Introduction to Machine Learning

Lecture 6 Representation and Clustering - (Linear) Dimensionality Reduction

Hongteng Xu

中國人民大學 RENMIN UNIVERSITY OF CHINA

高瓴人工智能学院 Gaoling School of Artificial Intelligence

Outline

Review

- **Non-linear regression:** parametric or nonparametric
- **Kernel** regression, RKHS, and representer theorem
- **Gaussian process:** Definition and basic formulation

Outline

Review

- **Non-linear regression:** parametric or nonparametric
- **Kernel** regression, RKHS, and representer theorem
- **Gaussian process:** Definition and basic formulation

Today

- Curse of dimensionality
- Principal component analysis (PCA)
- Linear projection and factorization model

Dimension of Sample Space

- Could you enumerate the methods/models **increasing** sample dimensions?

Dimension of Sample Space

- Could you enumerate the methods/models **increasing** sample dimensions?
    - Polynomial regression
    - Kernel method

Dimension of Sample Space

- Could you enumerate the methods/models **increasing** sample dimensions?
  - Polynomial regression
  - Kernel method
- Pros and Cons?

Dimension of Sample Space

- Could you enumerate the methods/models **increasing** sample dimensions?
  - Polynomial regression
  - Kernel method
- Pros and Cons?
  - Nonlinear mechansims, powerful models

Dimension of Sample Space

- Could you enumerate the methods/models **increasing** sample dimensions?
  - Polynomial regression
  - Kernel method
- Pros and Cons?
  - Nonlinear mechansims, powerful models
  - Risk of over-fitting
  - Sensitivity to noise
  - High computational complexity

Curse of Dimensionality

- First proposed by Richard Bellman when he studied dynamic programming.

Curse of Dimensionality

- First proposed by Richard Bellman when he studied dynamic programming.
- **Phenomenon:** When the dimensionality increases, the volume of the space increases so fast that the available data become sparse.
- Typically, the amount of required data often grows exponentially with the dimensionality.

Curse of Dimensionality: Examples

Combinatorial explosion (discrete examples)

- The number of different d-dimensional binary vectors is 2^d.
- Each additional dimension doubles the effort needed to try all combinations, e.g., the increase of search space of Chess/Go

Curse of Dimensionality: Examples

Combinatorial explosion (discrete examples)

- The number of different d-dimensional binary vectors is 2^d.
- Each additional dimension doubles the effort needed to try all combinations, e.g., the increase of search space of Chess/Go

High-dimensional sampling (continuous examples)

- For a d-dimensional unit hypercube with a lattice that has a spacing of 10^{-1} between adjacency points would require 10^d sample points.

Curse of Dimensionality: Examples

Combinatorial explosion (discrete examples)

- The number of different d-dimensional binary vectors is 2^d.
- Each additional dimension doubles the effort needed to try all combinations, e.g., the increase of search space of Chess/Go

High-dimensional sampling (continuous examples)

- For a d-dimensional unit hypercube with a lattice that has a spacing of 10^{-1} between adjacency points would require 10^d sample points.
- Sampling N samples randomly from a d-dimensional sample space, based on a distribution with identity covariance matrix:

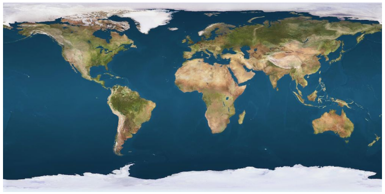
$$\lim_{d \to \infty} \mathbb{E}\left[\frac{D_{\max}(d) - D_{\min}(d)}{D_{\min}(d)}\right] \to 0. \tag{1}$$

Distance functions are losing their usefulness in high dimensions.

How to Suppress This Issue? Dimensionality Reduction

3D data

Linear map to 2D space


Nonlinear map to 2D space.

Linear Dimensionality Reduction

- Original high-dimensional sample space, i.e., $\boldsymbol{x} \in \mathcal{X} \subset \mathbb{R}^D$.
- We would like to find a linear projection $f : \mathcal{X} \mapsto \mathcal{Z} \subset \mathbb{R}^L$, where $L \ll D$.
- Generally, $\boldsymbol{z} = f(\boldsymbol{x}) = \boldsymbol{U}^T \boldsymbol{x}$, where $\boldsymbol{U} \in \mathbb{R}^{D \times L}$.

Linear Dimensionality Reduction

- Original high-dimensional sample space, i.e., $\boldsymbol{x} \in \mathcal{X} \subset \mathbb{R}^D$.
- We would like to find a linear projection $f : \mathcal{X} \mapsto \mathcal{Z} \subset \mathbb{R}^L$, where $L \ll D$.
- Generally, $\boldsymbol{z} = f(\boldsymbol{x}) = \boldsymbol{U}^T\boldsymbol{x}$, where $\boldsymbol{U} \in \mathbb{R}^{D \times L}$.

How to obtain the projection? What projection is desired?

Linear Dimensionality Reduction

- Original high-dimensional sample space, i.e., $\boldsymbol{x} \in \mathcal{X} \subset \mathbb{R}^D$.
- We would like to find a linear projection $f : \mathcal{X} \mapsto \mathcal{Z} \subset \mathbb{R}^L$, where $L \ll D$.
- Generally, $\boldsymbol{z} = f(\boldsymbol{x}) = \boldsymbol{U}^T\boldsymbol{x}$, where $\boldsymbol{U} \in \mathbb{R}^{D\times L}$.

How to obtain the projection? What projection is desired?

