APS1070

Foundations of Data Analytics and Machine Learning

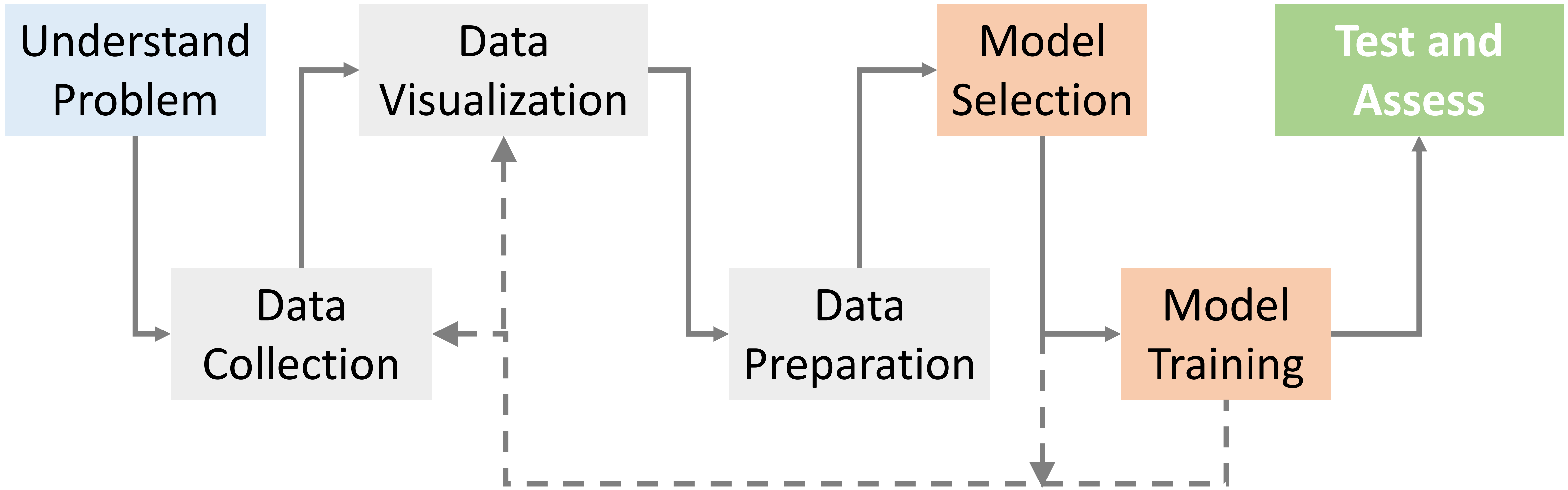
Winter 2022

Week 8:

- *SVD*
- *Applications*
- *Vector Calculus*

Sinisa Colic and Samin Aref

Course Theme



Slide Attribution

These slides contain materials from various sources. Special thanks to the following authors:

- Marc Deisenroth
- Jason Riordon

Last Time

- Looked into PCA for dimensionality reduction
  - Determinant
  - Trace
  - Eigendecomposition
  - PCA Applications

- Today we will introduce SVD for dimensionality reduction and interpretation of data.

Rectangular Matrices

- Eigendecomposition (and PCA) is limited to square matrices

- Q: How then do we achieve matrix decomposition for rectangular matrices?

Agenda

- Recap from last time
- SVD
  - SVD vs Eigendecomposition
  - Dimensionality Reduction
  - Interpretations
- Applications
- Vector Calculus
  - Matrix Differentiation

Theme:
Dimensionality Reduction and Interpretations

Matrix Decompositions Continued

Readings:

- **MML Chapter 4.5-8**

Recap: Principal Component Analysis

Data Matrix (rectangular)

	Mouse 1	Mouse 2	Mouse 3	Mouse 4	Mouse 5	Mouse 6
Gene 1	10	11	8	3	2	1
Gene 2	6	4	5	3	2.8	1

Covariance Matrix (square)

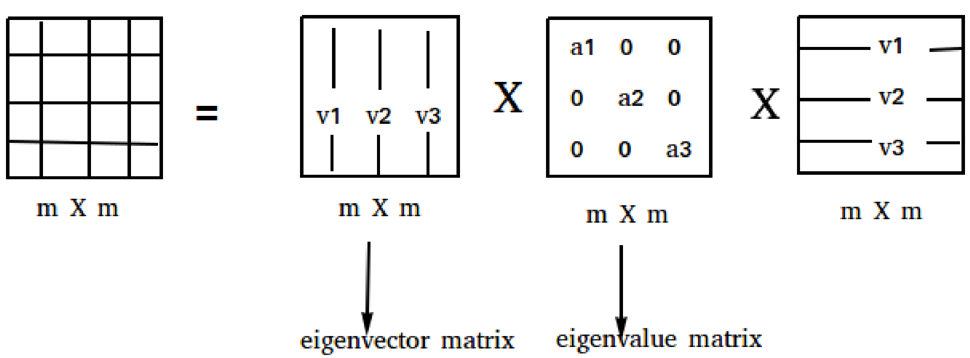
	feature 2	
feature 1	val1	**val2**
	val2	val4

- Taking the square (and symmetrical) covariance matrix we obtain:

$$A = PDP^T$$

Spectral Theorem

- To obtain principal components and associated scores:



Recap: Matrix Decomposition

- We would like a general approach for **decomposing rectangular matrices**.
- PCA is already applicable to rectangular matrices…
- Mathematically PCA is:

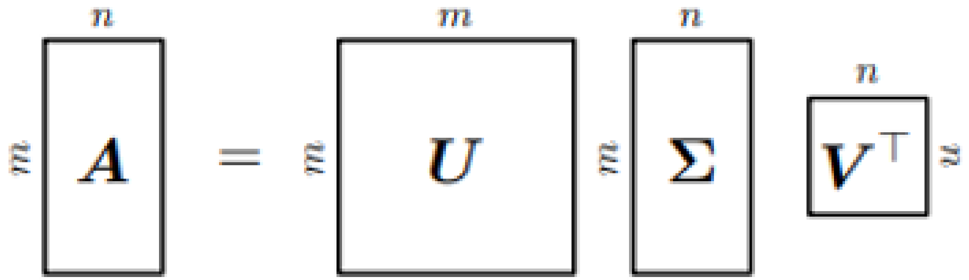
$$A = XX^T = PDP^T$$

when the mean/expectation across each variable/feature is zero

Singular Value Decomposition

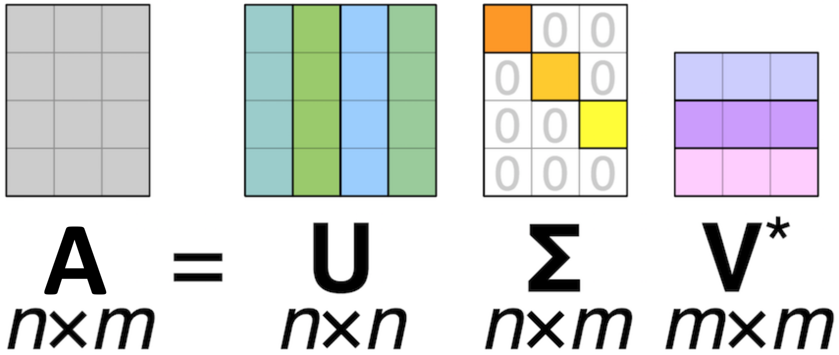
- We would like to obtain the following decomposition for a **rectangular matrix**:

$$A = U\Sigma V^T$$



where U contains the left-singular vectors, V has the right-singular vectors and Σ are the singular values.

Singular Value Decomposition



- Singular values matrix Σ has additional zero padding of rows or columns:

$$\Sigma = \begin{bmatrix} \sigma_1 & 0 & 0 & 0 & \dots & 0 \\ 0 & \ddots & 0 & \vdots & & \vdots \\ 0 & 0 & \sigma_n & 0 & \dots & 0 \end{bmatrix}$$

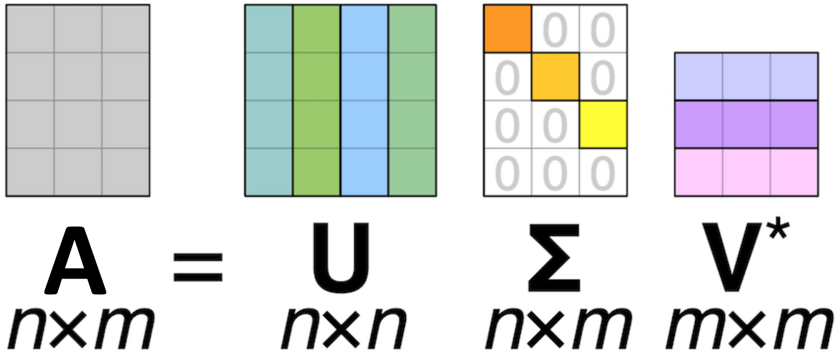
when
n < m

$$\Sigma = \begin{bmatrix} \sigma_1 & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & \sigma_m \\ 0 & \dots & 0 \\ \vdots & & \vdots \\ 0 & \dots & 0 \end{bmatrix}$$

when
m < n

Source: Wikipedia

Singular Value Decomposition



$$
\begin{array}{r}
 \\
\text{Star Wars} \\
\text{Blade Runner} \\
\text{Amelie} \\
\text{Delicatessen}
\end{array}
\begin{array}{c}
\begin{array}{ccc} \text{Ali} & \text{Beatrix} & \text{Chandra} \end{array} \\
\begin{bmatrix} 5 & 4 & 1 \\ 5 & 5 & 0 \\ 0 & 0 & 5 \\ 1 & 0 & 4 \end{bmatrix}
\end{array}
\begin{array}{l}
 \\
\text{Sci-fi} \\
 \\
\text{French} \\

