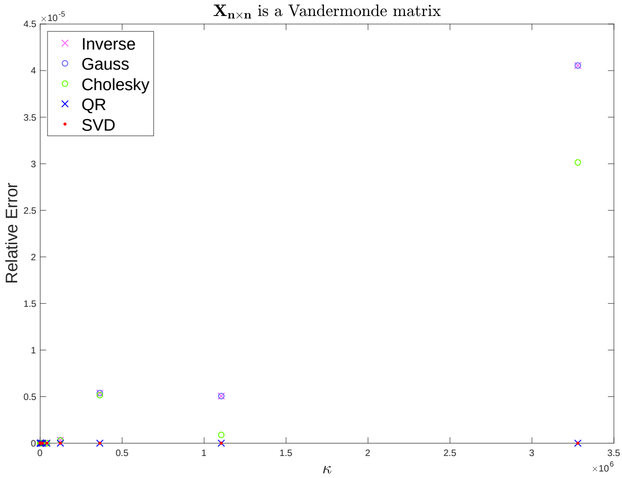
Increasing relative error



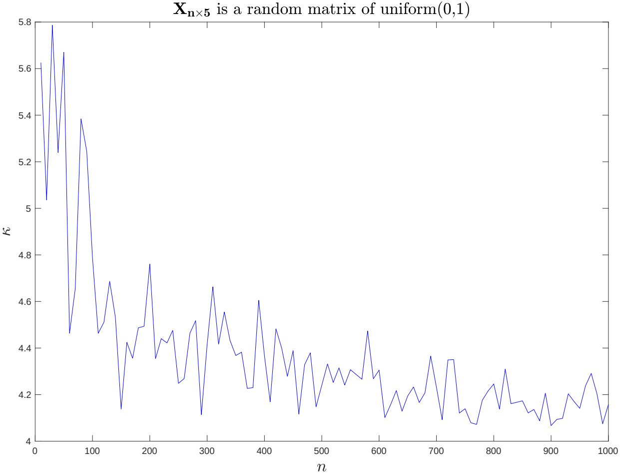
Really big κ



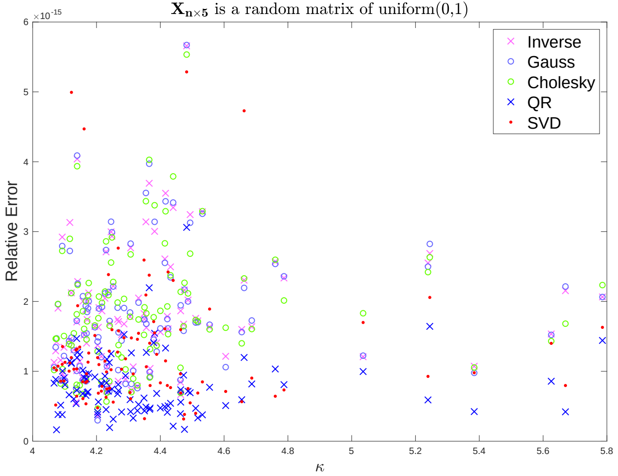
Really big κ really big relative error



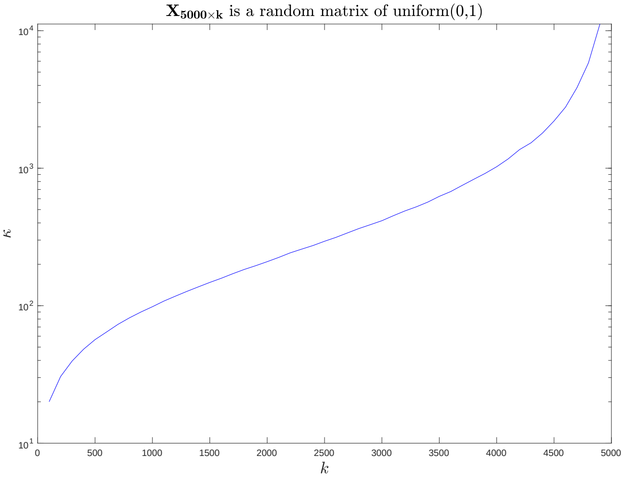
Decreasing κ as n increases for a fixed k