- **Minimizing reconstruction error**

$$\exists g : \mathcal{Z} \mapsto \mathcal{X},\ \boldsymbol{x} \approx g(f(\boldsymbol{x})).$$

Linear Dimensionality Reduction

- Original high-dimensional sample space, i.e., $\boldsymbol{x} \in \mathcal{X} \subset \mathbb{R}^D$.
- We would like to find a linear projection $f : \mathcal{X} \mapsto \mathcal{Z} \subset \mathbb{R}^L$, where $L \ll D$.
- Generally, $\boldsymbol{z} = f(\boldsymbol{x}) = \boldsymbol{U}^T \boldsymbol{x}$, where $\boldsymbol{U} \in \mathbb{R}^{D \times L}$.

How to obtain the projection? What projection is desired?

- **Minimizing reconstruction error**

$$\exists g : \mathcal{Z} \mapsto \mathcal{X},\ \boldsymbol{x} \approx g(f(\boldsymbol{x})). \tag{2}$$

- (Equivalently,) **Maximizing mutual information** (or minimizing information loss)

Linear Dimensionality Reduction

- Original high-dimensional sample space, i.e., $\boldsymbol{x} \in \mathcal{X} \subset \mathbb{R}^D$.
- We would like to find a linear projection $f : \mathcal{X} \mapsto \mathcal{Z} \subset \mathbb{R}^L$, where $L \ll D$.
- Generally, $\boldsymbol{z} = f(\boldsymbol{x}) = \boldsymbol{U}^T\boldsymbol{x}$, where $\boldsymbol{U} \in \mathbb{R}^{D \times L}$.

How to obtain the projection? What projection is desired?

- **Minimizing reconstruction error**

$$\exists g : \mathcal{Z} \mapsto \mathcal{X},\ \boldsymbol{x} \approx g(f(\boldsymbol{x})). \tag{2}$$

- (Equivalently,) **Maximizing mutual information** (or minimizing information loss)
- **(Nearly) Isometry:**

$$d_{\mathcal{Z}}(f(\boldsymbol{x}), f(\boldsymbol{x}')) \approx d_{\mathcal{X}}(\boldsymbol{x}, \boldsymbol{x}'). \tag{3}$$

Dimensionality Reduction by The Methods We Learned

Dimensionality Reduction by The Methods We Learned

Lasso-based Supervised Feature Selection

- Solve the following problem:

$$\min_{\boldsymbol{w}} \|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{w}\|_2^2 + \lambda\|\boldsymbol{w}\|_1. \tag{4}$$

- For $\boldsymbol{X} = [\boldsymbol{x}_1, ..., \boldsymbol{x}_D]$, the column $\boldsymbol{x}_d$ contributes to the estimation of $\boldsymbol{y}$ iff $w_d \neq 0$.

Dimensionality Reduction by The Methods We Learned

Lasso-based Supervised Feature Selection

- Solve the following problem:

$$\min_{\boldsymbol{w}} \|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{w}\|_2^2 + \lambda\|\boldsymbol{w}\|_1. \tag{4}$$

- For $\boldsymbol{X} = [\boldsymbol{x}_1, ..., \boldsymbol{x}_D]$, the column $\boldsymbol{x}_d$ contributes to the estimation of $\boldsymbol{y}$ iff $w_d \neq 0$.
- As a result, the useful features are

$$\widehat{\boldsymbol{X}} = [\boldsymbol{x}_d]_{d:w_d \neq 0} \in \mathbb{R}^{N \times L}, \quad L < D.$$

Dimensionality Reduction by The Methods We Learned

Lasso-based Supervised Feature Selection

- Solve the following problem:

$$\min_{\boldsymbol{w}} \|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{w}\|_2^2 + \lambda\|\boldsymbol{w}\|_1. \tag{4}$$

- For $\boldsymbol{X} = [\boldsymbol{x}_1, ..., \boldsymbol{x}_D]$, the column $\boldsymbol{x}_d$ contributes to the estimation of $\boldsymbol{y}$ iff $w_d \neq 0$.
- As a result, the useful features are

$$\widehat{\boldsymbol{X}} = [\boldsymbol{x}_d]_{d:w_d\neq 0} \in \mathbb{R}^{N\times L}, \quad L < D. \tag{5}$$

How to deal with the case without supervised signal $\boldsymbol{y}$?

Principal Component Analysis (PCA)

- Invented in 1901 by Karl Pearson.

Principal Component Analysis (PCA)

- Invented in 1901 by Karl Pearson.
- Reinvented independently by many researchers.
- **Implementations:** Assume $\boldsymbol{X}^T \mathbf{1}_N = \mathbf{0}_D$ (column-wise zero mean)
    - Singular value decomposition of $\boldsymbol{X} \in \mathbb{R}^{N \times D}$
    - Eigenvalue decomposition of $\boldsymbol{X}^T \boldsymbol{X} \in \mathbb{R}^{D \times D}$

Principal Component Analysis (PCA)

- Invented in 1901 by Karl Pearson.
- Reinvented independently by many researchers.
- **Implementations:** Assume $\boldsymbol{X}^T \mathbf{1}_N = \mathbf{0}_D$ (column-wise zero mean)
  - Singular value decomposition of $\boldsymbol{X} \in \mathbb{R}^{N \times D}$
  - Eigenvalue decomposition of $\boldsymbol{X}^T \boldsymbol{X} \in \mathbb{R}^{D \times D}$
- **Principals components of a collection of data points:**
  - L orthonormal vectors (an orthonormal basis) in the sample space $\mathcal{X} \subset \mathbb{R}^D$

$$\boldsymbol{V} = [\boldsymbol{v}_\ell] \in \mathbb{R}^{D \times L}, \quad \boldsymbol{V}^T \boldsymbol{V} = \boldsymbol{I}_L, \quad L \leq D \tag{6}$$

