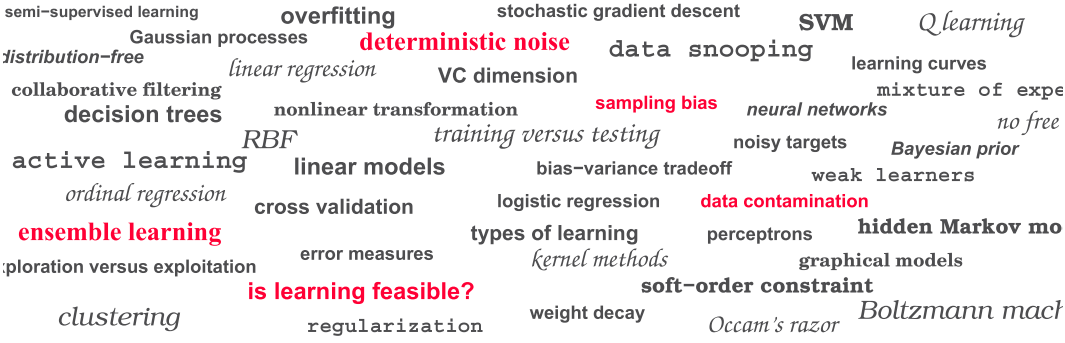
FSAN/ELEG815: Statistical Learning

Gonzalo R. Arce

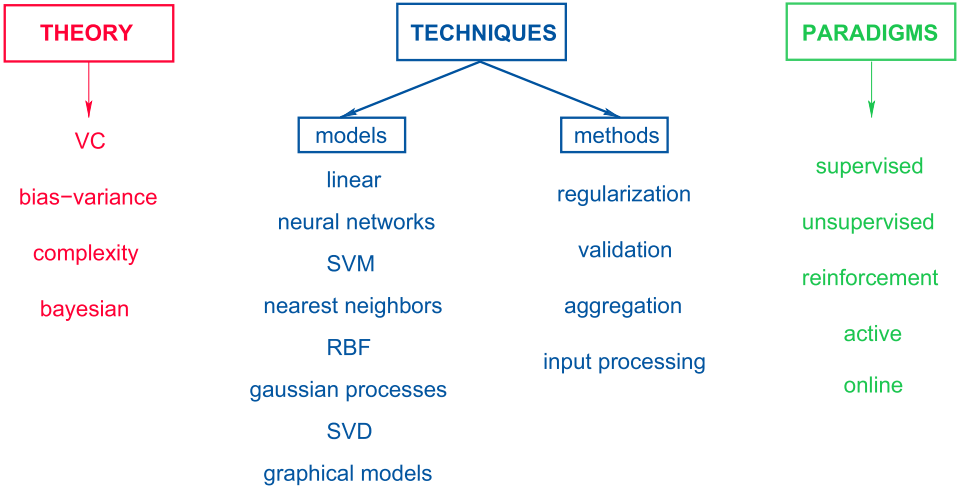
Department of Electrical and Computer Engineering
University of Delaware

Epilogue

It is a Jungle Out There



The Map

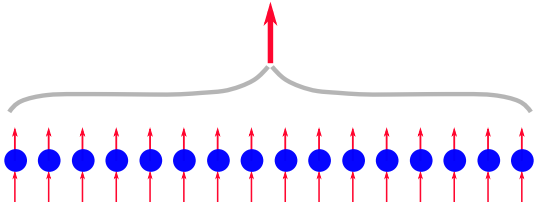

THEORY
VC
bias−variance
complexity
bayesian
TECHNIQUES
models
linear
neural networks
SVM
nearest neighbors
RBF
gaussian processes
SVD
graphical models
methods
regularization
validation
aggregation
input processing
PARADIGMS
supervised
unsupervised
reinforcement
active
online

Outline

1. Aggregation.
2. Kernel Methods.
3. Randomized Singular Value Decomposition.

What is Aggregation?

Combining different solutions $h_1, h_2, ..., h_T$ that were trained on $\mathcal{D}$

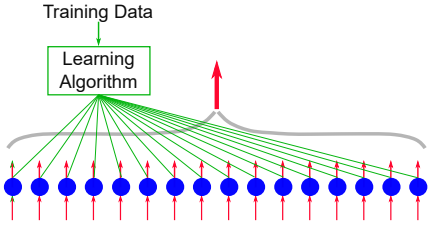


Regression: take an average.
Classification: take a vote.

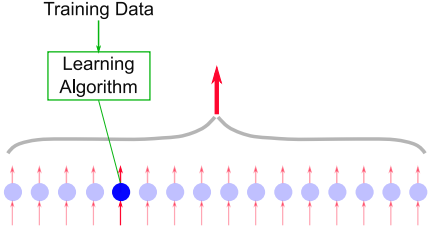
a.k.a ensemble learning and boosting.

