Lecture 6: Linear Dimensionality Reduction

Shukai Gong

1 Curse of Dimensionality

We have mentioned in Section **??** that high-dimensional data has stronger data representation ability and bring about more potentials to our model. However, high-dimensional data also has some drawbacks.

Combinatorial Explosion (Discrete Example)

The number of different $d-$dimensional binary vectors is 2^d. Each addtional dimension doubles the effort needed to try all combinations.
(The search space of chess)

High Dimensional Sampling (Continous Example)

Sampling N samples randomly from a $d-$dimensional sample space, based on a distribution with identity covariance matrix $\boldsymbol{\Sigma} = \boldsymbol{I}_d$, it can be proved that the smallest and largest Euclidean distance $d_{\min}(D)$ and $d_{\max}(D)$ between any two samples satisfy

$$\frac{d_{\min}(D)}{d_{\max}(D)} \approx \left(\frac{1}{2}\right)^{\frac{1}{d}} \to 1,\ d \to \infty$$

meaning that maximum distance becomes indiscernible compared to the minimum distance. **Euclidean distance functions are losing their usefulness in high dimensions.**

One method of dealing with the curse of dimensionality is **Dimensionality Reduction**, which is to map the high-dimensional data to a lower-dimensional space while preserving the essential structure of the data. Mathemtically, we would like to find a **linear projection**

$$\begin{aligned} f : \mathbb{R}^D &\longmapsto \mathbb{R}^L \\ \boldsymbol{x} &\longmapsto \boldsymbol{z} = f(\boldsymbol{x}) \end{aligned}$$

where $L \ll D$. Generally, $\boldsymbol{z} = f(\boldsymbol{x}) = \boldsymbol{U}^\top \boldsymbol{x}$, where $\boldsymbol{U} \in \mathbb{R}^{D \times L}$. A desired projection matrix can be obtained by **minimizing reconstruction error**. We want to see a small error when remapping $\boldsymbol{z}$ back to $\boldsymbol{x}$, i.e.

$$\begin{aligned} \exists g : \mathbb{R}^L &\longmapsto \mathbb{R}^D \\ \boldsymbol{z} &\longmapsto \boldsymbol{x} \approx g(f(\boldsymbol{x})) \end{aligned}$$

Another method is to consider **isometry**, which is to preserve the pairwise distance between data points in $\mathcal{X}$ and $\mathcal{Z}$, i.e.

$$d_{\mathcal{Z}}(f(\boldsymbol{x}_i), f(\boldsymbol{x}_j)) \approx d_{\mathcal{X}}(\boldsymbol{x}_i, \boldsymbol{x}_j)$$

2 Linear Dimensionality Reduction

2.1 Principal Component Analysis (PCA)

Recall that in LASSO-based **supervised** feature selection, we solve the optimization problem

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

For $\boldsymbol{X} = [\boldsymbol{x}_1, \boldsymbol{x}_2, \cdots, \boldsymbol{x}_D]$, the column $\boldsymbol{x_d}$ contributes to the estimation of $\boldsymbol{y}$ iff $\boldsymbol{w_d} \neq 0$. That's to say, the useful features are those with non-zero weights.

$$\hat{\boldsymbol{X}} = [\boldsymbol{x_d}]_{d:\boldsymbol{w_d}\neq 0} \in \mathbb{R}^{N\times L},\ \hat{\boldsymbol{w}} = [\boldsymbol{w_d}]_{d:\boldsymbol{w_d}\neq 0} \in \mathbb{R}^{L}, L < D$$

However, labeled training data $\boldsymbol{y}, \boldsymbol{X}$ might be hard or expensive to get, but unlabeled training data $\boldsymbol{X}$ might be more easily available. **PCA is able to extract meaningful directions from such unlabeled data.**

PCA

Principal Component Analysis (PCA) is an **unsupervised dimensionality reduction** technique. Given a matrix of data points $\boldsymbol{X} \in \mathbb{R}^{N\times D}$, it finds one or more **orthogonal directions** $\boldsymbol{v_i}$ that capture **the largest amount of variance** in the data.