\end{array}
$$

$$
= \begin{array}{c}
\begin{array}{cccc} \text{Sci-fi} & \text{French} & & \end{array} \\
\begin{bmatrix} -0.6710 & 0.0236 & 0.4647 & -0.5774 \\ -0.7197 & 0.2054 & -0.4759 & 0.4619 \\ -0.0939 & -0.7705 & -0.5268 & -0.3464 \\ -0.1515 & -0.6030 & 0.5293 & -0.5774 \end{bmatrix}
\end{array}
$$

$$
\begin{bmatrix} 9.6438 & 0 & 0 \\ 0 & 6.3639 & 0 \\ 0 & 0 & 0.7056 \\ 0 & 0 & 0 \end{bmatrix}
$$

$$
\begin{array}{r}
\text{Sci-fi} \\
\text{French} \\

\end{array}
\begin{bmatrix} -0.7367 & -0.6515 & -0.1811 \\ 0.0852 & 0.1762 & -0.9807 \\ 0.6708 & -0.7379 & -0.0743 \end{bmatrix}
$$

Source: [Wikipedia]

Implications

- If we can decompose our matrix, then that can give us insights into our data.

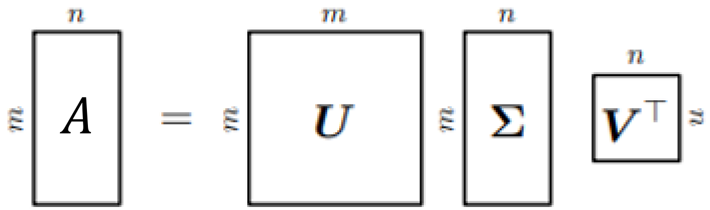
- SVD is behind some of the many machine learning applications
  - recommender systems
  - word embeddings
  - image compressions
  - background removal

SVD Algorithm

- Finding the left and right singular vectors shares some similarities with PCA:
- Right-singular vectors:

$$A^TA = VDV^T$$

- Left-singular vectors:



$$AA^T = UDU^T$$

- AA^T and A^TA are symmetrical (which makes eigendecomposition easier)

SVD Algorithm

- Right-Singular:

$$A^T A$$
$$= (U\Sigma V^T)^T (U\Sigma V^T)$$
$$= V\Sigma U^T U\Sigma V^T$$
$$= V\Sigma^2 V^T$$

eigendecomposition of $A^T A$, with $\Sigma = \sqrt{D}$

- Left-Singular:

$$AA^T$$
$$= (U\Sigma V^T)(U\Sigma V^T)^T$$
$$= U\Sigma V^T V\Sigma U^T$$
$$= U\Sigma^2 U^T$$

eigendecomposition of AA^T with $\Sigma = \sqrt{D}$

SVD Algorithm

- The right-singular vectors (V) and left-singular vectors (U) (which we know to be orthonormal) are connected through the singular value matrix:

$$AV = U\Sigma$$

min(m, n)

$$Av_i = \sigma_i u_i \qquad i = 1, \dots, r$$

$$\frac{1}{\sigma_i} A v_i = u_i$$

Example

Compute the singular value decomposition for matrix A.

$$A = \begin{bmatrix} 1 & 0 & 1 \\ -2 & 1 & 0 \end{bmatrix}$$

$$A = \begin{bmatrix} 1 & 0 & 1 \\ -2 & 1 & 0 \end{bmatrix}$$

$$\underset{2\times 3}{A} = \underset{2\times 2}{U}\,\underset{2\times 3}{\Sigma}\,\underset{3\times 3}{V^T}$$

$$\boxed{V = ?} \longrightarrow \det(A^TA - \lambda I) = 0$$

$$A^TA = \begin{bmatrix} 1 & -2 \\ 0 & 1 \\ 1 & 0 \end{bmatrix}\begin{bmatrix} 1 & 0 & 1 \\ -2 & 1 & 0 \end{bmatrix} = \begin{bmatrix} 5 & -2 & 1 \\ -2 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix}$$

$$\begin{bmatrix} 5-\lambda & -2 & 1 \\ -2 & 1-\lambda & 0 \\ 1 & 0 & 1-\lambda \end{bmatrix} \xrightarrow{\det} \boxed{-\lambda^3+7\lambda^2-6\lambda=0}$$

$$\underline{\lambda(-\lambda^2+7\lambda-6)=0} \quad \begin{cases} \boxed{\lambda_1=0} \\ \boxed{\lambda_2=1} \\ \boxed{\lambda_3=6} \end{cases}$$

$$\lambda=6 \to \begin{bmatrix} -1 & -2 & 1 \\ -2 & -5 & 0 \\ 1 & 0 & -5 \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}=0 \to \operatorname{Span}\left(\begin{bmatrix} 1 \\ -2/5 \\ 1/5 \end{bmatrix}\right) \xrightarrow[\text{Normalize}]{\sqrt{1+\frac{4}{25}+\frac{1}{25}}=\frac{\sqrt{30}}{5}} \begin{bmatrix} \frac{5}{\sqrt{30}} \\ \frac{-2}{\sqrt{30}} \\ \frac{1}{\sqrt{30}} \end{bmatrix}$$

$$\lambda=1 \to \begin{bmatrix} 4 & -2 & 1 \\ -2 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}=0 \to \operatorname{Span}\left(\begin{bmatrix} 0 \\ 1 \\ 2 \end{bmatrix}\right) \xrightarrow[\text{Normalize}]{\sqrt{1+4}=\sqrt{5}} \begin{bmatrix} 0 \\ 1/\sqrt{5} \\ 2/\sqrt{5} \end{bmatrix}$$

$$\lambda=0 \to \begin{bmatrix} 5 & -2 & 1 \\ -2 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}=0 \to \operatorname{span}\left(\begin{bmatrix} 1 \\ 2 \\ -1 \end{bmatrix}\right) \xrightarrow[\text{Normalize}]{\sqrt{1+4+1}=\sqrt{6}} \begin{bmatrix} 1/\sqrt{6} \\ 2/\sqrt{6} \\ -1/\sqrt{6} \end{bmatrix}$$

$$A^TA = \overset{P}{\begin{bmatrix} \frac{5}{\sqrt{30}} & 0 & -\frac{1}{\sqrt{6}} \\ \frac{-2}{\sqrt{30}} & \frac{1}{\sqrt{5}} & -\frac{2}{\sqrt{6}} \\ \frac{1}{\sqrt{30}} & \frac{2}{\sqrt{5}} & \frac{1}{\sqrt{6}} \end{bmatrix}} \overset{D}{\begin{bmatrix} 6 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}} \overset{P^T}{\begin{bmatrix} \cdots \end{bmatrix}}$$

This gives V^T for $\mathrm{A} = U\Sigma V^T$ because $A^TA = VDV^T$

We get Σ from D:

$$\Sigma = \sqrt{D} \longrightarrow \Sigma = \begin{bmatrix} \sqrt{6} & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}$$

$$\boxed{U = ?}$$

$$AA^T = P_2 D_2 P^T \qquad \boxed{U = P}$$

OR

$$u_i = \frac{1}{\sigma_i} A v_i$$

$$u_1 = \frac{1}{\sigma_1} A v_1 = \frac{1}{\sqrt{6}} \begin{bmatrix} 1 & 0 & 1 \\ -2 & 1 & 0 \end{bmatrix} \begin{bmatrix} \frac{5}{\sqrt{30}} \\ -2/\sqrt{30} \\ 1/\sqrt{30} \end{bmatrix}$$

$$u_1 = \begin{bmatrix} 1/\sqrt{5} \\ -2/\sqrt{5} \end{bmatrix}$$

$$u_2 = \frac{1}{\sigma_2} A v_2 \Longrightarrow \underline{U = \frac{1}{\sqrt{5}} \begin{bmatrix} 1 & 2 \\ -2 & 1 \end{bmatrix}}$$