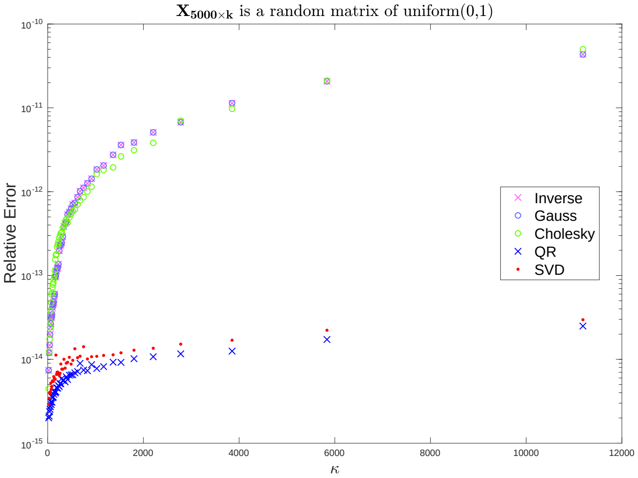
Only having really big κ is problematic



Increasing rapidly κ as k increases for a fixed n



A more complex data is more problematic than a large data



- So far we have considered the data matrix $\mathbf{X}$ that is full rank when solving

$$\arg\min_{\mathbf{b}\in\mathbb{R}^{k+1}} \|\mathbf{y}-\mathbf{X}\mathbf{b}\|^2$$

Q: What happens when $\mathbf{A}$ is rank deficient or "close" to being rank deficient?

- Such problems often arise with big data , e.g. extracting signals from noisy data, digital image restoration as well as big prediction or classification.

Theorem 0.2

Let σ_{min} denote the smallest singular value of $\mathbf{X}$ that is not zero and $\mathbf{U}_c\mathbf{\Sigma}_c\mathbf{V}_c^{\mathrm{T}}$ be the compact singular value decomposition of $\mathbf{X}$. If $\mathbf{b}$ minimises $\|\mathbf{y}-\mathbf{X}\mathbf{b}\|$, then

$$\|\mathbf{b}\| \geq \frac{\left|\mathbf{u}^{\mathrm{T}}\mathbf{b}\right|}{\sigma_{min}}$$

where $\mathbf{u}$ is the last column of $\mathbf{U}_c$. Furthermore changing $\mathbf{y}$ to $\mathbf{y}+\delta\mathbf{y}$ can induce a change of $\delta\mathbf{b}$ to $\mathbf{b}$, where $\|\delta\mathbf{b}\|$ is as large as $\|\delta\mathbf{y}\|/\sigma_{min}$.

Q: What does the last theorem tell us if $\mathbf{A}$ is nearly rank deficient?

Theorem 0.3

Let $\mathbf{X}$ *be a rank* r *matrix of* $n \times (k+1)$ *with* $n \geq k+1$*. If* $r < k+1$*, then there is an* $k+1-r$ *dimensional set of vectors* $\mathbf{b} \in \mathbb{R}^{k+1}$ *that solve the following*

$$\underset{\mathbf{b} \in \mathbb{R}^{k+1}}{\arg\min} \|\mathbf{y} - \mathbf{X}\mathbf{b}\|^2 \qquad \textit{where} \quad \mathbf{y} \in \mathbb{R}^n$$

Furthermore, the singular value decomposition of $\mathbf{X}$ *can be written as*

$$\mathbf{A} = \begin{bmatrix} \mathbf{U}_1 & \mathbf{U}_2 \end{bmatrix} \begin{bmatrix} \mathbf{\Sigma}_{r \times r} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{V}_1 & \mathbf{V}_2 \end{bmatrix}^{\mathrm{T}}, \quad \textit{where } \mathbf{U}_1 \textit{ and } \mathbf{V}_1 \textit{ have } r \textit{ columns,}$$

and all the minimisers take the following form

$$\mathbf{b} = \mathbf{V}_1 \mathbf{\Sigma}_{r \times r}^{-1} \mathbf{U}_1^{\mathrm{T}} \mathbf{y} + \mathbf{V}_2 \mathbf{z}, \qquad \textit{for any } \mathbf{z} \in \mathbb{R}^{k+1-r}.$$

Q: Which of those minimisers shall we use as the best $\mathbf{b}$?