APS1070

Foundations of Data Analytics and Machine Learning

Winter 2022
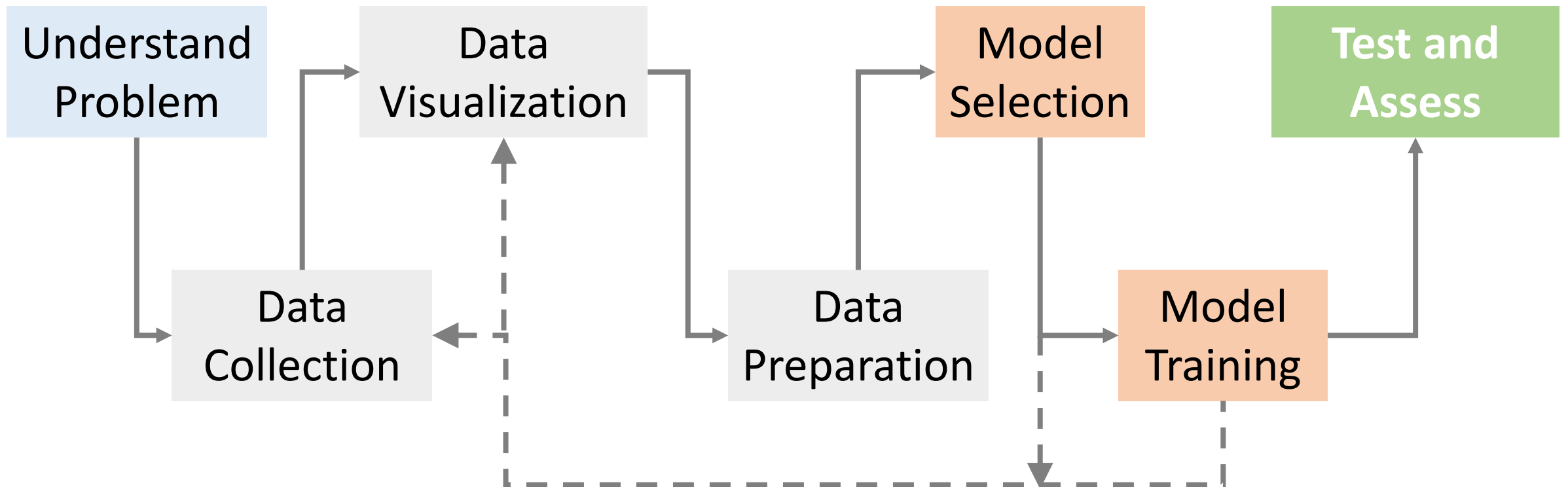
Week 8:
- *Singular Value Decomposition (SVD)*
- *Applications*
- *Vector Calculus*

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 can we perform matrix decomposition on rectangular matrices?

Motivation: Movie Recommendations

➢ SVD can also be used to find structure in data for making recommendations on movies.
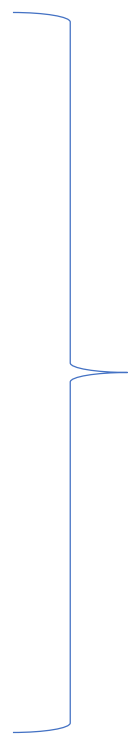
➢ More on this later…

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

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

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

Agenda

- ➤ Recap from last time
  - ➤ Eigendecomposition
  - ➤ PCA
- ➤ 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: Matrix Decomposition

- ❑ Eigenvectors
- ❑ Eigenvalues
- ❑ Determinants
- ❑ Eigendecomposition
- ❑ PCA
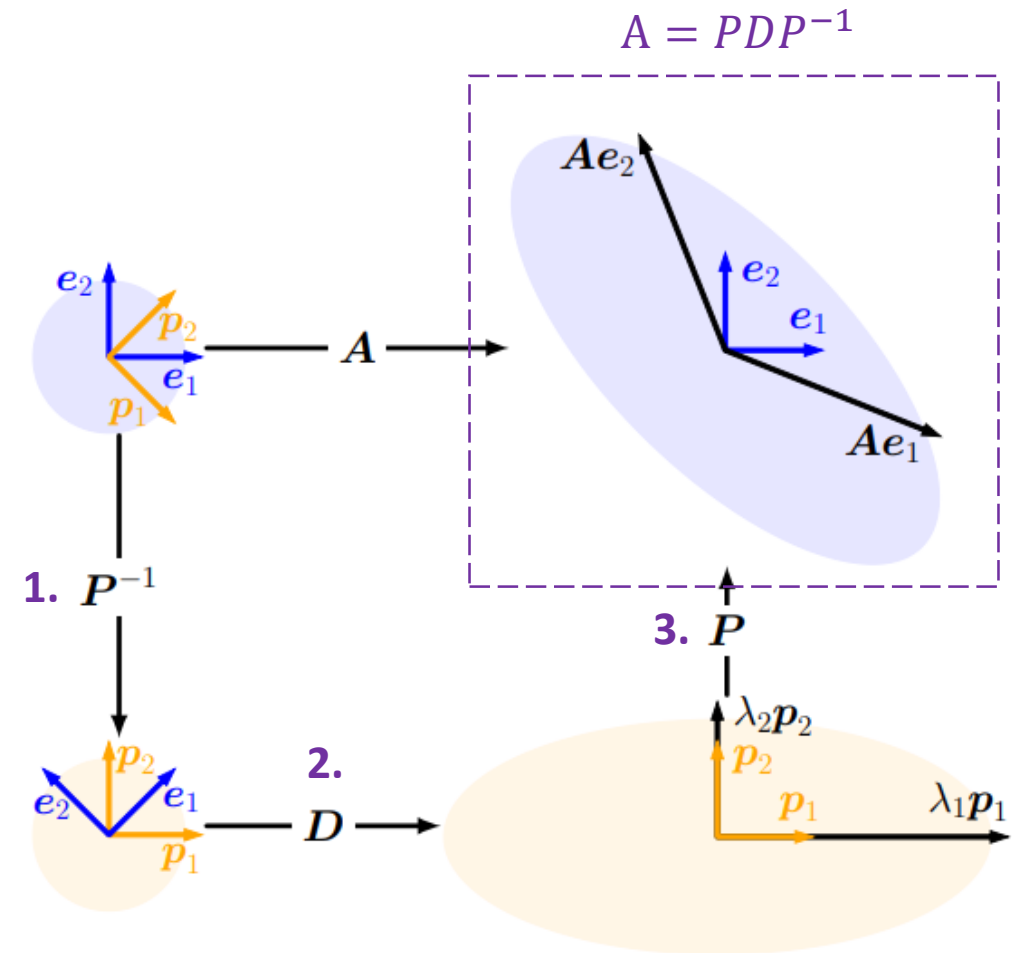
Recap: Eigendecomposition