Different from 2-Layer Learning

In a 2-layer model, all units learn **jointly**:



In aggregation, they learn **independently** then get combined:



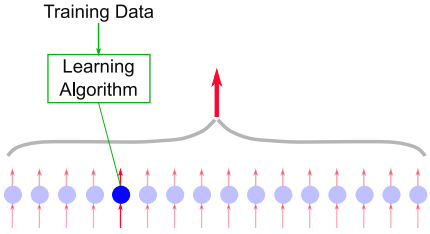
Two Types of Aggregation

After the fact: combines existing solutions

Example: Netflix teams merging "blending".

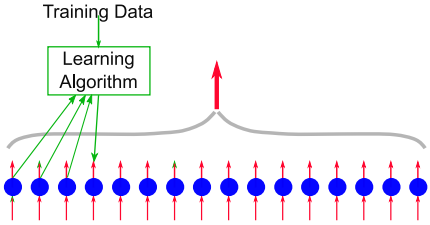
Before the fact: creates solutions to be combined

Example: Bagging - resampling $\mathcal{D}$.



Decorrelation - Boosting

Create $h_1, ..., h_t, ...$ sequentially:
Makes h_t decorrelated with previous h's:



Emphasize points in $\mathcal{D}$ that where misclassified.
Chose weight of h_t based on $E_{\text{in}}(h_t)$.

Blending - After the Fact

For regression, $h_1, h_2, ..., h_T \longrightarrow g(x) = \sum_{t=1}^{T} \alpha_t h_t(\mathbf{x})$

Principled choice of α_t's: minimize the error on an "aggregation data set"

pseudo-inverse

Some α_t's can come out negative.

Most valuable h_t in the blend?

Outline

1. Aggregation.
2. Kernel Methods.
3. Randomized Singular Value Decomposition.

Support Vector Machines - "Support Vectors" in $\mathcal{X}$ Space

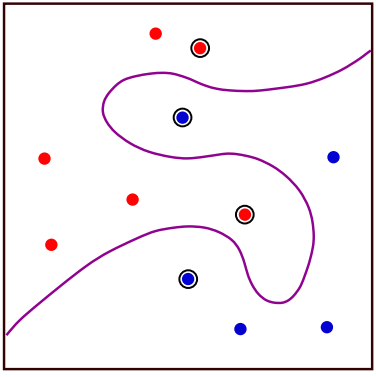
Support vectors live in the $\mathcal{Z}$ space.

In the $\mathcal{X}$ space, "pre-images" of support vectors.

The margin is maintened in the $\mathcal{Z}$ space.

Generalization result

$$\mathbb{E}[E_{out}] \leq \frac{\mathbb{E}[\#\text{ of SV's}]}{N-1}$$



What Do We Need from the $\mathcal{Z}$ Space?

$$\mathcal{L}(\alpha) = \sum_{n=1}^{N} \alpha_n - \frac{1}{2} \sum_{n=1}^{N} \sum_{m=1}^{N} y_n y_m \alpha_n \alpha_m \mathbf{z}_n^\top \mathbf{z}_m$$

Constraints: $\alpha_n \geq 0$ for $n = 1, ..., N$ and $\sum_{n=1}^{N} \alpha_n y_n = 0$.

$$\boxed{g(\mathbf{x}) = \text{sign}(\mathbf{w}^\top \mathbf{z} + b)} \rightarrow \text{need } \mathbf{z}_n^\top \mathbf{z}.$$

where $\mathbf{w} = \sum_{\mathbf{z}_n \text{is SV}} \alpha_n y_n \mathbf{z}_n$,

and $b : y_m(\mathbf{w}^\top \mathbf{z} + b) = 1) \rightarrow$ need $\mathbf{z}_n^\top \mathbf{z}_m$

Generalized Inner Product

Given two points $\mathbf{x}$ and $\mathbf{x}' \in \mathcal{X}$, we need $\mathbf{z}^\top \mathbf{z}'$

Let $\mathbf{z}^\top \mathbf{z}' = K(\mathbf{x}, \mathbf{x}')$ $\rightarrow$ The Kernel "Inner product" of $\mathbf{x}$ and $\mathbf{x}'$.