Intuitively, the directions with less variance contain less information and may be discarded without introducing too much error.

The Principal of PCA is to sequentially find the projections maximizing the preserved energy of data (Minimizing the residual that cannot be captured by the projections). Therefore, we start from the first component $\boldsymbol{v_1}$.

Derivation of Principle Components

The first L principal components of a collection of data points $\boldsymbol{X} \in \mathbb{R}^{N\times D}$ are **the eigenvectors corresponding to the largest L orthonormal eigenvalues of the $\boldsymbol{X}^\top\boldsymbol{X} \in \mathbb{R}^{D\times D}$.**

Let $\boldsymbol{X} \in \mathbb{R}^{N\times D}$ be the data matrix with N rows of $D-$dimensional data. $\boldsymbol{x_i}^\top$ are considered to be i.i.d. samples from some random vector $\boldsymbol{x}$.

First, we process the data to be columnwise zero-mean, i.e. $\boldsymbol{X}^\top\boldsymbol{1}_N = \boldsymbol{0}_D$. This is achieved by substracting the average of all the rows, i.e. $\bar{\boldsymbol{x}}^\top = \frac{1}{N}\sum_{i=1}^{N}\boldsymbol{x_i}^\top$ from each row.

$$\boldsymbol{X} = \begin{bmatrix} \boldsymbol{x_1}^\top \\ \boldsymbol{x_2}^\top \\ \vdots \\ \boldsymbol{x_N}^\top \end{bmatrix} \overset{\text{zero-meaned}}{\Longrightarrow} \boldsymbol{X'} = \begin{bmatrix} x_{11}-\bar{x}_1 & x_{12}-\bar{x}_2 & \cdots & x_{1D}-\bar{x}_D \\ x_{21}-\bar{x}_1 & x_{22}-\bar{x}_2 & \cdots & x_{2D}-\bar{x}_D \\ \vdots & \vdots & \ddots & \vdots \\ x_{N1}-\bar{x}_1 & x_{N2}-\bar{x}_2 & \cdots & x_{ND}-\bar{x}_D \end{bmatrix} = \begin{bmatrix} \boldsymbol{x_1}^\top - \bar{\boldsymbol{x}}^\top \\ \boldsymbol{x_2}^\top - \bar{\boldsymbol{x}}^\top \\ \vdots \\ \boldsymbol{x_N}^\top - \bar{\boldsymbol{x}}^\top \end{bmatrix} = \boldsymbol{X} - \boldsymbol{1}_N\bar{\boldsymbol{x}}^\top$$

Still we denote $\boldsymbol{X}$ as the zero-meaned data matrix for convenience.

Second, since $\boldsymbol{X}$ is zero-meaned, the sample variance of the **datapoints' projections** onto a **unit vector** $\boldsymbol{v}$ is given by

$$\begin{aligned}\mathrm{Var}(\boldsymbol{x}^\top\boldsymbol{v}) &= \mathbb{E}[(\boldsymbol{x}^\top\boldsymbol{v})^2] - \mathbb{E}[\boldsymbol{x}^\top\boldsymbol{v}]^2 = \mathbb{E}[(\boldsymbol{x}^\top\boldsymbol{v})^2] \\ &= \frac{1}{N}\sum_{i=1}^{N}(\boldsymbol{x_i}^\top\boldsymbol{v})^2 = \frac{1}{N}\|\boldsymbol{X}\boldsymbol{v}\| = \frac{1}{N}\boldsymbol{v}^\top\boldsymbol{X}^\top\boldsymbol{X}\boldsymbol{v}\end{aligned}$$