➢ A **square matrix** (n x n) can be factored into:

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

➢ If the square **matrix is** also **symmetrical** then the eigenvectors will also be orthonormal:

$$A = PDP^T$$

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:

A = Q X Λ X Q^T

m X m   m X m   m X m   m X m

eigenvector matrix   eigenvalue matrix

14

Recap: Matrix Decomposition

➢ We would like a general approach for **decomposing rectangular matrices**.

➢ PCA is already applied to rectangular matrices…

➢ Mathematically PCA is:

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

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

Singular Value Decomposition
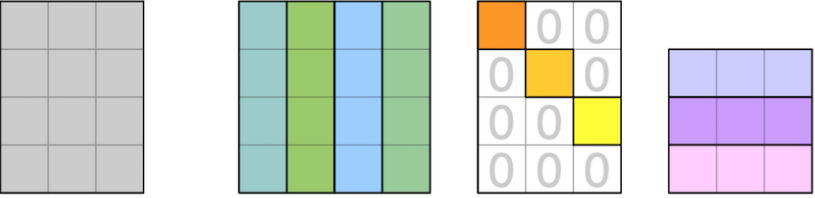
➢ We would like to obtain the following decomposition for a **rectangular matrix**:

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

U is referred to as the left-singular vector,

V is the right-singular vector,

Σ are the singular values (increasing order)

Singular Value Decomposition

$$X = U \Sigma V^*$$
$$n \times m \quad n \times n \quad n \times m \quad m \times m$$
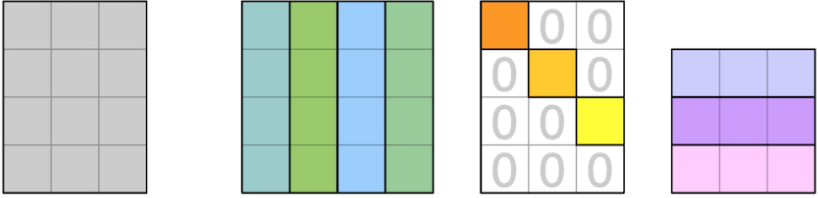
➤ Singular values matrix Σ has additional zero padding of rows or columns:

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

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 $m < n$

Singular Value Decomposition

$$A_{n \times m} = U_{n \times n} \; \Sigma_{n \times m} \; V^{*}_{m \times m}$$

	Ali	Beatrix	Chandra
Star Wars	5	4	1
Blade Runner	5	5	0
Amelie	0	0	5
Delicatessen	1	0	4

Sci-fi	French		
−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

9.6438	0	0
0	6.3639	0
0	0	0.7056
0	0	0

	Sci-fi		
Sci-fi	−0.7367	−0.6515	−0.1811
French	0.0852	0.1762	−0.9807
	0.6708	−0.7379	−0.0743

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

More on this later

SVD Algorithm

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

➢ Right-singular vectors:

$$X^T X = VDV^T$$

➢ Left-singular vectors:

$$XX^T = UDU^T$$

➢ XX^T and $X^T X$ are symmetrical

➢ SVD does not require that we normalized X to have zero mean

SVD Algorithm

➢ Right-Singular:

$$X^T X$$

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

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

$$= V\Sigma^2 V^T$$

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

➢ Left-Singular:

$$XX^T$$

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

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

$$= U\Sigma^2 U^T$$

eigendecomposition of XX^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:

$$AV = U\Sigma$$

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

$\min(m, n)$

Example 4.13

Compute the singular value decomposition for matrix A.