Example: $\mathbf{x} = (x_1, x_2) \longrightarrow$2nd-order $\mathbf{\Phi}$.

$$\mathbf{z} = \Phi(\mathbf{x}) = (1, x_1, x_2, x_1^2, x_2^2, x_1 x_2).$$

$$K(\mathbf{x}, \mathbf{x}') = \mathbf{z}^\top \mathbf{z} = 1 + x_1 x_1' + x_2 x_2' + x_1^2 x_1'^2 + x_2^2 x_2'^2 + + x_1 x_1' x_2 x_2'$$

The Trick

Can we compute $K(\mathbf{x}, \mathbf{x}')$ without transforming $\mathbf{x}$ and $\mathbf{x}'$?

Example: Consider

$$\begin{aligned} K(\mathbf{x}, \mathbf{x}') &= (1 + \mathbf{x}^\top \mathbf{x}')^2 \\ &= (1 + x_1 x_1' + x_2 x_2')^2 \\ &= 1 + x_1^2 x_1'^2 + x_2^2 x_2^2 + 2x_1 x_1' + 2x_2 x_2' + 2x_1 x_1' x_2 x_2' \end{aligned}$$

This is an inner product!

$$(1, x_1^2, x_2^2, \sqrt{2}x_1, \sqrt{2}x_2, \sqrt{2}x_1 x_2)$$

$$(1, x_1'^2, x_2'^2, \sqrt{2}x_1', \sqrt{2}x_2', \sqrt{2}x_1' x_2')$$

The Polynomial Kernel

$\mathcal{X} = \mathbb{R}^d$ and $\mathbf{\Phi} : \mathcal{X} \to \mathcal{Z}$ is polynomial of order Q.

The "equivalent" kernel:

$$\begin{aligned} K(\mathbf{x}, \mathbf{x}') &= (1 + \mathbf{x}^\top \mathbf{x}')^Q \\ &= (1 + x_1 x_1' + x_2 x_2' + \ldots + x_d x_d)^Q \end{aligned}$$

Compare for $d = 10$ and $Q = 100$

Can adjust scale: $K(\mathbf{x}, \mathbf{x}') = (a\mathbf{x}^\top, \mathbf{x}' + b)^Q$

We Only Need $\mathcal{Z}$ to Exists!

If $K(\mathbf{x},\mathbf{x}')$ is an inner product in **some** space $\mathcal{Z}$, we are good.

Example: $K(\mathbf{x},\mathbf{x}') = e^{-\gamma\|\mathbf{x}-\mathbf{x}'\|^2}$

Infinite-dimensional $\mathcal{Z}$: take simple case (one dimension).

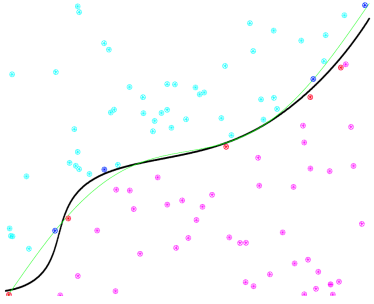
$$K(\mathbf{x},\mathbf{x}') = e^{-(x-x')^2}$$
$$= e^{-x^2} e^{-x'^2} \underbrace{\sum_{k=0}^{\infty} \frac{2^k x^k x'^k}{k!}}_{e^{2xx'}}$$

This Kernel in Action

Slightly non-separable case:

Transforming $\mathcal{X}$ into $\infty-$ dimensional $\mathcal{Z}$.

Overkill? Count the support vectors.



Kernel Formulation SVM

Remember quadratic programming? The only difference now is:

$$\underbrace{\begin{bmatrix} y_1 y_1 K(\mathbf{x}_1, \mathbf{x}_1) & y_1 y_2 K(\mathbf{x}_1, \mathbf{x}_2) & \dots & y_1 y_N K(\mathbf{x}_1, \mathbf{x}_N) \\ y_2 y_1 K(\mathbf{x}_2, \mathbf{x}_1) & y_2 y_2 K(\mathbf{x}_2, \mathbf{x}_2) & \dots & y_2 y_N K(\mathbf{x}_2, \mathbf{x}_N) \\ \dots & \dots & \dots & \dots \\ y_N y_1 K(\mathbf{x}_N, \mathbf{x}_1) & y_N y_2 K(\mathbf{x}_N, \mathbf{x}_1) & \dots & y_N y_N K(\mathbf{x}_N, \mathbf{x}_N) \end{bmatrix}}_{\text{quadratic coefficients}}$$

Everything else is the same

The Final Hypothesis

Express $g(\mathbf{x}) = \text{sign}(\mathbf{w}^\top \mathbf{z} + b)$ in terms of $K(-,-)$.

$$\mathbf{w} = \sum_{\mathbf{z}_n \text{ in SV}} \alpha_n y_n \mathbf{z}_n \Longrightarrow g(\mathbf{x}) = \text{sign}\left(\sum_{\alpha_n > 0} \alpha_n y_n K(\mathbf{x}_n, \mathbf{x}) + b \right)$$

where $b = y_m - \sum_{\alpha_n > 0} \alpha_n y_n K(\mathbf{x}_n, \mathbf{x}_m)$

for any support vector $(\alpha_m > 0)$.