Principal Component Analysis (PCA)

- Invented in 1901 by Karl Pearson.
- Reinvented independently by many researchers.
- **Implementations:** Assume $\boldsymbol{X}^T \mathbf{1}_N = \mathbf{0}_D$ (column-wise zero mean)
  - Singular value decomposition of $\boldsymbol{X} \in \mathbb{R}^{N \times D}$
  - Eigenvalue decomposition of $\boldsymbol{X}^T \boldsymbol{X} \in \mathbb{R}^{D \times D}$
- **Principals components of a collection of data points:**
  - L orthonormal vectors (an orthonormal basis) in the sample space $\mathcal{X} \subset \mathbb{R}^D$

$$\boldsymbol{V} = [\boldsymbol{v}_\ell] \in \mathbb{R}^{D \times L}, \quad \boldsymbol{V}^T \boldsymbol{V} = \boldsymbol{I}_L, \quad L \leq D \tag{6}$$

- **Principle of PCA:** Use some significant principal components to project data into a new coordinate system

Principal Component Analysis (PCA)

- Invented in 1901 by Karl Pearson.
- Reinvented independently by many researchers.
- **Implementations:** Assume $\boldsymbol{X}^T \mathbf{1}_N = \mathbf{0}_D$ (column-wise zero mean)
  - Singular value decomposition of $\boldsymbol{X} \in \mathbb{R}^{N\times D}$
  - Eigenvalue decomposition of $\boldsymbol{X}^T\boldsymbol{X} \in \mathbb{R}^{D\times D}$
- **Principals components of a collection of data points:**
  - L orthonormal vectors (an orthonormal basis) in the sample space $\mathcal{X} \subset \mathbb{R}^D$

$$\boldsymbol{V} = [\boldsymbol{v}_\ell] \in \mathbb{R}^{D\times L}, \quad \boldsymbol{V}^T\boldsymbol{V} = \boldsymbol{I}_L, \quad L \leq D \tag{6}$$

- **Principle of PCA:** Use some significant principal components to project data into a new coordinate system (equivalently, a represent the projected data via the orthonormal basis of the new space).

(Classic) Sequential Derivation of Principal Components

Motivation:

- Each principal component corresponds to an orthonormal base of the new space, which determines a **projection direction** of data.

(Classic) Sequential Derivation of Principal Components

Motivation:

- Each principal component corresponds to an orthonormal base of the new space, which determines a **projection direction** of data.

Principle:

- Sequentially find the projections maximizing the preserved energy of data **(Minimizing the residual that cannot be captured by the projections)**.

(Classic) Sequential Derivation of Principal Components

Motivation:

- Each principal component corresponds to an orthonormal base of the new space, which determines a **projection direction** of data.

Principle:

- Sequentially find the projections maximizing the preserved energy of data **(Minimizing the residual that cannot be captured by the projections)**.

Method:

1. Initialize residual matrix $\boldsymbol{R}^{(0)} = \boldsymbol{X}$

(Classic) Sequential Derivation of Principal Components

Motivation:

- Each principal component corresponds to an orthonormal base of the new space, which determines a **projection direction** of data.

Principle:

- Sequentially find the projections maximizing the preserved energy of data **(Minimizing the residual that cannot be captured by the projections)**.

Method:

1. Initialize residual matrix $\boldsymbol{R}^{(0)} = \boldsymbol{X}$
2. For $\ell = 1, ..., L$:
   - 2.1 Find the ℓ-th principal component:

$$\boldsymbol{v}_\ell = \arg\max_v \|\boldsymbol{R}^{(\ell-1)}\boldsymbol{v}\|_2^2, \qquad s.t. \|\boldsymbol{v}\|_2 = 1, \quad \boldsymbol{v}_i^T\boldsymbol{v} = 0, \forall \boldsymbol{i} = 1, ..., \ell - 1. \tag{7}$$

   - 2.2 Update residual:

$$\boldsymbol{R}^{(\ell)} = \boldsymbol{X} - \sum_{i=1}^{\ell} \boldsymbol{X}\boldsymbol{v}_\ell\boldsymbol{v}_\ell^T$$

(Classic) Sequential Derivation of Principal Components

Motivation:

- Each principal component corresponds to an orthonormal base of the new space, which determines a **projection direction** of data.

Principle:

- Sequentially find the projections maximizing the preserved energy of data **(Minimizing the residual that cannot be captured by the projections)**.

Method:

1. Initialize residual matrix $\boldsymbol{R}^{(0)} = \boldsymbol{X}$
2. For $\ell = 1, ..., L$:

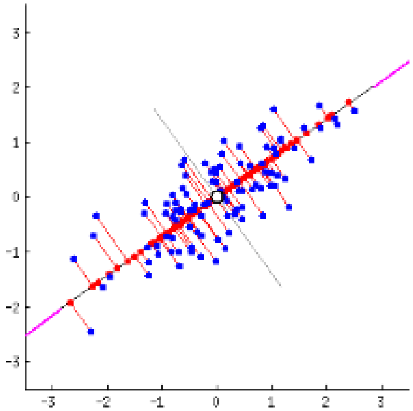
   2.1 Find the ℓ-th principal component:

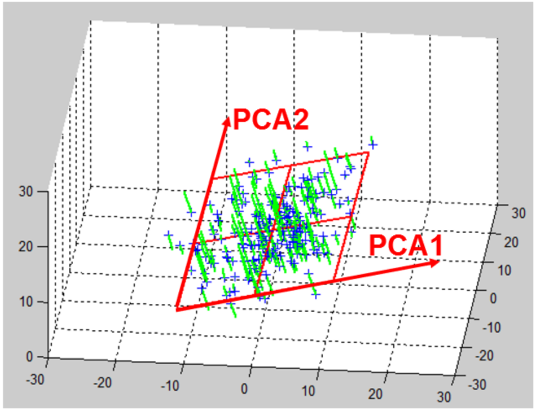
$$\boldsymbol{v}_\ell = \arg\max_v \|\boldsymbol{R}^{(\ell-1)}\boldsymbol{v}\|_2^2, \qquad s.t. \|\boldsymbol{v}\|_2 = 1, \quad \boldsymbol{v}_i^T\boldsymbol{v} = 0, \forall \boldsymbol{i} = 1, ..., \ell - 1. \tag{7}$$

   2.2 Update residual:

$$\boldsymbol{R}^{(\ell)} = \boldsymbol{X} - \sum_{i=1}^{\ell} \boldsymbol{X}\boldsymbol{v}_\ell\boldsymbol{v}_\ell^T = \boldsymbol{R}^{(\ell-1)} - \boldsymbol{R}^{(\ell-1)}\boldsymbol{v}_\ell\boldsymbol{v}_\ell^T. \tag{8}$$

Illustration of PCA





(Practical) Derivation of Principal Components

The above sequential derivation can be achieved by

- **Singular value decomposition (SVD)**

$$\boldsymbol{X} = \boldsymbol{U}\Sigma\boldsymbol{V}^T$$

(Practical) Derivation of Principal Components

The above sequential derivation can be achieved by

- **Singular value decomposition (SVD)**

$$\boldsymbol{X} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T \tag{9}$$

- **Eigenvalue decomposition**

$$\boldsymbol{X}^T\boldsymbol{X} = \boldsymbol{V}\underbrace{\Lambda}_{\boldsymbol{\Sigma}^2}\boldsymbol{V}^T$$

(Practical) Derivation of Principal Components

The above sequential derivation can be achieved by

- **Singular value decomposition (SVD)**

$$\boldsymbol{X} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T \tag{9}$$

- **Eigenvalue decomposition**

$$\boldsymbol{X}^T\boldsymbol{X} = \boldsymbol{V}\underbrace{\boldsymbol{\Lambda}}_{\boldsymbol{\Sigma}^2}\boldsymbol{V}^T \tag{10}$$

- $\boldsymbol{\Sigma} = \mathrm{diag}(\sigma_1, ..., \sigma_D)$ and $\sigma_1 \geq ... \geq \sigma_D \geq 0$, $\boldsymbol{\Lambda} = \mathrm{diag}(\lambda_1, ..., \lambda_D)$ and $\lambda_1 \geq ... \geq \lambda_D \geq 0$.

(Practical) Derivation of Principal Components

The above sequential derivation can be achieved by

- **Singular value decomposition (SVD)**

$$\boldsymbol{X} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T \quad (9)$$

- **Eigenvalue decomposition**

$$\boldsymbol{X}^T\boldsymbol{X} = \boldsymbol{V}\underbrace{\boldsymbol{\Lambda}}_{\boldsymbol{\Sigma}^2}\boldsymbol{V}^T \quad (10)$$

- $\boldsymbol{\Sigma} = \mathrm{diag}(\sigma_1, ..., \sigma_D)$ and $\sigma_1 \geq ... \geq \sigma_D \geq 0$, $\boldsymbol{\Lambda} = \mathrm{diag}(\lambda_1, ..., \lambda_D)$ and $\lambda_1 \geq ... \geq \lambda_D \geq 0$.
- The L principal components are the top-L columns of $\boldsymbol{V}$, Denoted as $\boldsymbol{V}_L = [\boldsymbol{v}_1, ..., \boldsymbol{v}_L]$

(Practical) Derivation of Principal Components

The above sequential derivation can be achieved by

- **Singular value decomposition (SVD)**

$$\boldsymbol{X} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T \tag{9}$$

- **Eigenvalue decomposition**

$$\boldsymbol{X}^T\boldsymbol{X} = \boldsymbol{V}\underbrace{\boldsymbol{\Lambda}}_{\boldsymbol{\Sigma}^2}\boldsymbol{V}^T \tag{10}$$

- $\boldsymbol{\Sigma} = \mathrm{diag}(\sigma_1, ..., \sigma_D)$ and $\sigma_1 \geq ... \geq \sigma_D \geq 0$, $\boldsymbol{\Lambda} = \mathrm{diag}(\lambda_1, ..., \lambda_D)$ and $\lambda_1 \geq ... \geq \lambda_D \geq 0$.
- The L principal components are the top-L columns of $\boldsymbol{V}$, Denoted as $\boldsymbol{V}_L = [\boldsymbol{v}_1, ..., \boldsymbol{v}_L]$
- $\boldsymbol{U}_L\boldsymbol{\Sigma}_L = \boldsymbol{X}\boldsymbol{V}_L \in \mathbb{R}^{N\times L}$: The new representation of the data (The projections obtained by PCs).

Revisit Data Whitening through PCA

Data Whitening of Data Matrix $\boldsymbol{X} \in \mathbb{R}^{N \times D}$

- Estimate expectation $\hat{\mu}_d = \frac{1}{N} \sum_{n=1}^{N} x_{nd}$ for $d = 1, \ldots, D$.
- Estimate covariance matrix

$$\widehat{\boldsymbol{\Gamma}} = \frac{1}{N-1} (\boldsymbol{X} - \mathbf{1}_N \hat{\boldsymbol{\mu}}^T)^T (\boldsymbol{X} - \mathbf{1}_N \hat{\boldsymbol{\mu}}^T) \in \mathbb{R}^{D \times D} \tag{11}$$

- Whitening: $\widehat{\boldsymbol{X}} = (\boldsymbol{X} - \mathbf{1}_N \hat{\boldsymbol{\mu}}^T) \widehat{\boldsymbol{\Gamma}}^{-\frac{1}{2}}$.