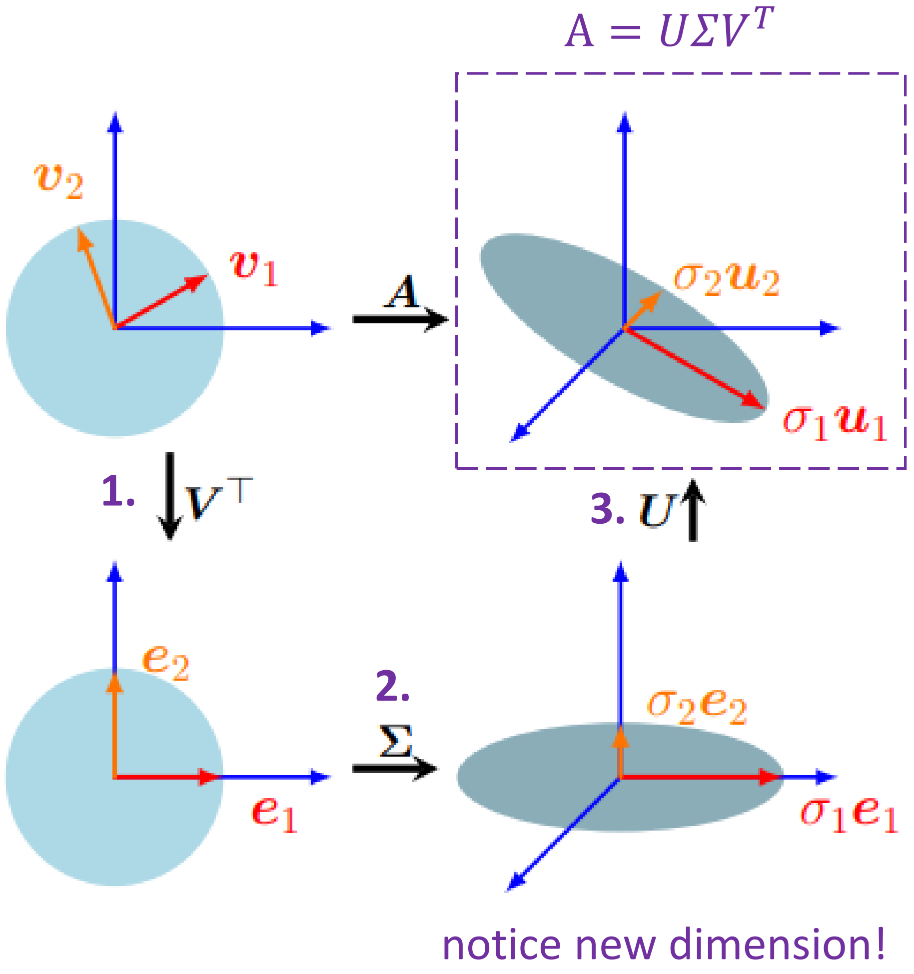
$$u_2 = \begin{bmatrix} 2/\sqrt{5} \\ 1/\sqrt{5} \end{bmatrix}$$

Geometric Interpretation of SVD

- A **rectangular matrix** (n x m) can be factored into:

$$A = U\Sigma V^T$$

- These can be seen as a sequence of transformations:
    1. rotation by right-singular V
    2. scaling by singular values Σ
    3. rotation by left-singular again U



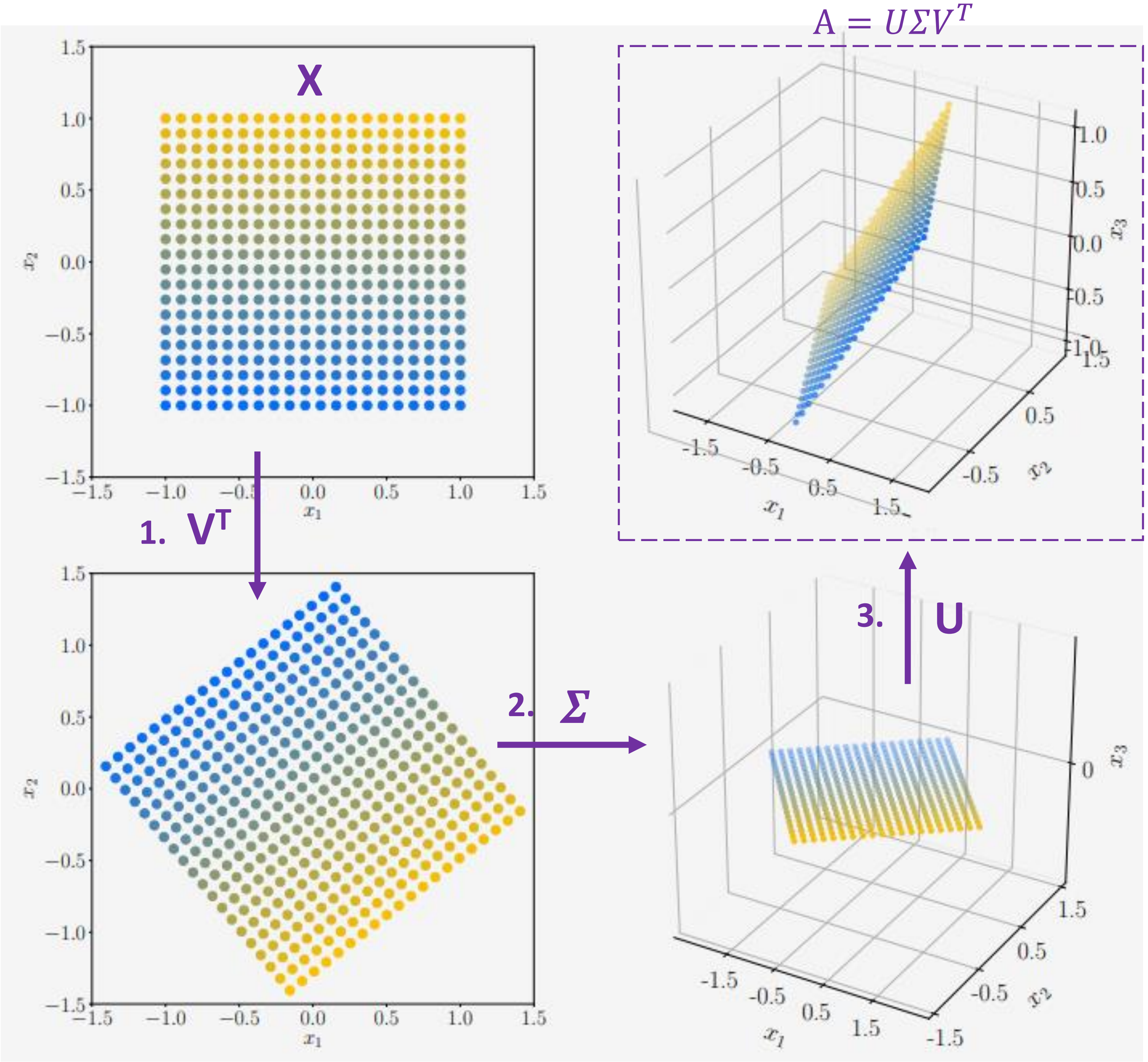
Example

Given: A set of data points X and transformation A.

$$\boldsymbol{A} = \begin{bmatrix} 1 & -0.8 \\ 0 & 1 \\ 1 & 0 \end{bmatrix} = \boldsymbol{U\Sigma V}^\top$$
$$= \begin{bmatrix} -0.79 & 0 & -0.62 \\ 0.38 & -0.78 & -0.49 \\ -0.48 & -0.62 & 0.62 \end{bmatrix} \begin{bmatrix} 1.62 & 0 \\ 0 & 1.0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} -0.78 & 0.62 \\ -0.62 & -0.78 \end{bmatrix}$$

X -> set of vectors

1. V^T -> rotates data X
2. Σ -> singular matrix maps onto $\mathbb{R}^3$ (3^{rd} dimension is 0, stretched in a plane by singular values)
3. U -> rotation within $\mathbb{R}^3$



Similarities to Eigendecomposition -1

- Eigenvalue Decomposition vs Singular Value Decomposition:
  - SVD exists for any matrix (m x n) whether square or rectangular
  - SVD left and right singular vectors can be made orthonormal
  - Both are a composition of three linear mappings:
    1. Change of basis in the domain
    2. Independent scaling of each new basis vector from domain to codomain
    3. Change of basis in codomain
  - SVD domain and codomain can be vector spaces of different dimensions.

Eigendecomposition

$$A = PDP^{-1}$$

Singular Value Decomposition

$$A = U\Sigma V^T$$

Similarities to Eigendecomposition -2

- SVD singular values are real and non-negative
- SVD and eigendecomposition are closely related through their projections:
  - The left-singular vectors of A are eigenvectors of AA^T
  - The right-singular vectors of A are eigenvectors of A^TA
  - The nonzero singular values of A are the square roots of nonzero eigenvalues of AA^T and A^TA

Eigendecomposition

$$A = PDP^{-1}$$

Singular Value Decomposition

$$A = U\Sigma V^T$$

- **For symmetric square matrices (n x n) eigendecomposition and SVD are the same**

  when the features are normalized.