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

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

$A = U\Sigma V^T$

notice new dimension!
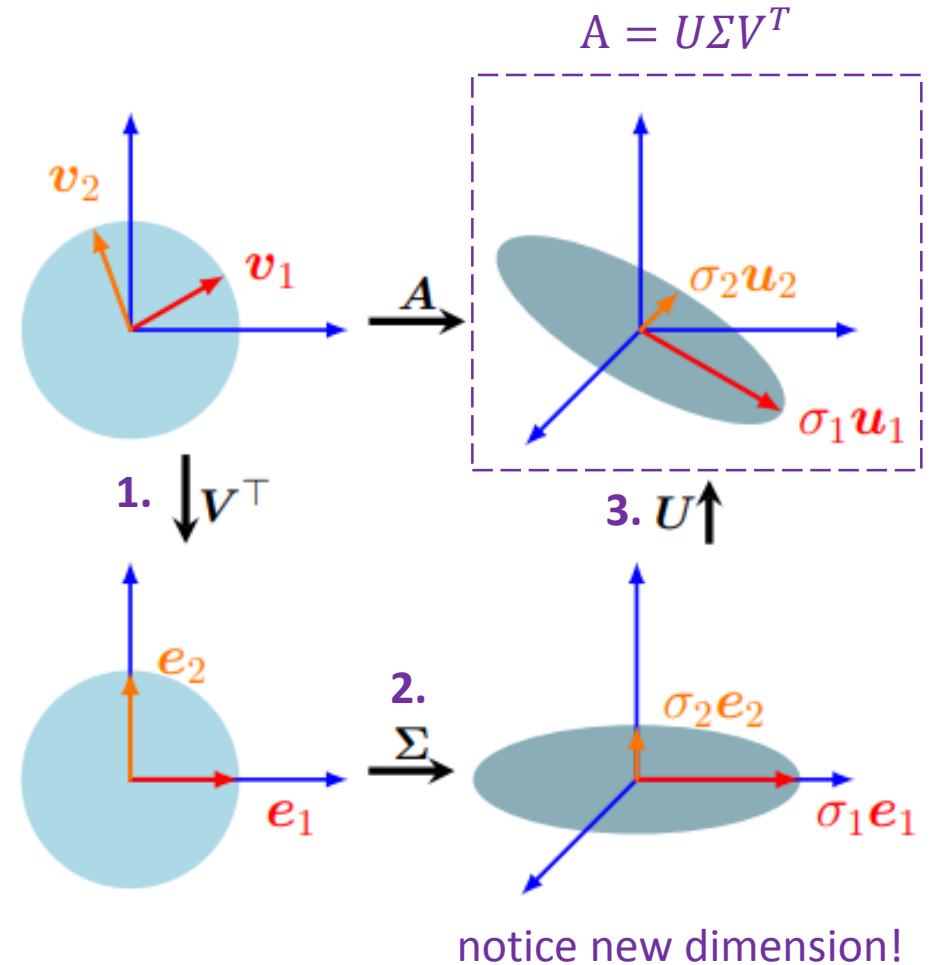
Example

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

$$A = \begin{bmatrix} 1 & -0.8 \\ 0 & 1 \\ 1 & 0 \end{bmatrix} = U\Sigma V^\top$$

$$= \begin{bmatrix} -0.79 & 0 & -0.62 \\ 0.38 & -0.78 & -0.49 \\ -0.48 & -0.62 & 0.62 \end{bmatrix} \begin{bmatrix} 1.62 & 0 \\ 0 & 1.0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} -0.78 & 0.62 \\ -0.62 & -0.78 \end{bmatrix}$$