Revisit Data Whitening through PCA

Data Whitening of Data Matrix $\boldsymbol{X} \in \mathbb{R}^{N \times D}$

- Estimate expectation $\hat{\mu}_d = \frac{1}{N} \sum_{n=1}^{N} x_{nd}$ for $d = 1, \ldots, D$.
- Estimate covariance matrix

$$\widehat{\boldsymbol{\Gamma}} = \frac{1}{N-1} (\boldsymbol{X} - \mathbf{1}_N \hat{\boldsymbol{\mu}}^T)^T (\boldsymbol{X} - \mathbf{1}_N \hat{\boldsymbol{\mu}}^T) \in \mathbb{R}^{D \times D} \tag{11}$$

- Whitening: $\widehat{\boldsymbol{X}} = (\boldsymbol{X} - \mathbf{1}_N \hat{\boldsymbol{\mu}}^T) \widehat{\boldsymbol{\Gamma}}^{-\frac{1}{2}}$.

PCA of Data Matrix $\boldsymbol{X} \in \mathbb{R}^{N \times D}$

- Shifting to zero-mean data: $\tilde{\boldsymbol{X}} = \boldsymbol{X} - \mathbf{1}_N \hat{\boldsymbol{\mu}}^T$

Revisit Data Whitening through PCA

Data Whitening of Data Matrix $\boldsymbol{X} \in \mathbb{R}^{N \times D}$

- Estimate expectation $\hat{\mu}_d = \frac{1}{N} \sum_{n=1}^{N} x_{nd}$ for $d = 1, \ldots, D$.
- Estimate covariance matrix

$$\widehat{\boldsymbol{\Gamma}} = \frac{1}{N-1} (\boldsymbol{X} - \mathbf{1}_N \hat{\boldsymbol{\mu}}^T)^T (\boldsymbol{X} - \mathbf{1}_N \hat{\boldsymbol{\mu}}^T) \in \mathbb{R}^{D \times D} \tag{11}$$

- Whitening: $\widehat{\boldsymbol{X}} = (\boldsymbol{X} - \mathbf{1}_N \hat{\boldsymbol{\mu}}^T) \widehat{\boldsymbol{\Gamma}}^{-\frac{1}{2}}$.

PCA of Data Matrix $\boldsymbol{X} \in \mathbb{R}^{N \times D}$

- Shifting to zero-mean data: $\tilde{\boldsymbol{X}} = \boldsymbol{X} - \mathbf{1}_N \hat{\boldsymbol{\mu}}^T$
- Applying eigenvalue decomposition

$$\tilde{\boldsymbol{X}}^T \tilde{\boldsymbol{X}} = \boldsymbol{V} \boldsymbol{\Lambda} \boldsymbol{V}^T = (N-1) \widehat{\boldsymbol{\Gamma}}$$

Revisit Data Whitening through PCA

Data Whitening of Data Matrix $\boldsymbol{X} \in \mathbb{R}^{N \times D}$

- Estimate expectation $\hat{\mu}_d = \frac{1}{N} \sum_{n=1}^{N} x_{nd}$ for $d = 1, ..., D$.
- Estimate covariance matrix

$$\widehat{\boldsymbol{\Gamma}} = \frac{1}{N-1} (\boldsymbol{X} - \mathbf{1}_N \hat{\boldsymbol{\mu}}^T)^T (\boldsymbol{X} - \mathbf{1}_N \hat{\boldsymbol{\mu}}^T) \in \mathbb{R}^{D \times D} \tag{11}$$

- Whitening: $\widehat{\boldsymbol{X}} = (\boldsymbol{X} - \mathbf{1}_N \hat{\boldsymbol{\mu}}^T) \widehat{\boldsymbol{\Gamma}}^{-\frac{1}{2}}$.

PCA of Data Matrix $\boldsymbol{X} \in \mathbb{R}^{N \times D}$

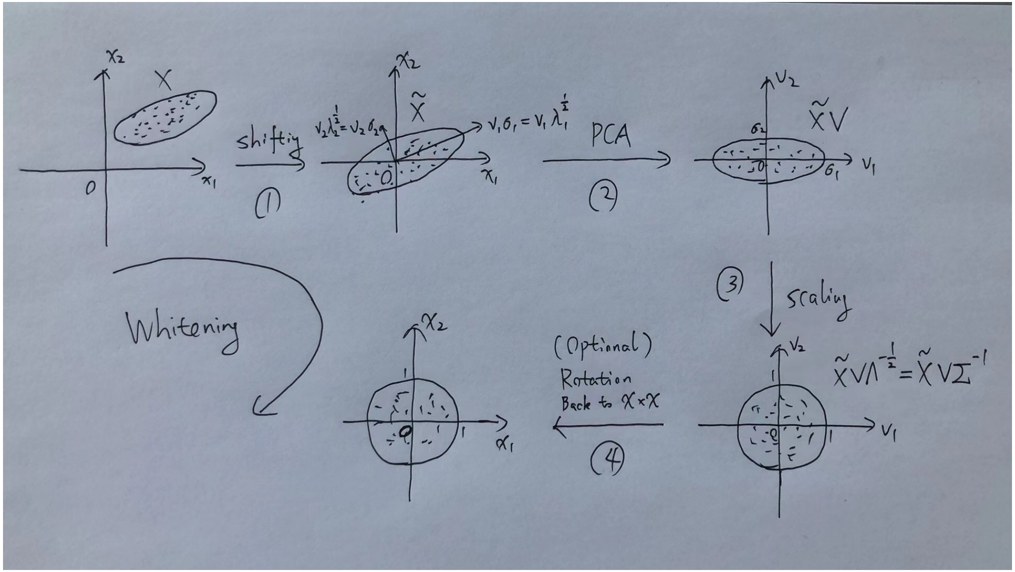
- Shifting to zero-mean data: $\tilde{\boldsymbol{X}} = \boldsymbol{X} - \mathbf{1}_N \hat{\boldsymbol{\mu}}^T$
- Applying eigenvalue decomposition