How Do We Know That $\mathcal{Z}$ Exists...

... for a given $K(\mathbf{x}, \mathbf{x}')$? valid kernel

Three approaches:

1. By construction.
2. Math properties (*Mercer's condition*)
3. Who cares? ☺

Design Your Own Kernel

$K(\mathbf{x}, \mathbf{x}')$ is a valid iff:

1. It is symmetric

2. The matrix:

$$\begin{bmatrix} K(\mathbf{x}_1, \mathbf{x}_1) & K(\mathbf{x}_1, \mathbf{x}_2) & ... & K(\mathbf{x}_1, \mathbf{x}_N) \\ K(\mathbf{x}_2, \mathbf{x}_1) & K(\mathbf{x}_2, \mathbf{x}_2) & ... & K(\mathbf{x}_2, \mathbf{x}_N) \\ ... & ... & ... & ... \\ K(\mathbf{x}_N, \mathbf{x}_1) & K(\mathbf{x}_N, \mathbf{x}_1) & ... & K(\mathbf{x}_N, \mathbf{x}_N) \end{bmatrix}$$

is positive semi-definite

for any $\mathbf{x}_1, ..., \mathbf{x}_n$ (Mercer's condition)

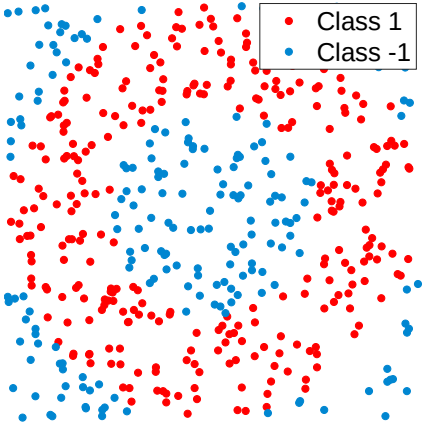
Kernel Methods: Example

- 500 observations 2 dimensions.
- Radial Basis Function kernel (RBF):

$$K(x_i, x_j) = e^{-\gamma \|x_i - x_j\|_2^2}$$

- $\gamma = 0.25$

$$\mathbf{K} = \begin{bmatrix} 1 & 0.28 & 0.52 & ... & 0.79 \\ 0.28 & 1 & 0.03 & ... & 0.08 \\ 0.51 & 0.03 & 1 & ... & 0.91 \\ ... & ... & ... & ... & \\ 079 & 0.08 & 0.91 & ... & 1 \end{bmatrix}$$



Kernel Methods: Example

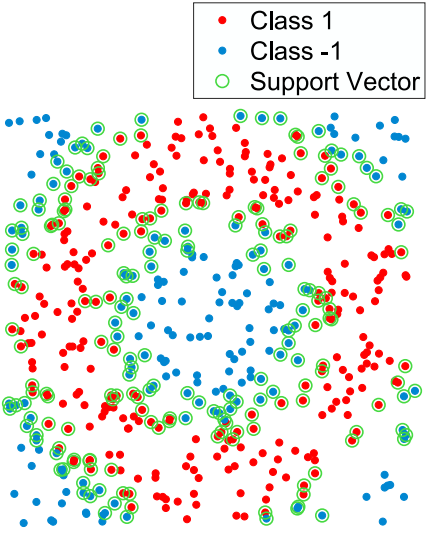
- Solve the quadratic problem (Matlab toolbox):

$$\min_{\alpha} \frac{1}{2}\alpha^{\top}\mathbf{M}\alpha + (-\mathbf{1}^{\top})\alpha,$$

where

$$\mathbf{M}(i,j) = y_i y_j K(\mathbf{x}_i, \mathbf{x}_j)$$

$$\alpha = \begin{bmatrix} 0 & 10 & 10 & 0 & \dots & 10 & 0 & 0 \end{bmatrix}^{\top}$$



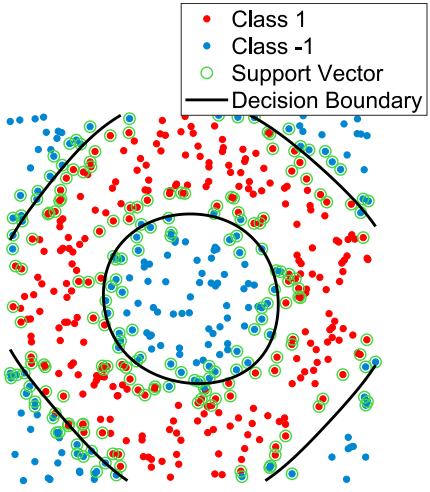
Kernel Methods: Example

- The model:

$$b = y_m - \sum_{\alpha_n > 0} \alpha_n y_n K(\mathbf{x}_n, \mathbf{x}_m)$$

$$\boxed{b = -7.78}$$

$$g(\mathbf{x}) = \text{sign}\left(\sum_{\alpha_n > 0} \alpha_n y_n K(\mathbf{x}_n, \mathbf{x}) + b \right)$$