X -> set of vectors

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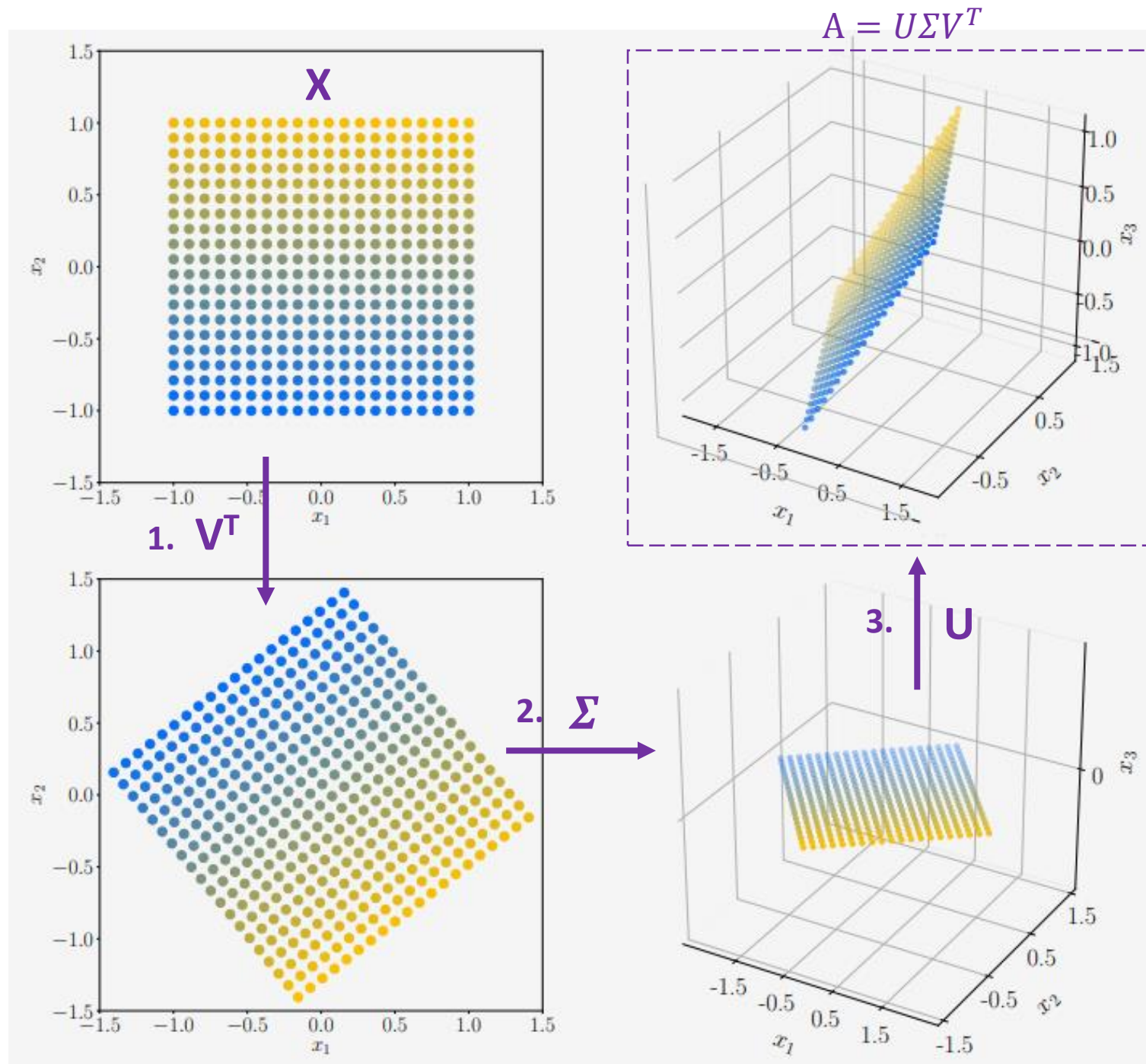
Similarities to Eigendecomposition -1

➢ 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
$$X = PDP^{-1}$$

Singular Value Decomposition
$$X = U\Sigma V^T$$

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^{T}A
  ➢ The nonzero singular values of A are the square roots of nonzero eigenvalues of AA^T and A^{T}A

➢ **For symmetric square matrices (n x n) eigendecomposition and SVD are the same when features are normalized**

Eigendecomposition
$$X = PDP^{-1}$$

Singular Value Decomposition
$$X = U\Sigma V^T$$

$$\longrightarrow \quad P = U = V$$

Relationship between PCA and SVD

➢ PCA and SVD are closely related:

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

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

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

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

SVD columns of V are principal direction in PCA

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

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

eigenvalues   **singular values**

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

$X =$

$pixel_1$
$pixel_2$

$pixel_k$

$face_1$ $face_2$ $face_d$

3.

The Covariance Matrix

$$Cov(f) = f\,f^T = \begin{bmatrix} f_i^2 & f_jf_i \\ f_if_j & \end{bmatrix}$$

4.

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

z_3 z_2 PC_1
PC_2
PC_3
z_1

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

5.

6.

Image Compression

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

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

$$A = \sum_{i=1}^{r} \sigma_i u_i v_i^{\top} = \sum_{i=1}^{r} \sigma_i A_i$$
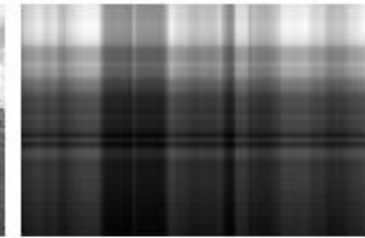
(a) Original image A. (b) Rank-1 approximation $\widehat{A}(1)$. (c) Rank-2 approximation $\widehat{A}(2)$.
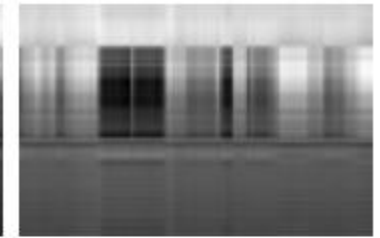
(d) Rank-3 approximation $\widehat{A}(3)$. (e) Rank-4 approximation $\widehat{A}(4)$. (f) Rank-5 approximation $\widehat{A}(5)$.