$$\tilde{\boldsymbol{X}}^T \tilde{\boldsymbol{X}} = \boldsymbol{V} \boldsymbol{\Lambda} \boldsymbol{V}^T = (N-1) \widehat{\boldsymbol{\Gamma}} \tag{12}$$

- **Consider all PCs:** Data Whitening $\simeq$ Shifting + PCA + Scaling

$$(N-1) \tilde{\boldsymbol{X}} \widehat{\boldsymbol{\Gamma}}^{-\frac{1}{2}} = \tilde{\boldsymbol{X}} \boldsymbol{V} \boldsymbol{\Lambda}^{-\frac{1}{2}} \boldsymbol{V}^T = (\tilde{\boldsymbol{X}} \boldsymbol{V} \boldsymbol{\Lambda}^{-\frac{1}{2}}) \boldsymbol{V}^T \simeq \tilde{\boldsymbol{X}} \boldsymbol{V} \boldsymbol{\Lambda}^{-\frac{1}{2}} \tag{13}$$

Revisit Data Whitening through PCA



PCA is Least-Square Data Denoising in Statistical ML

- An i.i.d. Gaussian noise model for observed data $\boldsymbol{X}$

$$\boldsymbol{X}_{noisy} = \boldsymbol{X}_{clean} + \boldsymbol{E}, \quad \forall \epsilon_{nd} \in \boldsymbol{E} \sim \mathcal{N}(0, \sigma^2)$$

PCA is Least-Square Data Denoising in Statistical ML

- An i.i.d. Gaussian noise model for observed data $\boldsymbol{X}$

$$\boldsymbol{X}_{noisy} = \boldsymbol{X}_{clean} + \boldsymbol{E}, \quad \forall \epsilon_{nd} \in \boldsymbol{E} \sim \mathcal{N}(0, \sigma^2) \tag{14}$$

- The least square data denoising problem:

$$\widehat{\boldsymbol{X}} = \arg\min_{\boldsymbol{X} \in \Omega} \|\boldsymbol{X}_{noisy} - \boldsymbol{X}\|_F^2$$

PCA is Least-Square Data Denoising in Statistical ML

- An i.i.d. Gaussian noise model for observed data $\boldsymbol{X}$

$$\boldsymbol{X}_{noisy} = \boldsymbol{X}_{clean} + \boldsymbol{E}, \quad \forall \epsilon_{nd} \in \boldsymbol{E} \sim \mathcal{N}(0, \sigma^2) \tag{14}$$

- The least square data denoising problem:

$$\widehat{\boldsymbol{X}} = \arg\min_{\boldsymbol{X} \in \Omega} \|\boldsymbol{X}_{noisy} - \boldsymbol{X}\|_F^2 \tag{15}$$

- When the feasible domain corresponds to a low-rank constraint

$$\Omega := \{\boldsymbol{X} \in \mathbb{R}^{N \times D} \mid \text{rank}(\boldsymbol{X}) \leq L\}. \tag{16}$$

We have

$$\widehat{\boldsymbol{X}} = \arg\min_{\boldsymbol{X} \in \Omega} \|\boldsymbol{X}_{noisy} - \boldsymbol{X}\|_F^2 = \boldsymbol{U}_L \boldsymbol{\Sigma}_L \boldsymbol{V}_L^T, \quad \text{where } \boldsymbol{X}_{noisy} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T. \tag{17}$$

(It is natural – recall the sequential minimization of residual.)

Extend PCA to More Generalized Data Denoising Model

Robust PCA: Consider sparse noise

$$\begin{aligned}
&\boldsymbol{X}_{noisy} = \boldsymbol{X}_{clean} + \boldsymbol{E}, \quad \forall \epsilon_{nd} \in \boldsymbol{E} \sim \text{Laplace}(0, \sigma) \\
&\widehat{\boldsymbol{X}} = \arg\min_{\boldsymbol{X} \in \Omega} \|\boldsymbol{X}_{noisy} - \boldsymbol{X}\|_1 \\
&\Omega := \{\boldsymbol{X} \in \mathbb{R}^{N \times D} \mid \text{rank}(\boldsymbol{X}) \le L\}.
\end{aligned} \tag{18}$$

Extend PCA to More Generalized Data Denoising Model

Robust PCA: Consider sparse noise

$$\begin{aligned}
&\boldsymbol{X}_{noisy} = \boldsymbol{X}_{clean} + \boldsymbol{E}, \quad \forall \epsilon_{nd} \in \boldsymbol{E} \sim \text{Laplace}(0, \sigma) \\
&\widehat{\boldsymbol{X}} = \arg\min_{\boldsymbol{X} \in \Omega} \|\boldsymbol{X}_{noisy} - \boldsymbol{X}\|_1 \\
&\Omega := \{\boldsymbol{X} \in \mathbb{R}^{N \times D} \mid \text{rank}(\boldsymbol{X}) \leq L\}.
\end{aligned} \tag{18}$$