With the motivation of maximizing the variance, we have the optimization problem

$$\max_{\boldsymbol{v}} \boldsymbol{v}^\top\boldsymbol{X}^\top\boldsymbol{X}\boldsymbol{v},\ \text{s.t. } \boldsymbol{v}^\top\boldsymbol{v} = 1$$

The Lagrangian of the optimization problem is

$$\begin{aligned}&\mathcal{L}(\boldsymbol{v}, \lambda) = \boldsymbol{v}^\top\boldsymbol{X}^\top\boldsymbol{X}\boldsymbol{v} + \lambda(\boldsymbol{v}^\top\boldsymbol{v} - 1) \\ \Rightarrow &\frac{\partial\mathcal{L}}{\partial\boldsymbol{v}} = 2\boldsymbol{X}^\top\boldsymbol{X}\boldsymbol{v} - 2\lambda\boldsymbol{v} = 0 \Rightarrow \boldsymbol{X}^\top\boldsymbol{X}\boldsymbol{v} = \lambda\boldsymbol{v}\end{aligned}$$

that the first component $\boldsymbol{v_1}$ satisfies $\boldsymbol{X}^\top \boldsymbol{X}\boldsymbol{v_1} = \lambda \boldsymbol{v_1}$, i.e. $\boldsymbol{v_1}$ is an eigenvector of $(\boldsymbol{X}^\top \boldsymbol{X})_{D\times D}$. Since our constraint is $\boldsymbol{v}^\top \boldsymbol{v} = 1$, we have

$$\boldsymbol{v_1}^\top \boldsymbol{X}^\top \boldsymbol{X}\boldsymbol{v_1} = \lambda \boldsymbol{v_1}^\top \boldsymbol{v_1} = \lambda$$
$$\Rightarrow \max_{\boldsymbol{v}} \boldsymbol{v}^\top \boldsymbol{X}^\top \boldsymbol{X}\boldsymbol{v} = \lambda_{\max}(\boldsymbol{X}^\top \boldsymbol{X})$$

So the first component $\boldsymbol{v_1}$ **is the orthonormal eigenvector corresponding to the largest eigenvalue of** $\boldsymbol{X}^\top \boldsymbol{X}$**.** Furthermore, it can be proved that the $i-$th component $\boldsymbol{v_i}$ is the eigenvector corresponding to the $i-$th largest eigenvalue of $\boldsymbol{X}^\top \boldsymbol{X}$.

Therefore, **The first L principal components of a collection of data points are the eigenvectors corresponding to the largest L orthonormal eigenvalues of the $\boldsymbol{X}^\top \boldsymbol{X}$.** It can also be observed that The first L principal components are the top-L columns of $\boldsymbol{V}$, the right-singular matrix of $\boldsymbol{X}$. Therefore, we can obtain PCs by performing SVD on $\boldsymbol{X}$.

Once we have computed the principal component $\boldsymbol{v_i}$, we can use them as a **new coordinate system.** The $k-$th principal component of **a $D-$ dimensional datapoint** $\boldsymbol{x_i} \in \mathbb{R}^D$ is $\boldsymbol{x_i}^\top \boldsymbol{v_k}$, the scalar projection of $\boldsymbol{x_i}$ onto the $k-$th principal component $\boldsymbol{v_k}$. Denote $\boldsymbol{V}_{D\times L} = [\boldsymbol{v_1}, \cdots, \boldsymbol{v_L}]$, **we can compute all the PCs of all the datapoints by**

$$\boldsymbol{Z}_{N\times L} = \boldsymbol{X}_{N\times D}\boldsymbol{V}_{D\times L}, \qquad L < D$$

We have successfully reduced the dimensionality of the data from D to L! **A reconstruct of $\boldsymbol{X}$ from $\boldsymbol{Z}$ will be**

$$\hat{\boldsymbol{X}}_{N\times D} = \boldsymbol{Z}_{N\times L}\boldsymbol{V}^\top_{L\times D} = \boldsymbol{X}_{N\times D}\boldsymbol{V}_{D\times L}\boldsymbol{V}^\top_{L\times D}$$