Application 1 – Background Removal

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

Singe frame from video

Reconstruction from SVD (low rank)

Reconstruction from SVD (original matrix – low rank)

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
  ➤ …

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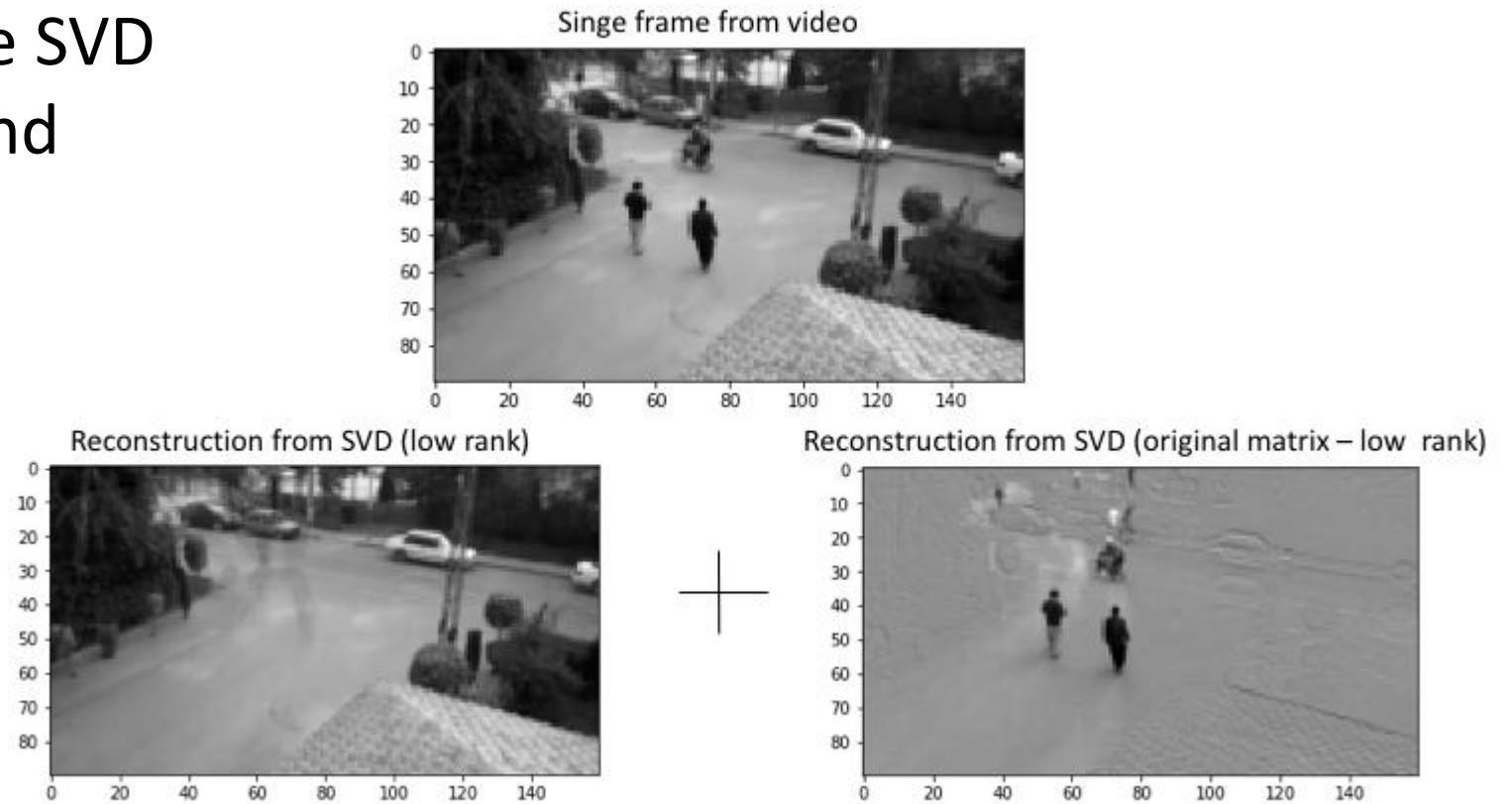\begin{array}{r}
\text{Star Wars} \\
\text{Blade Runner} \\
\text{Amelie} \\
\text{Delicatessen}
\end{array}
\begin{bmatrix}
5 & 4 & 1 \\
5 & 5 & 0 \\
0 & 0 & 5 \\
1 & 0 & 4
\end{bmatrix}
\end{array}
$$

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

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

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

Example Google Colab Code

Application 3 – Text Embeddings

Source: Wikipedia

- Word relationships from documents
  - Latent semantic representation
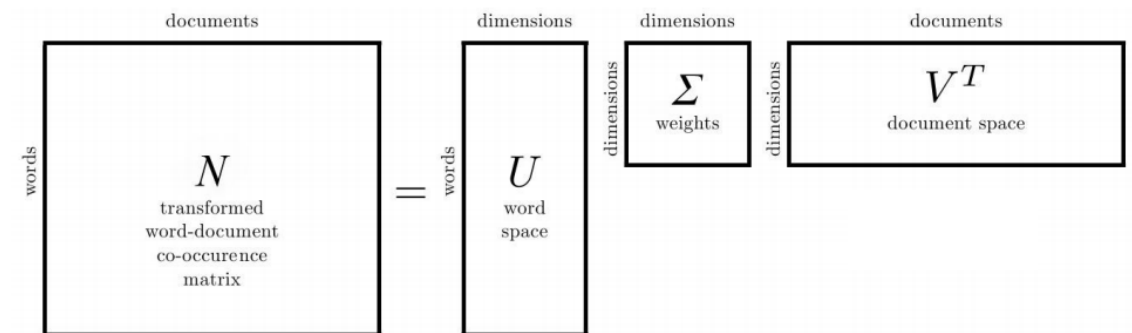
Example Google Colab Code