Nonnegative Matrix Factorization (NMF)

$$\begin{aligned}
&\widehat{\boldsymbol{X}} = \arg\min_{\boldsymbol{X} \in \Omega} \|\boldsymbol{X}_{noisy} - \boldsymbol{X}\|_F^2 \\
&\Omega := \{\boldsymbol{X} = \boldsymbol{U}\boldsymbol{V}^T \mid \text{rank}(\boldsymbol{X}) = L, \boldsymbol{U}, \boldsymbol{V} \geq \boldsymbol{0}\}
\end{aligned} \tag{19}$$

Extend PCA to More Generalized Data Denoising Model

Robust PCA: Consider sparse noise

$$\begin{aligned} &\boldsymbol{X}_{noisy} = \boldsymbol{X}_{clean} + \boldsymbol{E}, \quad \forall \epsilon_{nd} \in \boldsymbol{E} \sim \text{Laplace}(0, \sigma) \\ &\widehat{\boldsymbol{X}} = \arg\min_{\boldsymbol{X} \in \Omega} \|\boldsymbol{X}_{noisy} - \boldsymbol{X}\|_1 \\ &\Omega := \{\boldsymbol{X} \in \mathbb{R}^{N \times D} \mid \text{rank}(\boldsymbol{X}) \leq L\}. \end{aligned} \tag{18}$$

Nonnegative Matrix Factorization (NMF)

$$\begin{aligned} &\widehat{\boldsymbol{X}} = \arg\min_{\boldsymbol{X} \in \Omega} \|\boldsymbol{X}_{noisy} - \boldsymbol{X}\|_F^2 \\ &\Omega := \{\boldsymbol{X} = \boldsymbol{U}\boldsymbol{V}^T \mid \text{rank}(\boldsymbol{X}) = L, \boldsymbol{U}, \boldsymbol{V} \geq \boldsymbol{0}\} \end{aligned} \tag{19}$$

Solving these problems requires iterative low-rank factorization/SVD and factor updating.

Extend PCA to More Generalized Data Denoising Model

Subspace Clustering:

- Recall the supervised feature selection:

$$\min_{\boldsymbol{w}} \|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{w}\|_2^2 + \gamma\|\boldsymbol{w}\|_1 .$$

Extend PCA to More Generalized Data Denoising Model

Subspace Clustering:

- Recall the supervised feature selection:

$$\min_{\boldsymbol{w}} \|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{w}\|_2^2 + \gamma\|\boldsymbol{w}\|_1. \tag{20}$$

- When the supervised signal $\boldsymbol{y}$ is the data itself:

$$\min_{\boldsymbol{W}\in\mathbb{R}^{D\times D}} \|\boldsymbol{X} - \boldsymbol{X}\boldsymbol{W}\|_F^2 + \gamma_1\|\boldsymbol{W}\|_1 + \gamma_2\|\boldsymbol{W}\|_*.$$

Extend PCA to More Generalized Data Denoising Model

Subspace Clustering:

- Recall the supervised feature selection:

$$\min_{\boldsymbol{w}} \|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{w}\|_2^2 + \gamma\|\boldsymbol{w}\|_1. \tag{20}$$

- When the supervised signal $\boldsymbol{y}$ is the data itself:

$$\min_{\boldsymbol{W}\in\mathbb{R}^{D\times D}} \|\boldsymbol{X} - \boldsymbol{X}\boldsymbol{W}\|_F^2 + \gamma_1\|\boldsymbol{W}\|_1 + \gamma_2\|\boldsymbol{W}\|_*. \tag{21}$$

Dictionary Learning:

$$\boldsymbol{\Psi}, \boldsymbol{A} = \arg\min_{\boldsymbol{\Psi},\boldsymbol{A}} \|\boldsymbol{X} - \boldsymbol{\Psi}\boldsymbol{A}\|_F^2 + \gamma\|\boldsymbol{A}\|_1 \tag{22}$$

Do We Have To Learn The Projection? — Compressive Sensing

For $\boldsymbol{x} \in \mathbb{R}^N$, if the following condition holds:

$$\exists \boldsymbol{\Psi}, \quad \boldsymbol{x} \approx \boldsymbol{\Psi}\boldsymbol{\alpha} \quad \text{and} \quad \|\boldsymbol{\alpha}\|_0 \leq S. \tag{23}$$

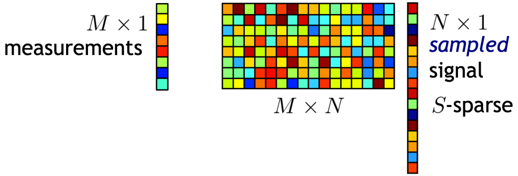
Do We Have To Learn The Projection? — Compressive Sensing

For $\boldsymbol{x} \in \mathbb{R}^N$, if the following condition holds:

$$\exists \boldsymbol{\Psi}, \quad \boldsymbol{x} \approx \boldsymbol{\Psi}\boldsymbol{\alpha} \quad \text{and} \quad \|\boldsymbol{\alpha}\|_0 \leq S. \tag{23}$$

Compressive sensing: $\boldsymbol{y} \in \mathbb{R}^M, M \ll N$

- $\boldsymbol{y} = \boldsymbol{\Phi}\boldsymbol{x}$.
- $\boldsymbol{\Phi}$ is a **random** matrix.