$\longrightarrow$ $\mathbf{P} = \mathbf{U} = \mathbf{V}$

Relationship between PCA and SVD

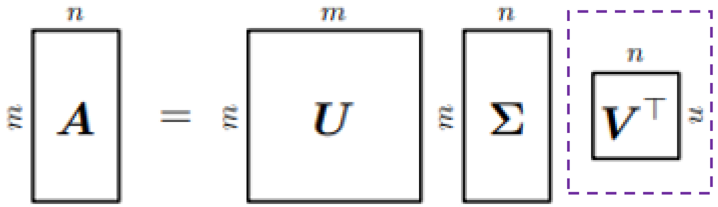
- PCA and SVD are closely related:

$$Cov(A) = \frac{A^T A}{n-1}$$

$$= \frac{(U\Sigma V^T)^T (U\Sigma V^T)}{n-1}$$

$$= \frac{V\Sigma U^T U\Sigma V^T}{n-1}$$

$$= V \frac{\Sigma^2}{n-1} V^T$$



SVD columns of V are principal direction in PCA

- The result is the same form as eigendecomposition of A and hence:

$$\boxed{D} = \frac{\boxed{\Sigma}^2}{n-1}$$

eigenvalues

singular values

Applications of Matrix Decompositions

Dimensionality Reduction
Data Interpretation

Recap: Eigenfaces

- Last time we saw how PCA could be used to obtain a compressed version of face images.

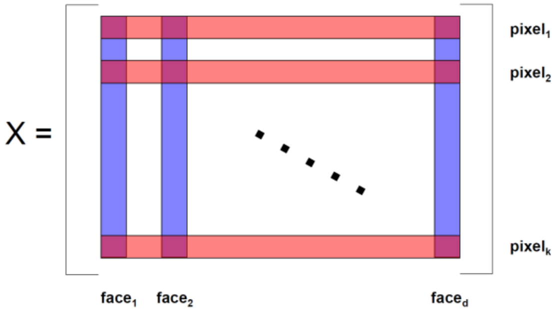
1.

Normalized Face Data

2.

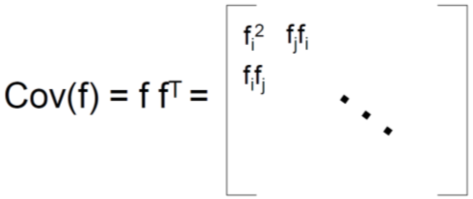
The Data Matrix X

- Matrix with columns as faces, rows as pixels



3.

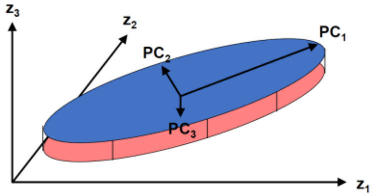
The Covariance Matrix



4.

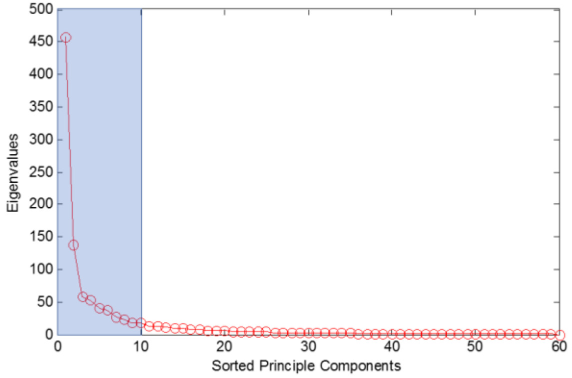
Principle Components

- **Why look at eigenvectors of covariance?**

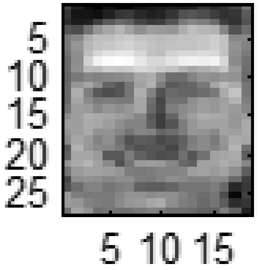


- **If data lives in linear subspace…**
  - Covariance indicates principle data dimensions
  - Then eigenvectors = 'principle data components'

5.



6.



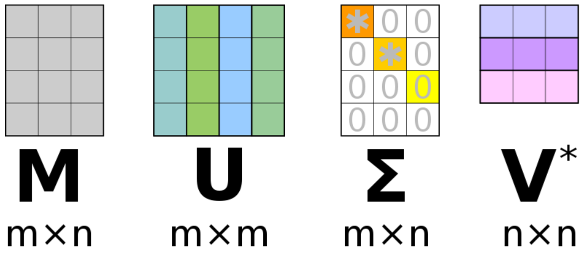
Algorithmic complexity

- Eigenvalue Decomposition:
  - For an mxn matrix, is $O(n^3)$ if there are n features
  - Impractical for large n

$$A = PDP^{-1}$$

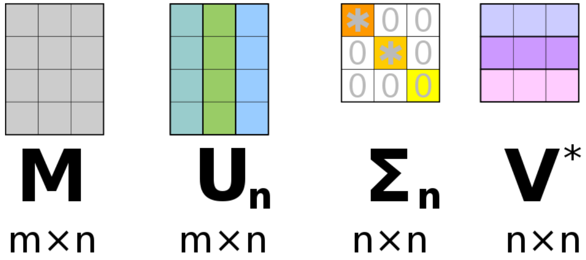
- Full SVD:
  - For an mxn matrix, it is $O(mn \min(n, m))$
  - More computationally tractable for large n or large m

Full SVD $A = U\Sigma V^T$



- Thin (economy-sized) SVD:
  - For an mxn matrix, it is $O(\max(m, n). \min(m, n))$
  - Even faster and more economical

Thin SVD $A = U_n\Sigma_n V^T$



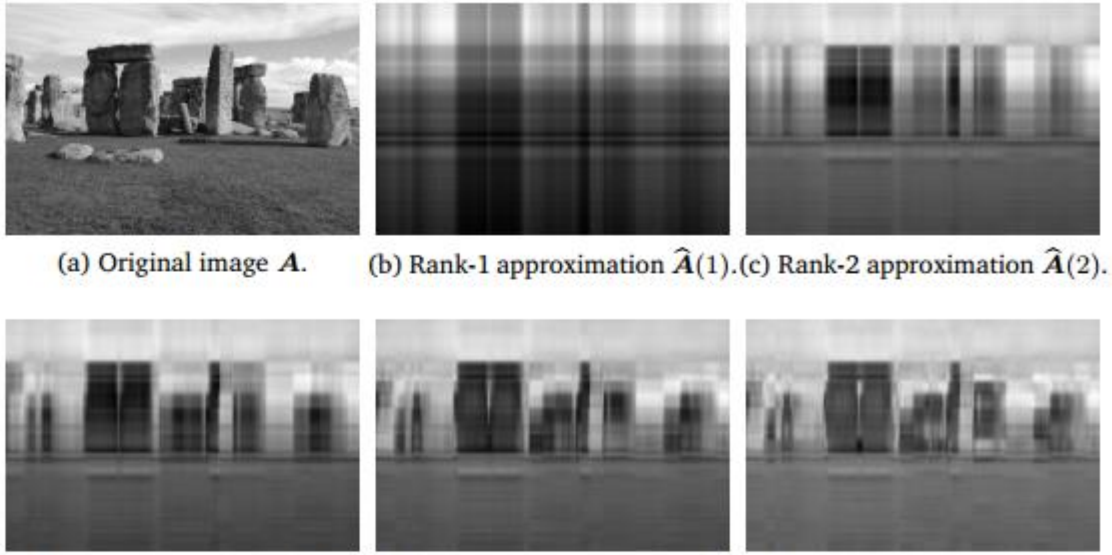
Source: [Wikipedia]

Image Compression

- Analogous to decomposing face we can also decompose a single image for compression.