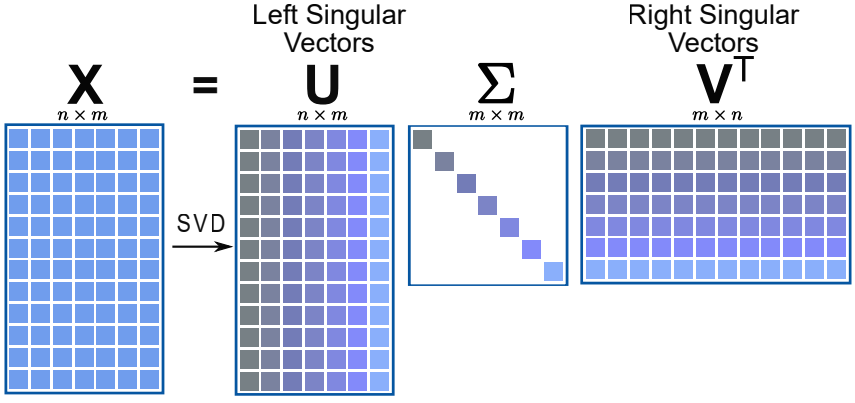
$$g(\mathbf{x}) = \text{sign}\left((-1)10K(\mathbf{x}_2, \mathbf{x}) + (-1)10K(\mathbf{x}_3, \mathbf{x}) + (1)10K(\mathbf{x}_{21}, \mathbf{x}) + \ldots - 7.78\right)$$

Outline

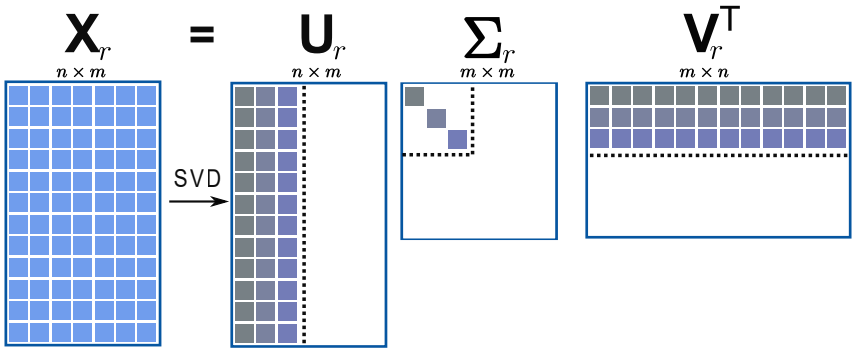
1. Aggregation.
2. Kernel Methods.
3. Randomized Singular Value Decomposition.

Randomized Singular Value Decomposition

SVD: Low-rank approximation and PCA



Randomized Singular Value Decomposition



Low-rank approximation

$$\mathbf{U}_r\mathbf{U}_r^\top\mathbf{X} = \underset{\mathbf{Z}:\, rank(\mathbf{Z})=r}{\operatorname{argmin}} \| \mathbf{X} - \mathbf{Z} \|_2$$

Randomized Singular Value Decomposition

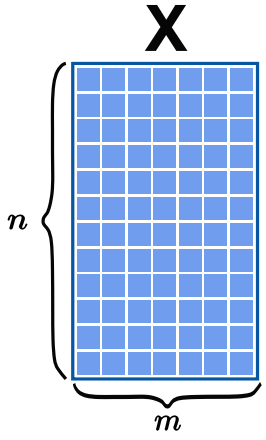
SVD $\longrightarrow$ Computationally expensive $\mathcal{O}(nm^2)$

Dimensions $\longrightarrow$ Increasing (4k,8k...)

Assumption: Low intrinsic rank r.

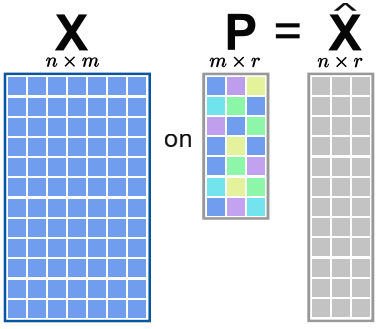
$$\mathbf{X} \approx \mathbf{U}_r \mathbf{\Sigma} \mathbf{V}_r^\top$$

Randomly sample the column space of $\mathbf{X} \in \mathbb{R}^{n \times m}$.