2.2 Whitening

Let's revisit Whitening. Given $\boldsymbol{X} = [\boldsymbol{x_1}, \cdots, \boldsymbol{x_D}]$ with D features, we similarily process the data to be columnwise zero-meaned:

$$\boldsymbol{X'} = \boldsymbol{X} - \boldsymbol{1}_N \bar{\boldsymbol{x}}^\top$$

where $\bar{\boldsymbol{x}} = \frac{1}{N}\sum_{i=1}^{N} \boldsymbol{x_i}$. Then the estimate covariance matrix measuring the covariance between feature i and j is given by

$$\hat{\boldsymbol{\Sigma}} = \frac{1}{N-1}(\boldsymbol{X} - \boldsymbol{1_N}\bar{\boldsymbol{x}}^\top)^\top(\boldsymbol{X} - \boldsymbol{1_N}\bar{\boldsymbol{x}}^\top) = \frac{1}{N-1}\boldsymbol{X'}^\top \boldsymbol{X'}$$

Then we can **whiten the data** by

$$\tilde{\boldsymbol{X}} = (\boldsymbol{X} - \boldsymbol{1_N}\bar{\boldsymbol{x}}^\top)\hat{\boldsymbol{\Sigma}}^{-\frac{1}{2}} = \boldsymbol{X'}\hat{\boldsymbol{\Sigma}}^{-\frac{1}{2}}$$

This is very similar to PCA. By EVD we know

$$\boldsymbol{X'}^\top \boldsymbol{X'} = \boldsymbol{V}\boldsymbol{\Lambda}\boldsymbol{V}^\top = (N-1)\hat{\boldsymbol{\Sigma}}$$

Therefore

$$(N-1)\tilde{\boldsymbol{X}} = (N-1)\boldsymbol{X'}\hat{\boldsymbol{\Sigma}}^{-\frac{1}{2}} = \underbrace{\boldsymbol{X'}}_{\text{Shifting}} \quad \underbrace{\boldsymbol{V}}_{\text{PCA Matrix}} \quad \underbrace{\boldsymbol{\Lambda}^{-\frac{1}{2}}}_{\text{Scaling}} \boldsymbol{V}^\top$$

2.3 Data Denoising by PCA

Data Denoising by PCA

We claim that PCA is least-square data denoising in statistical ML. Suppose an i.i.d Gaussian noise model for observed data $\boldsymbol{X}$

$$\boldsymbol{X}_{\text{noisy}} = \boldsymbol{X}_{\text{clean}} + \boldsymbol{E},\ \ \boldsymbol{E} \sim \mathcal{N}(0, \sigma^2 \boldsymbol{I})$$

The least square data denoising problem is denoted as

$$\hat{\boldsymbol{X}} = \arg\min_{\boldsymbol{X}} \|\boldsymbol{X}_{\text{noisy}} - \boldsymbol{X}\|_F^2$$

When the feasible domain corresponds to a low-rank constraint

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

We have the **closed-form solution**

$$\hat{\boldsymbol{X}} = \arg\min_{\boldsymbol{X} \in \Omega} \|\boldsymbol{X}_{\text{noisy}} - \boldsymbol{X}\|_F^2 = \boldsymbol{U}_L \boldsymbol{\Sigma}_L \boldsymbol{V}_L^\top,\ \text{where } \boldsymbol{X}_{\text{noisy}} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^\top$$

where $\boldsymbol{\Sigma}_L$ is the top-L singular values matrix and $\boldsymbol{U}_L, \boldsymbol{V}_L$ contains only the top-L singular vectors.

Intuitively, noises and unimportant information are usually captured by the last few principal components. Therefore, we only keep the first L principal components to denoise the data.