$$A = U\Sigma V^\top \in \mathbb{R}^{m\times n}$$

$$A = \sum_{i=1}^{r} \sigma_i u_i v_i^\top = \sum_{i=1}^{r} \sigma_i A_i$$

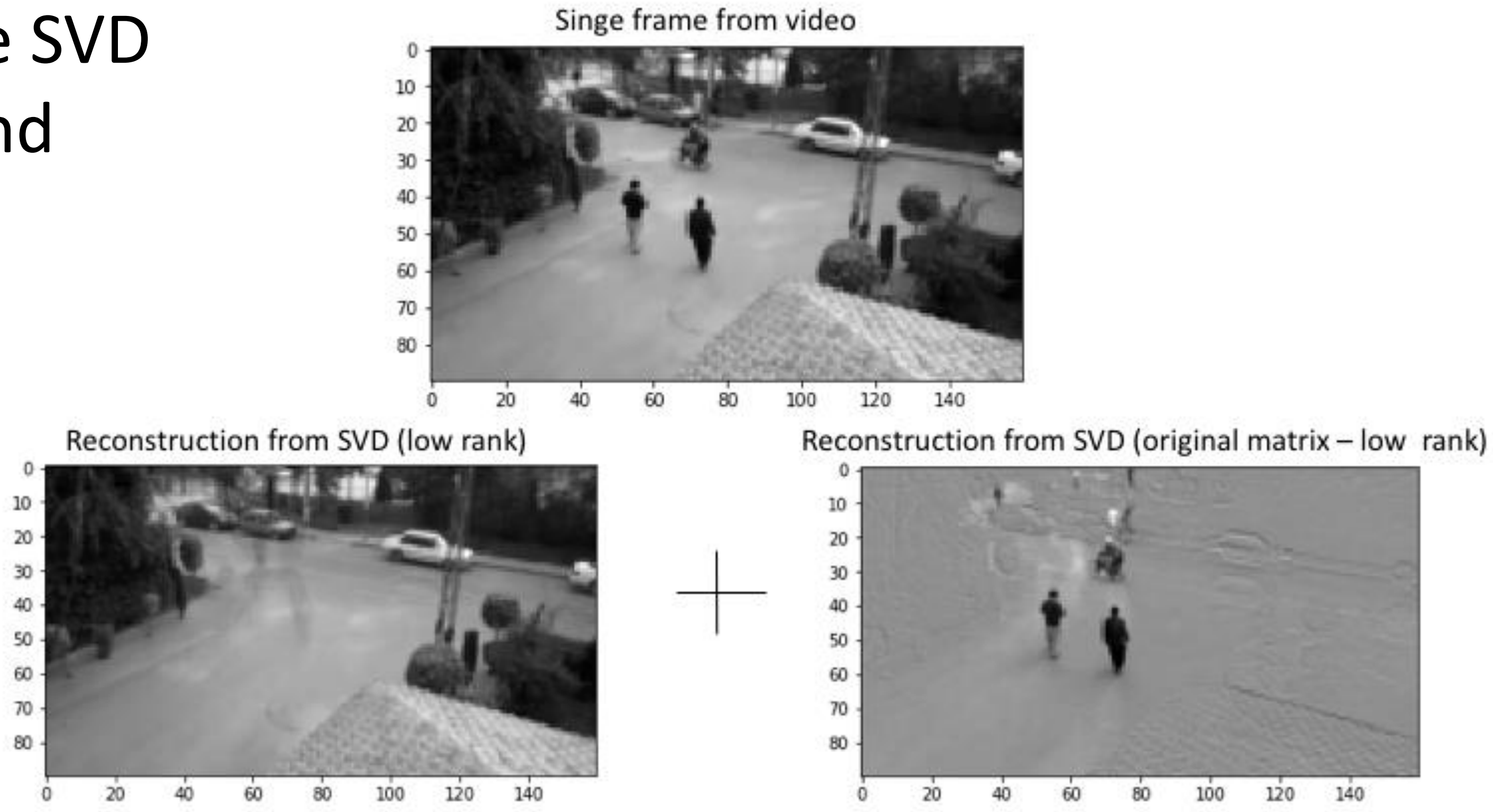


(a) Original image $\boldsymbol{A}$. (b) Rank-1 approximation $\widehat{\boldsymbol{A}}(1)$. (c) Rank-2 approximation $\widehat{\boldsymbol{A}}(2)$.

(d) Rank-3 approximation $\widehat{\boldsymbol{A}}(3)$. (e) Rank-4 approximation $\widehat{\boldsymbol{A}}(4)$. (f) Rank-5 approximation $\widehat{\boldsymbol{A}}(5)$.

Application 1 – Background Removal

- Given a video we can use SVD to remove the background



Example Google Colab Code

Application 2 – Recommender Systems

- SVD can also be used to find structure in data for making recommendations.

- SVD is strongly related to recommender systems applications:
  - Movie recommendations
  - Product recommendations
  - Restaurant recommendations
  - Website recommendations
  - …

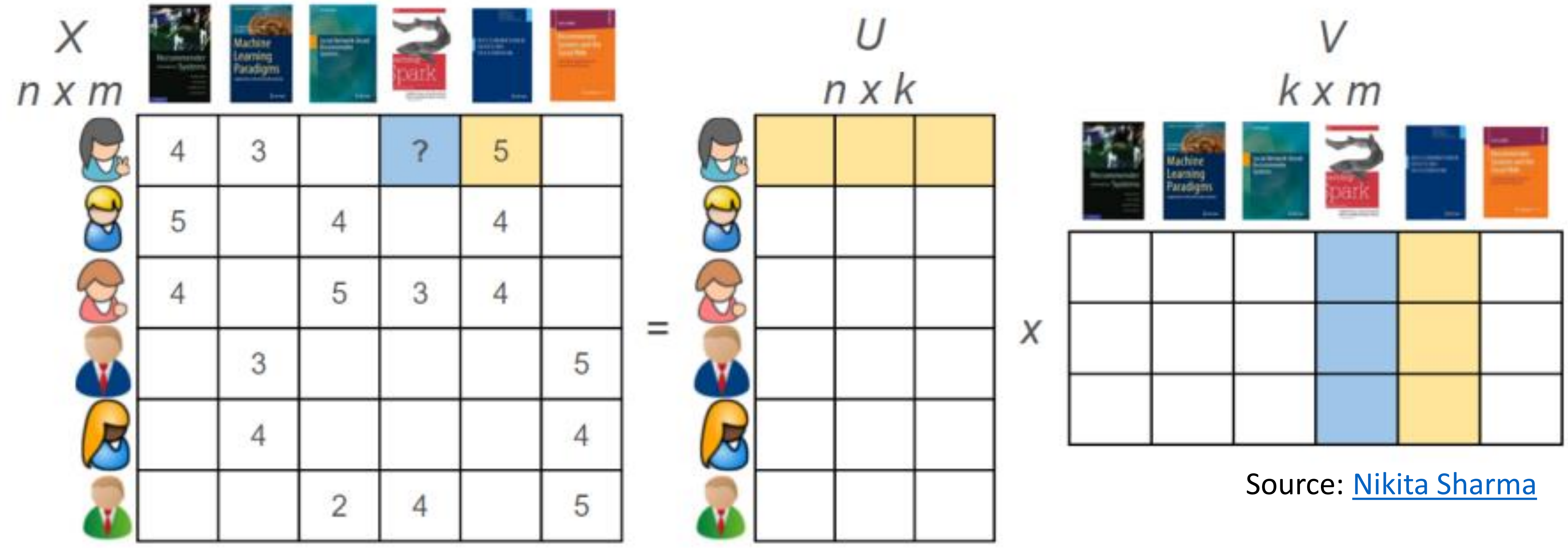
$$\begin{array}{r c} & \begin{array}{ccc} \text{Ali} & \text{Beatrix} & \text{Chandra} \end{array} \\ \begin{array}{r} \text{Star Wars} \\ \text{Blade Runner} \\ \text{Amelie} \\ \text{Delicatessen} \end{array} & \begin{bmatrix} 5 & 4 & 1 \\ 5 & 5 & 0 \\ 0 & 0 & 5 \\ 1 & 0 & 4 \end{bmatrix} \end{array}$$

$$= \begin{bmatrix} -0.6710 & 0.0236 & 0.4647 & -0.5774 \\ -0.7197 & 0.2054 & -0.4759 & 0.4619 \\ -0.0939 & -0.7705 & -0.5268 & -0.3464 \\ -0.1515 & -0.6030 & 0.5293 & -0.5774 \end{bmatrix}$$

$$\begin{bmatrix} 9.6438 & 0 & 0 \\ 0 & 6.3639 & 0 \\ 0 & 0 & 0.7056 \\ 0 & 0 & 0 \end{bmatrix}$$

$$\begin{bmatrix} -0.7367 & -0.6515 & -0.1811 \\ 0.0852 & 0.1762 & -0.9807 \\ 0.6708 & -0.7379 & -0.0743 \end{bmatrix}$$