Randomized Singular Value Decomposition

Project $\mathbf{X}$ into $\mathbf{X}$:



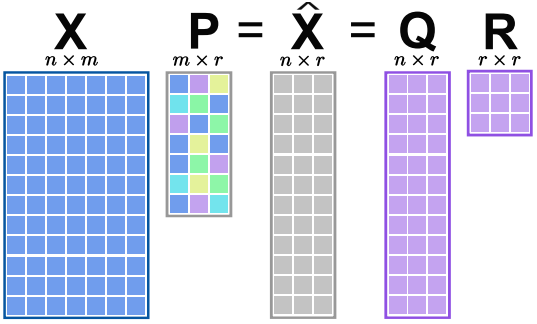
$$\begin{aligned} \| \mathbf{X} - \widehat{\mathbf{U}}_r \widehat{\mathbf{U}}_r^\top \mathbf{X} \|_F &\leq (1+\epsilon) \| \mathbf{X} - \mathbf{X}_r \|_F = (1+\epsilon) \sum_{i=r+1}^{m} \sigma_i^2 \\ &\leq (1+\epsilon) \| \mathbf{X} - \mathbf{U}_r \mathbf{U}_r^\top \mathbf{X} \|_F \end{aligned}$$

Randomized Singular Value Decomposition: Step 1

- Random projection $\mathbf{P} \in \mathbb{R}^{m \times r}$.
- $\widehat{\mathbf{X}} = \mathbf{XP}$, $\widehat{\mathbf{X}} \in \mathbb{R}^{n \times r}$
- QR decomposition $\widehat{\mathbf{X}} = \mathbf{QR}$.

$\mathbf{Q}$: Orthonormal basis for $\widehat{\mathbf{X}}$ (and $\mathbf{X}$).

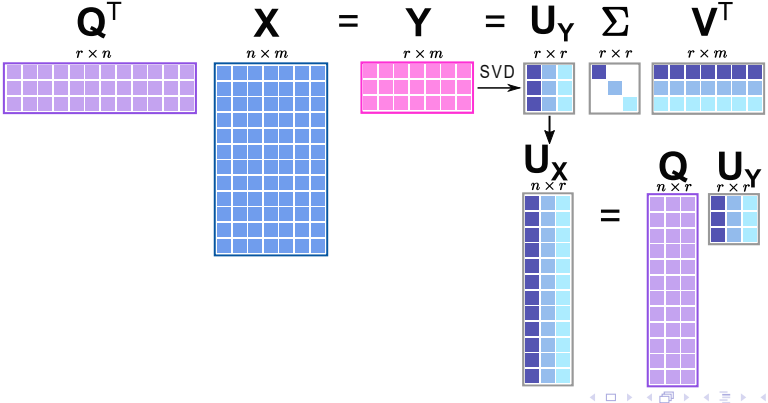
$$\mathbf{X} \approx \mathbf{Q}\mathbf{Q}^{\top}\mathbf{X}$$



Randomized Singular Value Decomposition: Step 2

- Project $\mathbf{X}$ into $\mathbf{Q}$: $\mathbf{Y} = \mathbf{Q}^\top \mathbf{X}$, $\mathbf{Y} \in \mathbb{R}^{r \times m}$.
- Compute SVD: $\mathbf{Y} = \mathbf{U_Y} \Sigma \mathbf{V}^\top$
- Complexity ofs SVD of $\mathbf{Y}$ $<$ Complexity of SVD of $\mathbf{X}$

$\Sigma \mathbf{V}^\top$ Same as for $\mathbf{X}$ and $\boxed{\mathbf{U_X} = \mathbf{Q}\mathbf{U_Y}}$



Randomized Singular Value Decomposition

Proof:

$$\begin{aligned} \mathbf{Y} &= \mathbf{Q}^\top \mathbf{X} \quad \longrightarrow \quad \text{SVD of } \mathbf{Q} \text{ and } \mathbf{X} \\ \mathbf{Y} &= (\mathbf{U_Q}\Sigma_\mathbf{Q}\mathbf{V_Q}^\top)^\top \mathbf{U_X}\Sigma_\mathbf{X}\mathbf{V_X}^\top \\ \mathbf{Y} &= \mathbf{V_Q}\Sigma_\mathbf{Q}^\top \mathbf{U_Q}^\top \mathbf{U_X}\Sigma_\mathbf{X}\mathbf{V_X}^\top \end{aligned}$$