2.4 Other projection and factorization models

Robust PCA

Consider sparse noise

$$\boldsymbol{X}_{\text{noisy}} = \boldsymbol{X}_{\text{clean}} + \boldsymbol{E},\ \ \boldsymbol{E} \sim \mathcal{L}(0, \sigma^2 \boldsymbol{I})$$

Still we have the feasible domain corresponds to a low-rank constraint

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

The denoised data is given by

$$\hat{\boldsymbol{X}} = \arg\min_{\boldsymbol{X} \in \Omega} \|\boldsymbol{X}_{\text{noisy}} - \boldsymbol{X}\|_1$$

Non-negative Matrix Factorization (NMF)

The denoised data is given by

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

where the feasible domain is

$$\Omega := \{\boldsymbol{X} = \boldsymbol{U}\boldsymbol{V}^\top : \text{rank}(\boldsymbol{X}) = L,\ \boldsymbol{U}, \boldsymbol{V} \geq 0\}$$

Subspace Clustering

Subspace clustering is similar to LASSO, but the supervised signal $\boldsymbol{y}$ is the data itself:

$$\hat{\boldsymbol{W}} = \arg\min_{\boldsymbol{W}\in\mathbb{R}^{D\times D}} \|\boldsymbol{X} - \boldsymbol{X}\boldsymbol{W}\|_F^2 + \lambda\|\boldsymbol{W}\|_1 + \lambda_2\|\boldsymbol{W}\|_*$$

where $\|\boldsymbol{W}\|_*$ is the nuclear norm of $\boldsymbol{W}$, i.e. the sum of its singular values.

Before introducing **Compressive Sensing**, let's first look at an example of image recovering.

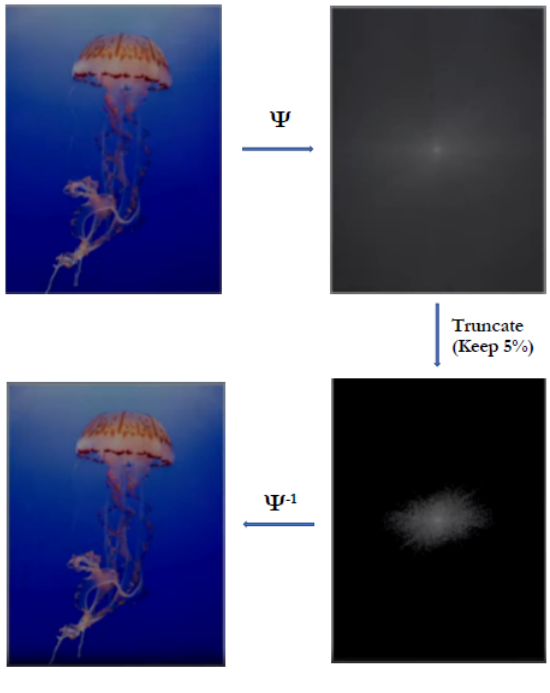


Image 1: Showing traditional Image compressing and decompressing using a basis function Ψ over a High Resolution-Image.

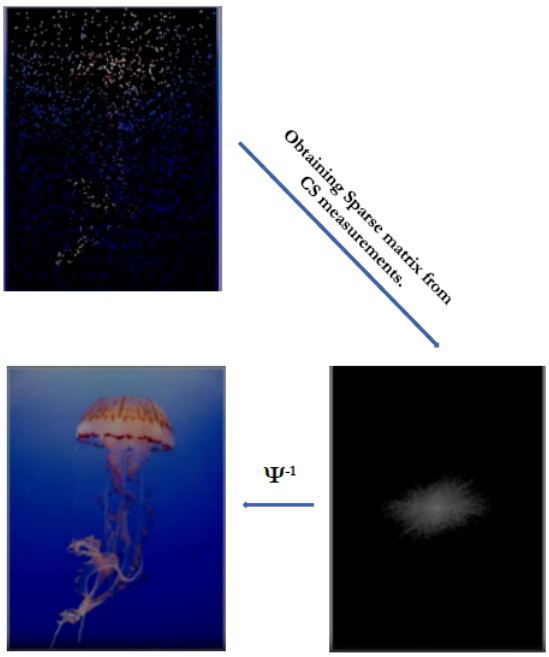


Fig 2: Recovering Original Image x by sampling few pixels (Φx) while sensing and solving for the sparse representation over a basis Ψ.