Example: Collaborative Filtering

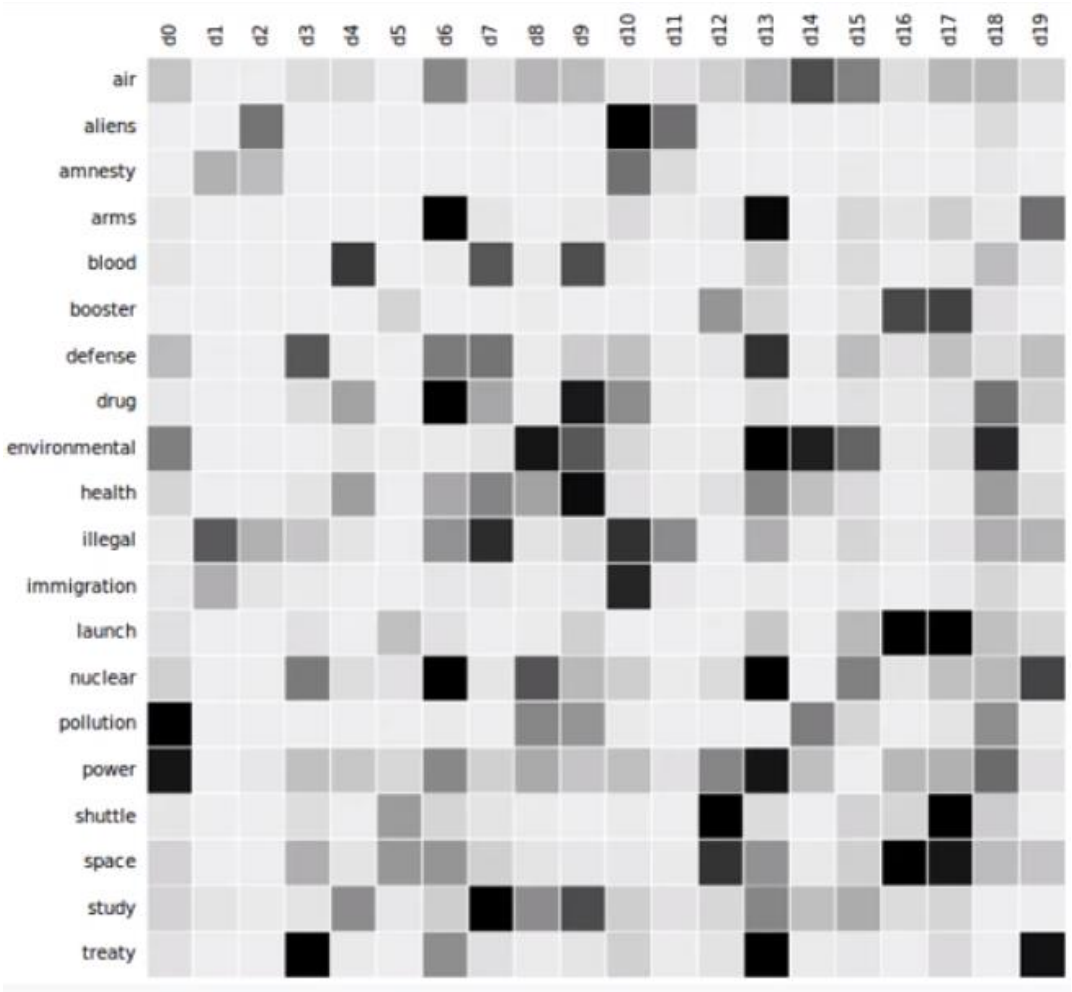


Source: [Nikita Sharma]

Matrix decomposition using SVD does not work well with missing data.
Gradient descent can be used to learn U and V matrices to make movie recommendations.

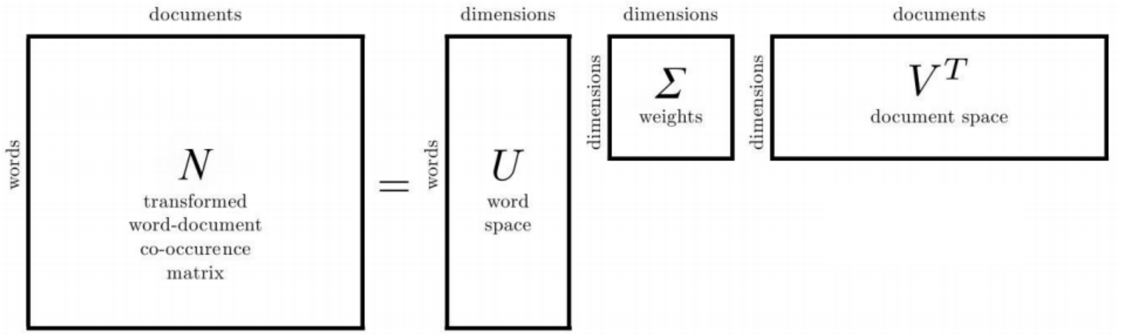
Example Google Colab Code

Application 3 – Text Embeddings



Source: Wikipedia

- Word relationships from documents
  - Latent semantic representation



Example Google Colab Code

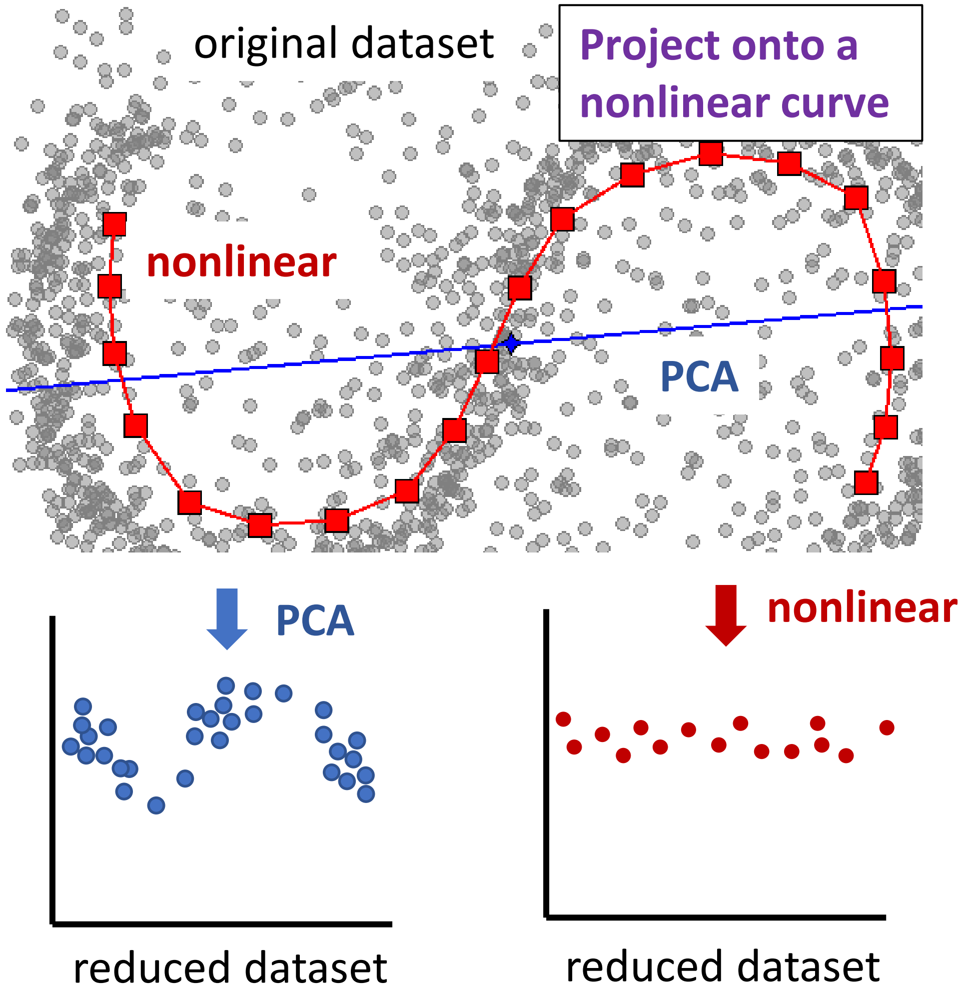
Summary

- Eigendecomposition and Singular Value Decomposition offer ways to factor matrices, much like we can factor numbers into primary numbers.
- This can lead to dimensionality reduction, data insights and algorithm speed ups.
- Eigendecomposition is limited to square matrices, so we use **SVD which is guaranteed to work in all circumstances.**
- **Both apply only for linear transformations/mappings**

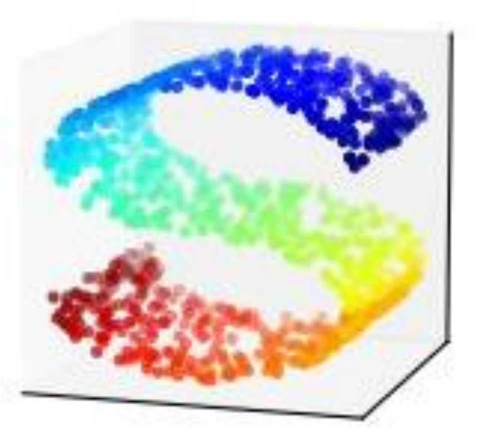
Nonlinear Dimensionality Reduction

- Data can have nonlinear relationships for which our decomposition techniques would be ineffective.

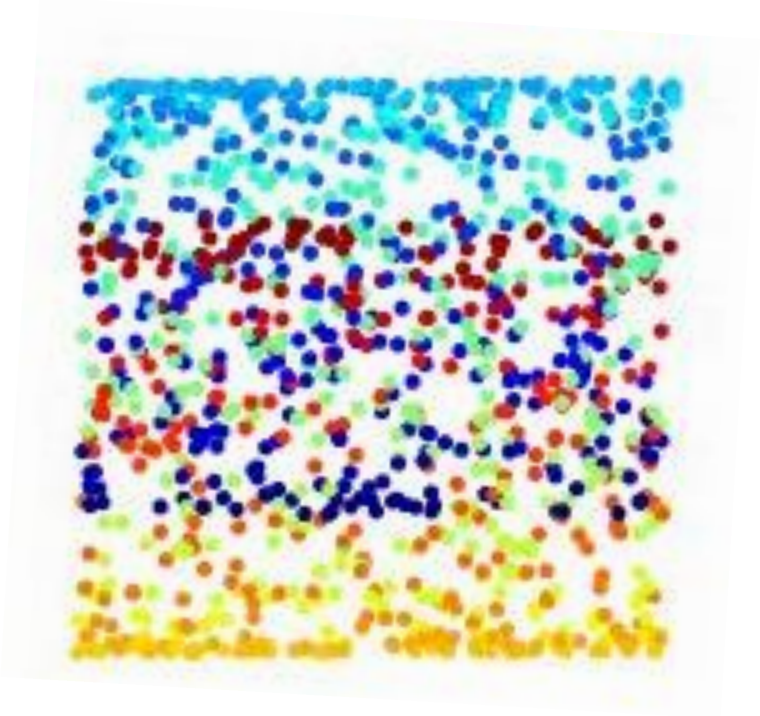
- Data Visualization/Reduction
  - t-SNE
  - Isomap
  - LLE
  - Kernel PCA
  - Deep Autoencoders (type of neural network)