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.**

➢ **(limitation) Both apply linear transformations/mappings**

Nonlinear Dimensionality Reduction

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

➢ Nonlinear Data Visualization/Reduction

  ➢ t-SNE

  ➢ Isomap

  ➢ LLE

  ➢ Kernel PCA

  ➢ Deep Autoencoders

    (type of neural network)

original dataset

Project onto a nonlinear curve

nonlinear

PCA

PCA

nonlinear

reduced dataset

reduced dataset
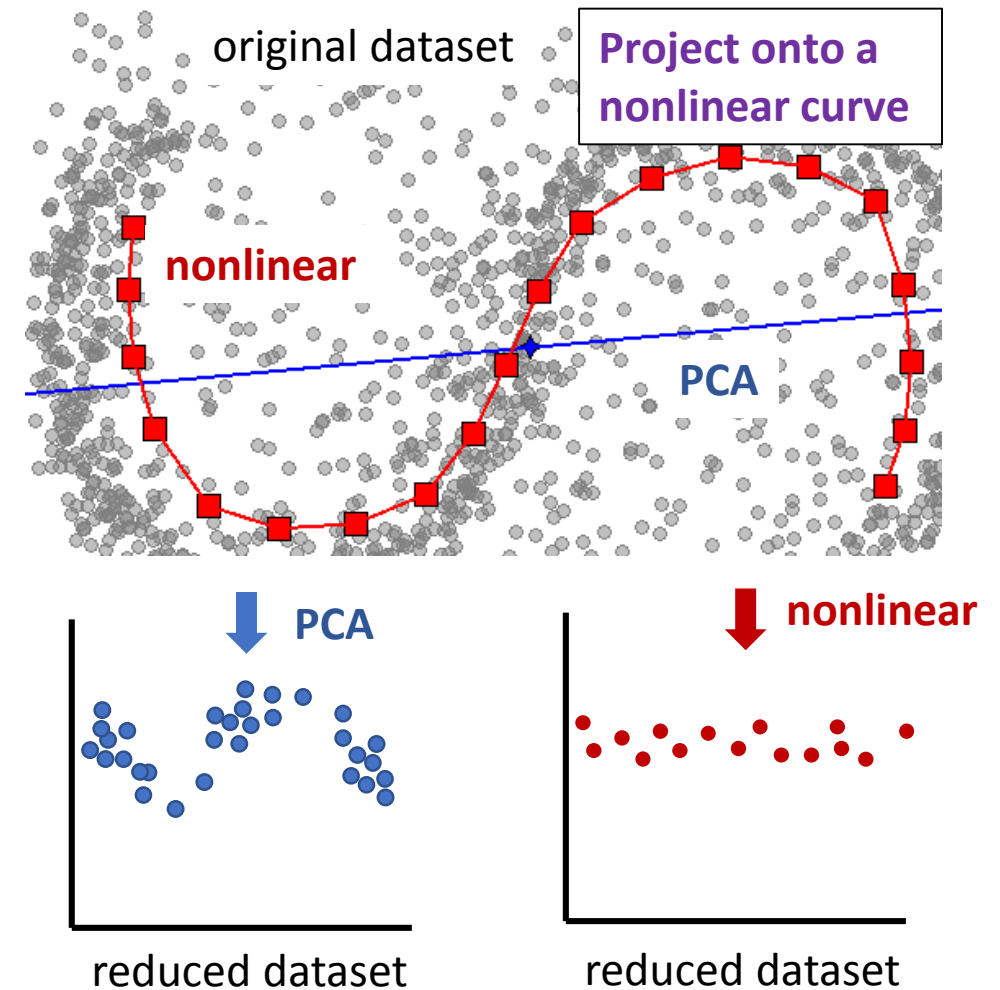
Example: Nonlinear Projections
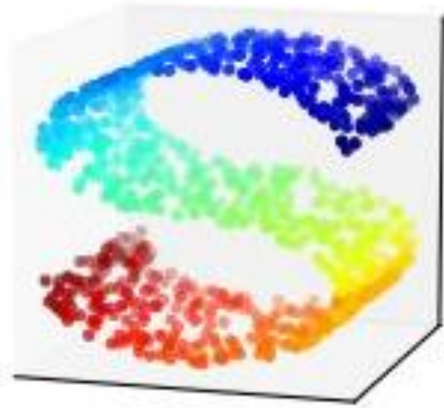
Nonlinear Data
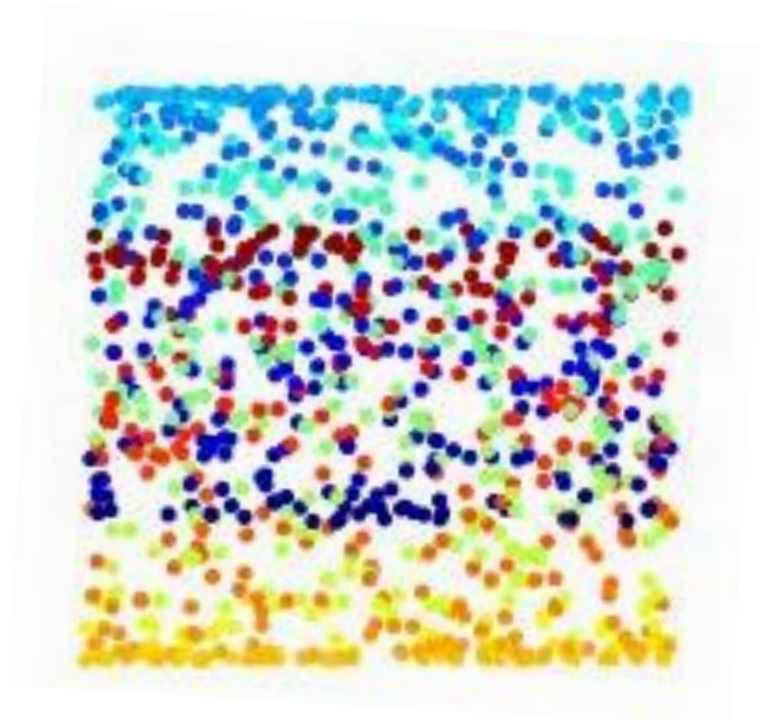
PCA Projection
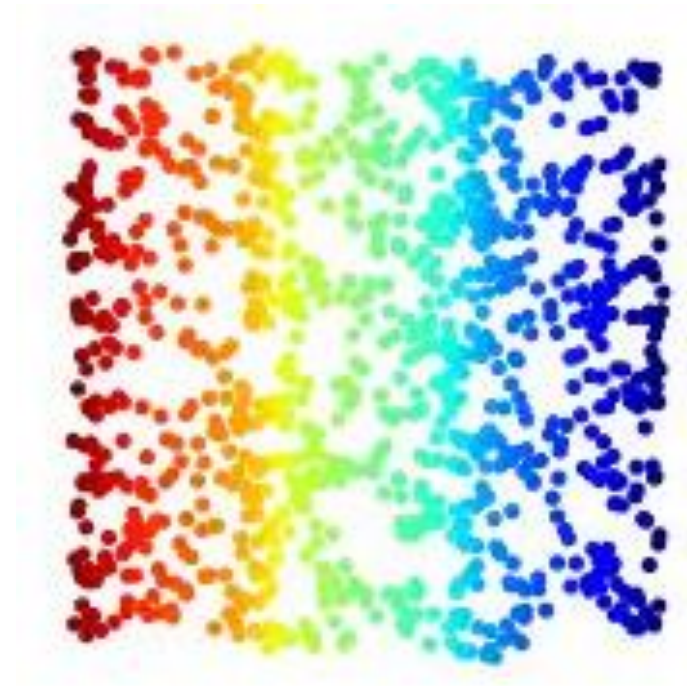
Isomap Projection