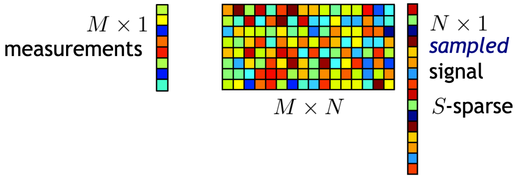
Do We Have To Learn The Projection? — Compressive Sensing

For $\boldsymbol{x} \in \mathbb{R}^N$, if the following condition holds:

$$\exists \boldsymbol{\Psi}, \quad \boldsymbol{x} \approx \boldsymbol{\Psi}\boldsymbol{\alpha} \quad \text{and} \quad \|\boldsymbol{\alpha}\|_0 \leq S. \tag{23}$$

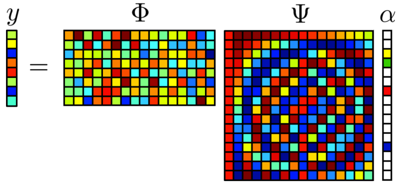
Compressive sensing: $\boldsymbol{y} \in \mathbb{R}^M, M \ll N$

- $\boldsymbol{y} = \boldsymbol{\Phi}\boldsymbol{x}$.
- $\boldsymbol{\Phi}$ is a **random** matrix.



Recovery guarantee of $\boldsymbol{x}$:

1. $\hat{\alpha} = \arg\min \|\boldsymbol{y} - \boldsymbol{\Phi}\boldsymbol{\Psi}\boldsymbol{\alpha}\|_2^2 + \gamma\|\boldsymbol{\alpha}\|_1$.



2. $\tilde{\boldsymbol{x}} = \boldsymbol{\Psi}\hat{\alpha}$.

Compressive Sensing: Main Theorem

When $\mathbf{\Phi}$ satisfies the **Restricted Isometry Property (RIP)**,

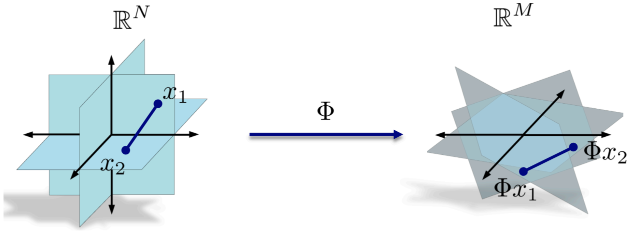
$$\forall \boldsymbol{x}_1, \boldsymbol{x}_2 \in \{\boldsymbol{x} \in \mathbb{R}^N \mid \|\boldsymbol{x}\|_0 \leq S\}, \quad \exists \delta < 1 \quad , 1 - \delta \leq \frac{\|\mathbf{\Phi}\boldsymbol{x}_1 - \mathbf{\Phi}\boldsymbol{x}_2\|_2^2}{\|\boldsymbol{x}_1 - \boldsymbol{x}_2\|_2^2} \leq 1 + \delta \tag{24}$$

We have

- $M \geq \frac{1}{C} S \log \frac{N}{S}$ leads to an exact reconstruction with a probability $1 - \mathcal{O}(N^{-M})$.
- Solving $\min_{\boldsymbol{y}=\mathbf{\Phi}\boldsymbol{\alpha}} \|\boldsymbol{\alpha}\|_1$ is equivalent to $\min_{\boldsymbol{y}=\mathbf{\Phi}\boldsymbol{\alpha}} \|\boldsymbol{\alpha}\|_0$.

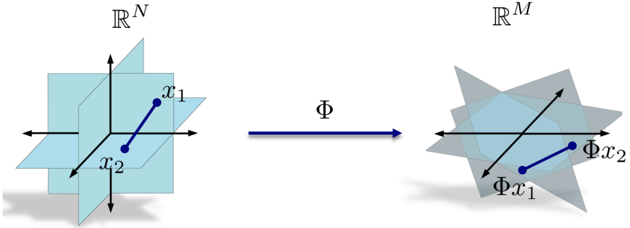
Compressive Sensing: Restricted Isometry Property (RIP)

$$\forall \boldsymbol{x}_1, \boldsymbol{x}_2 \in \{\boldsymbol{x} \in \mathbb{R}^N \mid \|\boldsymbol{x}\|_0 \leq S\}, \quad \exists \delta < 1 \quad , 1 - \delta \leq \frac{\|\boldsymbol{\Phi x}_1 - \boldsymbol{\Phi x}_2\|_2^2}{\|\boldsymbol{x}_1 - \boldsymbol{x}_2\|_2^2} \leq 1 + \delta \tag{25}$$



Compressive Sensing: Restricted Isometry Property (RIP)

$$\forall \boldsymbol{x}_1, \boldsymbol{x}_2 \in \{\boldsymbol{x} \in \mathbb{R}^N \mid \|\boldsymbol{x}\|_0 \leq S\}, \quad \exists \delta < 1 \quad , 1 - \delta \leq \frac{\|\boldsymbol{\Phi x}_1 - \boldsymbol{\Phi x}_2\|_2^2}{\|\boldsymbol{x}_1 - \boldsymbol{x}_2\|_2^2} \leq 1 + \delta \qquad (25)$$



- **Sub-Gaussian matrix:** Gaussian, Bernoulli ($\{0, 1\}$), Rademacher (± 1), Any Bounded Distributions.
- **Random Fourier submatrix:** $\boldsymbol{SFD}$, $\boldsymbol{S} \in \{0,1\}^{M \times N}$, $\boldsymbol{S1}_N = \boldsymbol{1}_M$, $\boldsymbol{D} = \text{diag}(\pm 1)$.

In Summary

- The motivation of dimensionality reduction
- Principle component analysis and its algorithms
- Other projection and factorization models

Next...

- Nonlinear dimensionality reduction (manifold learning, kernel PCA, ...)