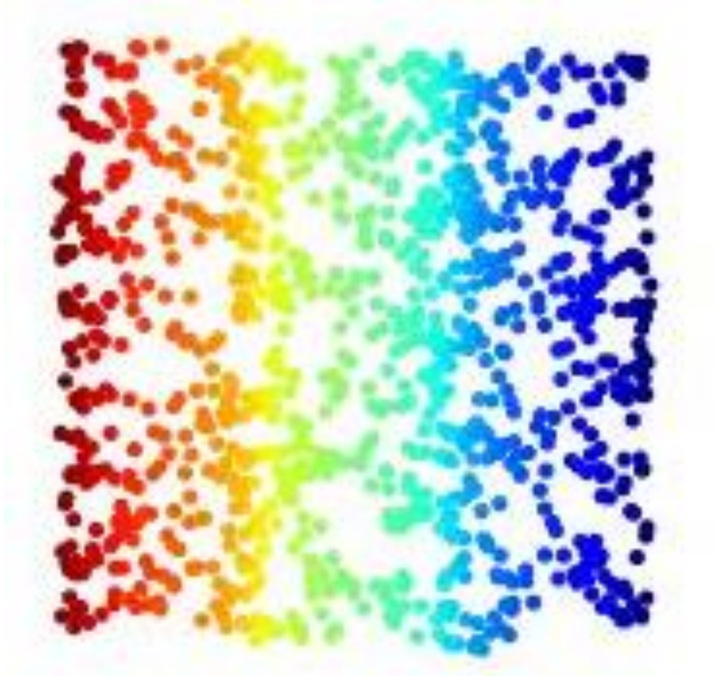
Example: Nonlinear Projections



Nonlinear Data



PCA Projection



Isomap Projection

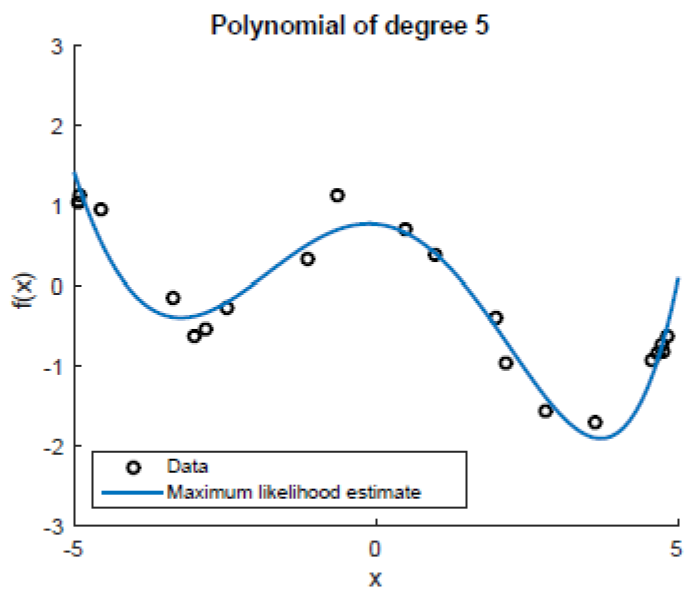
Vector Calculus

Readings:

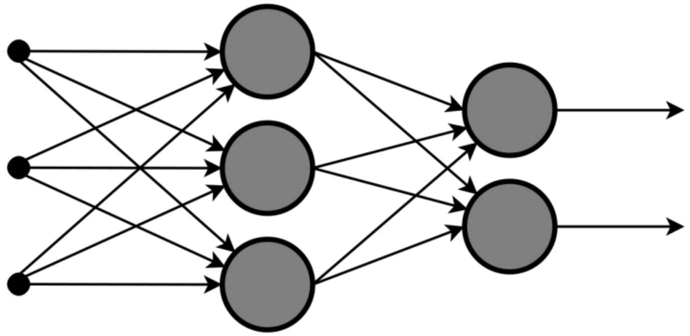
- MML Chapter 5.1-4

Why Vector Calculus?

- Vector calculus is used extensively in optimization
  - We will see it the next couple of lectures when we discuss model-based learning using **linear regression** and **neural network models**



- Most of the python frameworks such as Tensorflow, PyTorch, etc. handle these calculations using numerical methods



Types of Differentiation

1. Scalar differentiation: $f : \mathbb{R} \to \mathbb{R}$

   $y \in \mathbb{R}$ w.r.t. $x \in \mathbb{R}$

2. Multivariate case: $f : \mathbb{R}^N \to \mathbb{R}$

   $y \in \mathbb{R}$ w.r.t. vector $\boldsymbol{x} \in \mathbb{R}^N$

3. Vector fields: $f : \mathbb{R}^N \to \mathbb{R}^M$

   vector $\boldsymbol{y} \in \mathbb{R}^M$ w.r.t. vector $\boldsymbol{x} \in \mathbb{R}^N$

4. General derivatives: $f : \mathbb{R}^{M \times N} \to \mathbb{R}^{P \times Q}$

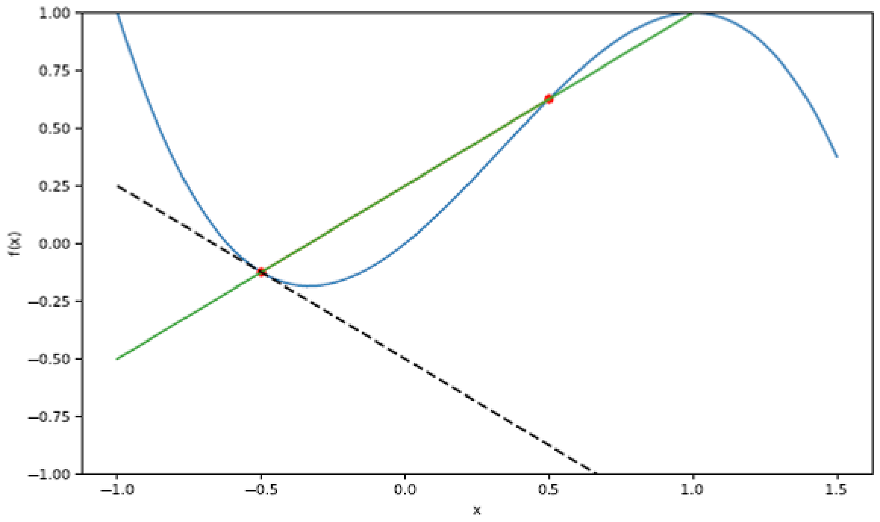
   matrix $\boldsymbol{y} \in \mathbb{R}^{P \times Q}$ w.r.t. matrix $\boldsymbol{X} \in \mathbb{R}^{M \times N}$

1. Scalar Differentiation

- Derivative defined as the limit of the difference quotient:

$$f'(x) = \frac{df}{dx} = \lim_{h \to 0} \frac{f(x+h) - f(x)}{h}$$

- Slope of a secant line through f(x) and f(x+h)



Some Examples

$$
\begin{aligned}
f(x) &= x^n & f'(x) &= nx^{n-1} \\
f(x) &= \sin(x) & f'(x) &= \cos(x) \\
f(x) &= \tanh(x) & f'(x) &= 1 - \tanh^2(x) \\
f(x) &= \exp(x) & f'(x) &= \exp(x) \\
f(x) &= \log(x) & f'(x) &= \frac{1}{x}
\end{aligned}
$$

Rules

- Sum Rule

$$(f(x)+g(x))' = f'(x)+g'(x) = \frac{df(x)}{dx}+\frac{dg(x)}{dx}$$

- Product Rule

$$(f(x)g(x))' = f'(x)g(x)+f(x)g'(x) = \frac{df(x)}{dx}g(x)+f(x)\frac{dg(x)}{dx}$$

- Chain Rule

$$(g\circ f)'(x) = (g(f(x)))' = g'(f(x))f'(x) = \frac{dg(f(x))}{df}\frac{df(x)}{dx}$$

- Quotient Rule

$$\left(\frac{f(x)}{g(x)}\right)' = \frac{f(x)'g(x)-f(x)g(x)'}{(g(x))^2} = \frac{\frac{df}{dx}g(x)-f(x)\frac{dg}{dx}}{(g(x))^2}$$

Example: Scalar Chain Rule

$$(g \circ f)'(x) = (g(f(x)))' = g'(f(x))f'(x) = \frac{dg}{df}\frac{df}{dx}$$

$$\begin{aligned} g(z) &= 6z + 3 \\ z &= f(x) = -2x + 5 \\ (g \circ f)'(x) &= \underbrace{(6)}_{dg/df} \underbrace{(-2)}_{df/dx} \\ &= -12 \end{aligned}$$