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 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
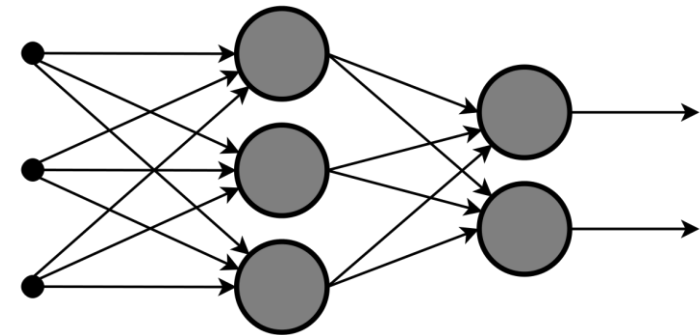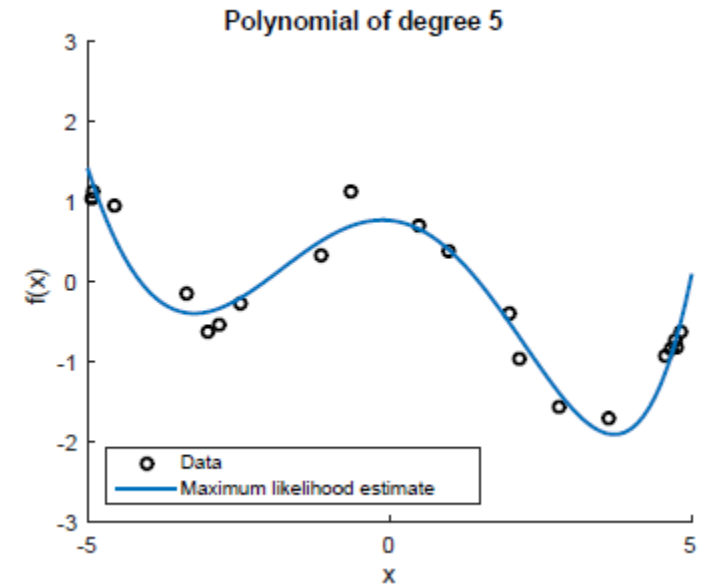
Types of Differentiation