Note that $\mathbf{Q}\mathbf{Q}^\top = \mathbf{I}$ (QR decomposition) $\longrightarrow \mathbf{U_Q} = \mathbf{Q}$, $\mathbf{V_Q} = \mathbf{I}$ and $\Sigma_\mathbf{Q} = \mathbf{I}$:

$$\begin{aligned} \mathbf{Y} &= \mathbf{I}\mathbf{I}^\top \mathbf{Q}^\top \mathbf{U_X}\Sigma_\mathbf{X}\mathbf{V_X}^\top \quad \longrightarrow \quad \mathbf{I}\mathbf{I} = \mathbf{I} \\ \mathbf{Y} &= \mathbf{I}\mathbf{Q}^\top \mathbf{U_X}\Sigma_\mathbf{X}\mathbf{V_X}^\top \quad \longrightarrow \quad \mathbf{I}\mathbf{Q} = \mathbf{Q} \\ \mathbf{Y} &= \mathbf{Q}^\top \mathbf{U_X}\Sigma_\mathbf{X}\mathbf{V_X}^\top \quad \longrightarrow \quad \text{SVD of } \mathbf{Y} \\ \mathbf{U_Y}\Sigma_\mathbf{Y}\mathbf{V_Y}^\top &= \mathbf{Q}^\top \mathbf{U_X}\Sigma_\mathbf{X}\mathbf{V_X}^\top \end{aligned}$$

Then $\Sigma_\mathbf{Y} = \Sigma_\mathbf{X}$, $\Sigma_\mathbf{Y} = \Sigma_\mathbf{X}$ and $\mathbf{U_Y} = \mathbf{Q}^\top \mathbf{U_X}$.

Randomized Singular Value Decomposition

Proof:
Solving for $\mathbf{U_X}$:

$$\begin{aligned} \mathbf{U_Y} &= \mathbf{Q}^\top \mathbf{U_X} \longrightarrow \text{Multiply both sides by } \mathbf{Q} \\ \mathbf{Q}\mathbf{U_Y} &= \mathbf{Q}\mathbf{Q}^\top \mathbf{U_X} \longrightarrow \mathbf{Q}\mathbf{Q}^\top = \mathbf{I} \\ \mathbf{Q}\mathbf{U_Y} &= \mathbf{Q}^\top \mathbf{U_X} \end{aligned}$$

$$\boxed{\mathbf{U_X} = \mathbf{Q}\mathbf{U_Y}}$$

Randomized Singular Value Decomposition

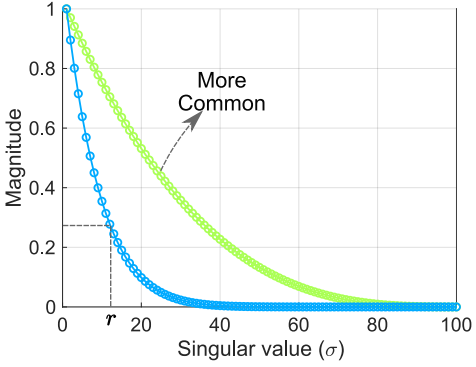
Only r columns aren't always completely effective.
Solutions:

- Oversampling: add p columns to $\mathbf{P}$, $\mathbf{P} \in \mathbb{R}^{m\times(r+p)}$.
- Power iteration: $\mathbf{X}^q = (\mathbf{X}\mathbf{X}^\top)^q\mathbf{X} \longrightarrow \mathbf{X}^q = \mathbf{U}\Sigma^{2q-1}\mathbf{V}^\top$.

If $\mathbf{X}$ is symmetric,

$$\begin{aligned} \mathbf{X} &= \mathbf{U}\Sigma\mathbf{U}^\top \\ \mathbf{X}^q &= \mathbf{U}\Sigma^q\mathbf{U}^\top \end{aligned}$$

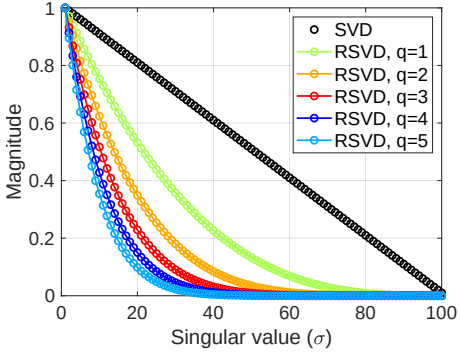
$\| \mathbf{X}^q - \mathbf{X}_r^q \|_F = \sum_{i=r+}^{m} \sigma_i^q \rightarrow$ is much smaller



Randomized Singular Value Decomposition

Power iteration: $\mathbf{X}^q = (\mathbf{X}\mathbf{X}^\top)^q\mathbf{X}$:

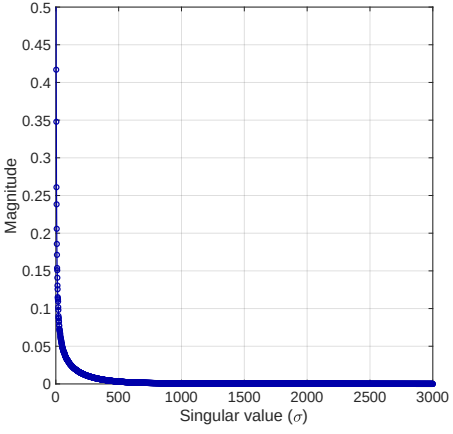
1. $\widehat{\mathbf{X}} = \mathbf{X}\mathbf{P}$
2. Repeat q times $\widehat{\mathbf{X}}^q = \mathbf{X}(\mathbf{X}^\top\widehat{\mathbf{X}}^{q-1})$
3. QR decomposition $\widehat{\mathbf{X}}^q = \mathbf{Q}\mathbf{R}$
4. Project $\mathbf{X}$ into $\mathbf{Q}$: $\mathbf{Y} = \mathbf{Q}^\top\mathbf{X}$.
5. Compute SVD: $\mathbf{Y} = \mathbf{U}_\mathbf{Y}\Sigma\mathbf{V}^\top \longrightarrow \mathbf{U}_\mathbf{X} = \mathbf{Q}\mathbf{U}_\mathbf{Y}$.



- Random Matrix $\mathbf{X} \in \mathbb{R}^{1000\times100}$.
- Singular value spectrum of $\mathbf{X}^q$ decays more rapidly.

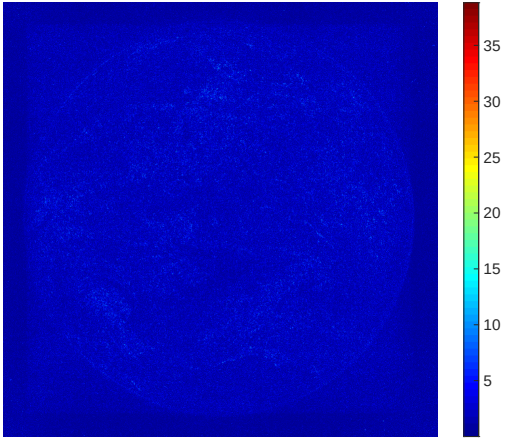
Randomized Singular Value Decomposition: Example

- Planet earth photography: 3000×3000 pixels.
- Rank: $r \approx 400$



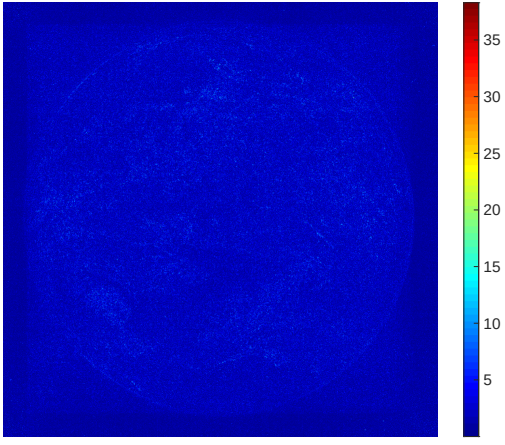
Randomized Singular Value Decomposition: Example

- SVD 400-rank approximation.
- MSE=5.7



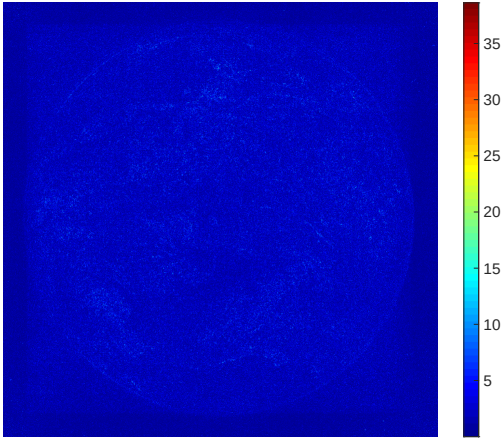
Randomized Singular Value Decomposition: Example

- Randomized SVD 400-rank approximation $\mathbf{P} \in \mathbb{R}^{3000 \times 400}$
- MSE=6.7



Randomized Singular Value Decomposition: Example

- Oversampling randomized SVD 400-rank approximation $p = 20$.
- $\mathbf{P} \in \mathbb{R}^{3000 \times (400+20)}$.
- MSE=6.3



Randomized Singular Value Decomposition: Example

- Power iterations randomized SVD 400-rank approximation $q = 2$.
- MSE=6.05