2. Multivariate Differentiation f: R^N to R

$$y = f(\boldsymbol{x}),$$

$$\boldsymbol{x} = \begin{bmatrix} x_1 \\ \vdots \\ x_N \end{bmatrix} \in \mathbb{R}^N$$

- Partial derivative (change one coordinate at a time)

$$\frac{\partial f}{\partial x_i} = \lim_{h \to 0} \frac{f(x_1, \ldots, x_{i-1}, x_i + h, x_{i+1}, \ldots, x_N) - f(\boldsymbol{x})}{h}$$

- The gradient collects all partial derivatives:

$$\frac{\mathrm{d}f}{\mathrm{d}\boldsymbol{x}} = \begin{bmatrix} \frac{\partial f}{\partial x_1} & \cdots & \frac{\partial f}{\partial x_N} \end{bmatrix} \in \mathbb{R}^{1 \times N}$$

results in a row vector

Example: Multivariate Differentiation

- Given:

$$f : \mathbb{R}^2 \to \mathbb{R}$$

$$f(x_1, x_2) = x_1^2 x_2 + x_1 x_2^3 \in \mathbb{R}$$

- Partial derivative

$$\frac{\partial f(x_1, x_2)}{\partial x_1} = 2x_1 x_2 + x_2^3$$

$$\frac{\partial f(x_1, x_2)}{\partial x_2} = x_1^2 + 3x_1 x_2^2$$

- Gradient

$$\frac{\mathrm{d}f}{\mathrm{d}\boldsymbol{x}} = \begin{bmatrix} \frac{\partial f(x_1,x_2)}{\partial x_1} & \frac{\partial f(x_1,x_2)}{\partial x_2} \end{bmatrix} \in \mathbb{R}^{1\times 2}$$

$$= \begin{bmatrix} 2x_1 x_2 + x_2^3 & x_1^2 + 3x_1 x_2^2 \end{bmatrix}$$

3. Vector Field Differentiation f: R^N to R^M

$$\boldsymbol{y} = \boldsymbol{f}(\boldsymbol{x}) \in \mathbb{R}^M, \quad \boldsymbol{x} \in \mathbb{R}^N$$

$$\begin{bmatrix} y_1 \\ \vdots \\ y_M \end{bmatrix} = \begin{bmatrix} f_1(\boldsymbol{x}) \\ \vdots \\ f_M(\boldsymbol{x}) \end{bmatrix} = \begin{bmatrix} f_1(x_1, \ldots, x_N) \\ \vdots \\ f_M(x_1, \ldots, x_N) \end{bmatrix}$$

- Jacobian matrix (collection of all partial derivatives)

$$\begin{bmatrix} \frac{dy_1}{dx} \\ \vdots \\ \frac{dy_M}{dx} \end{bmatrix} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \cdots & \frac{\partial f_1}{\partial x_N} \\ \vdots & & \vdots \\ \frac{\partial f_M}{\partial x_1} & \cdots & \frac{\partial f_M}{\partial x_N} \end{bmatrix} \in \mathbb{R}^{M \times N}$$

Example: Vector Field Differentiation

- Given:

$$\boldsymbol{f}(\boldsymbol{x}) = \boldsymbol{A}\boldsymbol{x}\,, \qquad \boldsymbol{f}(\boldsymbol{x}) \in \mathbb{R}^M, \quad \boldsymbol{A} \in \mathbb{R}^{M\times N}, \quad \boldsymbol{x} \in \mathbb{R}^N$$

$$\begin{bmatrix} y_1 \\ \vdots \\ y_M \end{bmatrix} = \begin{bmatrix} f_1(\boldsymbol{x}) \\ \vdots \\ f_M(\boldsymbol{x}) \end{bmatrix} = \begin{bmatrix} A_{11}x_1 + A_{12}x_2 + & \cdots & + A_{1N}x_N \\ \vdots \quad\quad \vdots & \vdots & \vdots \\ A_{M1}x_1 + A_{M2}x_2 + & \cdots & + A_{MN}x_N \end{bmatrix}$$

- Compute the Jacobian

$$f_i(\boldsymbol{x}) = \sum_{j=1}^{N} A_{ij}x_j$$

$$\frac{\partial f_i}{\partial x_j} = A_{ij}$$

$$\frac{\mathrm{d}\boldsymbol{f}}{\mathrm{d}\boldsymbol{x}} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \cdots & \frac{\partial f_1}{\partial x_N} \\ \vdots & & \vdots \\ \frac{\partial f_M}{\partial x_1} & \cdots & \frac{\partial f_M}{\partial x_N} \end{bmatrix} = \begin{bmatrix} A_{11} & \cdots & A_{1N} \\ \vdots & & \vdots \\ A_{M1} & \cdots & A_{MN} \end{bmatrix} = \boldsymbol{A} \in \mathbb{R}^{M\times N}$$

Dimensionality of the Jacobian

- In general: A function $f\colon \mathbb{R}^{\mathrm{N}} \to \mathbb{R}^{\mathrm{M}}$ has a gradient that is an M x N matrix with:

$$\frac{\mathrm{d}f}{\mathrm{d}x} \in \mathbb{R}^{M \times N}, \qquad \mathrm{d}f[m,n] = \frac{\partial f_m}{\partial x_n}$$

Jacobian dimension: #target dimensions x #input dimensions

Example: Chain Rule

- Given $f\colon \mathbb{R}^2 \to \mathbb{R}$, $x\colon \mathbb{R} \to \mathbb{R}^2$, $t \in \mathbb{R}$

$$f(\boldsymbol{x}) = f(x_1, x_2) = x_1^2 + 2x_2\,,$$

$$\boldsymbol{x}(t) = \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix} = \begin{bmatrix} \sin(t) \\ \cos(t) \end{bmatrix}$$

- What are the dimensions of df/dx and dx/dt?

  1 x 2 and 2 x 1

- Compute the gradient df/dt using the chain rule:

$$\frac{\mathrm{d}f}{\mathrm{d}t} = \frac{\mathrm{d}f}{\mathrm{d}\boldsymbol{x}}\frac{\mathrm{d}\boldsymbol{x}}{\mathrm{d}t} = \begin{bmatrix} \frac{\partial f}{\partial x_1} & \frac{\partial f}{\partial x_2} \end{bmatrix} \begin{bmatrix} \frac{\partial x_1}{\partial t} \\ \frac{\partial x_2}{\partial t} \end{bmatrix} = \begin{bmatrix} 2\sin t & 2 \end{bmatrix} \begin{bmatrix} \cos t \\ -\sin t \end{bmatrix}$$

$$= 2\sin t\cos t - 2\sin t = 2\sin t(\cos t - 1)$$

4. Derivatives with Respect to Matrices

- Recall: A function $f\colon \mathbb{R}^{\mathrm{N}} \rightarrow \mathbb{R}^{\mathrm{M}}$ has a gradient that is an M x N matrix with:

$$\frac{\mathrm{d}\boldsymbol{f}}{\mathrm{d}\boldsymbol{x}} \in \mathbb{R}^{M \times N}, \qquad \mathrm{d}\boldsymbol{f}[m,n] = \frac{\partial f_m}{\partial x_n}$$

Gradient dimension: #target dimensions x #input dimensions

- **This generalizes to when the inputs (N) or targets (M) are matrices.**

- Function $f\colon \mathbb{R}^{\mathrm{MxN}} \rightarrow \mathbb{R}^{\mathrm{PxQ}}$ has a gradient that is a (P x Q) x (M x N) **tensor**:

$$\frac{\mathrm{d}\boldsymbol{f}}{\mathrm{d}\boldsymbol{X}} \in \mathbb{R}^{(P \times Q) \times (M \times N)}, \qquad \mathrm{d}\boldsymbol{f}[p,q,m,n] = \frac{\partial f_{pq}}{\partial X_{mn}}$$

More Examples:

Example:

- Given: $f(x) = Ax$, A = $\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}$, $\mathrm{x} = [x_1\ x_2]^T$, find df(x)/dx

Example:

- Given: $f(x) = x^T Ax$, A = $\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}$, $x = [x_1\ x_2]^T$, find df(x)/dx

PCA by derivation

- Requires some familiarity with vector calculus for matrix differentiation
- **Suggested reading:** [Deep Learning Textbook (pgs 45 – 50)] by Ian Goodfellow, Yoshua Bengio and Aaron Courville.

Next Time

- Week 8 Q/A Support Session: Project 3 Support
- Project 3 is due on 13 March at 23:00 (extended deadline)
- Guest Lecture on 15 March at 10:00
  - Dr. Sophie Lohmann: “Limits of measurement - who are we measuring?”
  - Zoom link https://utoronto.zoom.us/j/86722516215
- Week 9 Lecture – Linear Regression
  - Monte Carlo Simulation
  - Empirical Risk Minimization
  - Maximum Likelihood
  - Probabilistic Modelling and Inference