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

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

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

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

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

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

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

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

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)
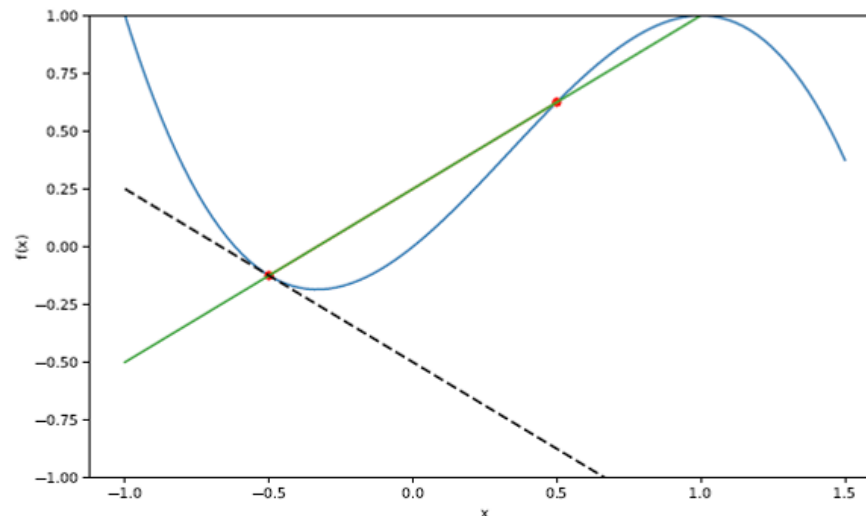
Some Examples

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

Rules

- Sum Rule

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

- Product Rule

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

- Chain Rule

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

- Quotient Rule

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

64

Example: Scalar Chain Rule

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

$$g(z) = 6z + 3$$

$$z = f(x) = -2x + 5$$

$$(g \circ f)'(x) = \underbrace{(6)}_{dg/df} \underbrace{(-2)}_{df/dx}$$

$$= -12$$

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

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

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

➤ 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

$$\frac{\mathrm{d}f}{\mathrm{d}\boldsymbol{x}} = \left[\frac{\partial f(x_1, x_2)}{\partial x_1} \quad \frac{\partial f(x_1, x_2)}{\partial x_2} \right] \in \mathbb{R}^{1 \times 2}$$

$$= \left[2x_1 x_2 + x_2^3 \quad x_1^2 + 3x_1 x_2^2 \right]$$

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

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

➤ Jacobian matrix (collection of all partial derivatives)

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

Example: Vector Field Differentiation

➤ Given:

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

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

➤ Compute the Jacobian

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

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

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

Dimensionality

➢ In general: A function $f: \mathbb{R}^{\text{N}} \rightarrow \mathbb{R}^{\text{M}}$ has a result that is an M x N matrix with:

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

Gradient dimension: #target dimensions x #input dimensions

Example: Chain Rule

➢ Given $f\colon \mathbb{R}^2 \to \mathbb{R},\ x\colon \mathbb{R} \to \mathbb{R}^2$

$$f(x) = f(x_1, x_2) = x_1^2 + 2x_2,$$

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

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

      1 x 2 and 2 x 1

➢ Compute df/dt using the chain rule:

$$\frac{df}{dt} = \frac{df}{dx}\frac{dx}{dt} = \begin{bmatrix} \frac{\partial f}{\partial x_1} & \frac{\partial f}{\partial x_2} \end{bmatrix} \begin{bmatrix} \frac{\partial x_1}{\partial t} \\ \frac{\partial x_2}{\partial t} \end{bmatrix} = \begin{bmatrix} 2\sin t & 2 \end{bmatrix} \begin{bmatrix} \cos t \\ -\sin t \end{bmatrix}$$

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

4. Derivatives with Respect to Matrices

➤ Recall: A function $f: \mathbb{R}^N \to \mathbb{R}^M$ has a Jacobian that is an M x N matrix with:

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

Gradient dimension: #target dimensions x #input dimensions

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

➤ Function $f: \mathbb{R}^{M \times N} \to \mathbb{R}^{P \times Q}$ has a Jacobian that is a (P x Q) x (M x N) **tensor**:

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

More Examples:

Example:

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

➤ Find df(x)/dx

Example:

➢ Given: $f(x) = x^T A x,$ A = $\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix},$ x = $[x_1 \; x_2]^T$

➢ Find df(x)/dx

Example: PCA Derivation

➢ Requires some familiarity with vector calculus for matrix differentiation

➢ There are three perspectives:

1. Maximum Variance Perspective

2. Projection Perspective

3. Latent Variable Perspective

➢ Lead to the same result using different derivations.

➢ **Suggested derivation:** Deep Learning Textbook (pgs 45 – 50) by Ian Goodfellow, Yoshua Bengio and Aaron Courville.

Next Time

- Week 8 Support Session
  - Project 3 Support

- Reading Assignment

- Lecture – Linear Regression
  - Data, Models and Learning
  - Empirical Risk Minimization
  - Maximum Likelihood
  - Probabilistic Modelling and Inference