Suppose figure $\boldsymbol{x}$ is Fourier transformed $\boldsymbol{\Phi}$ to frequency domain $\boldsymbol{s}$, and after truncating the signal in $\boldsymbol{s}$, we still can expect to reconstruct a high quality image after the removal of some high frequency signals of lower energy.

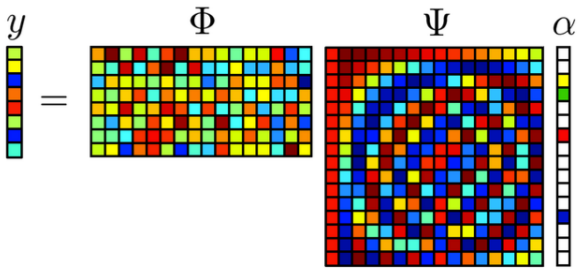
However, in practical, we want to reconstruct a high-quality original image $\boldsymbol{x'}$ using a low-quality observation $\boldsymbol{y}$, i.e., an undersampling of the true image $\boldsymbol{x}$, by transforming it to get a sparse space $\boldsymbol{s}$ of $\boldsymbol{x}$. Mathematically

Compressive Sensing

For an unknown signal $\boldsymbol{x}_{N\times 1}$, suppose it can be sparsely represented in sparse basis $\boldsymbol{\Psi}_{N\times K}$, i.e. $\boldsymbol{x} = \boldsymbol{\Psi}\boldsymbol{s}$, where coefficient vector $\boldsymbol{s}_{K\times 1}$ is sparse. We expect to recover $\boldsymbol{s}$ from observation $\boldsymbol{y}_{M\times 1}$. The optimization problem is

$$\begin{aligned}\hat{\boldsymbol{s}} &= \arg\min_{\boldsymbol{s}} \|\boldsymbol{y}_{M\times 1} - \boldsymbol{\Phi}_{M\times N}\boldsymbol{\Psi}_{N\times K}\boldsymbol{s}_{K\times 1}\|_2^2 + \lambda\|\boldsymbol{s}\|_1 \\ &= \arg\min_{\boldsymbol{s}} \|\boldsymbol{y}_{M\times 1} - \boldsymbol{\Theta}_{M\times K}\boldsymbol{s}_{K\times 1}\|_2^2 + \lambda\|\boldsymbol{s}\|_1\end{aligned}$$

where $\boldsymbol{\Phi}$ is a **random** measurement matrix, $\boldsymbol{\Psi}$ is the sparse basis matrix, and $\boldsymbol{s}$ is the sparse coefficient vector. Specifically, $\boldsymbol{\Theta}_{M\times K} = \boldsymbol{\Phi}_{M\times N}\boldsymbol{\Psi}_{N\times K}$ is named as sensing matrix.



If $M = N$ here, we can directly solve $\boldsymbol{y} = \boldsymbol{\Theta s}$. But normally $M < N$, there are infinitely many solutions to $\boldsymbol{y} = \boldsymbol{\Theta s}$. Therefore, we add a L_1 regularization term to the optimization problem to find the sparsest solution.

Furthermore, when $\boldsymbol{\Phi}$ satisfies the **Restricted Isometry property (RIP)**,

Restricted Isometry Property (RIP)

A matrix $\boldsymbol{\Phi}$ satisfies the RIP with restricted isometry constant $\delta_<1$ if for all $K-$sparse vectors $\boldsymbol{x_1}, \boldsymbol{x_2}$, i.e. $\forall \boldsymbol{x_1}, \boldsymbol{x_2} \in \{\boldsymbol{x} \in \mathbb{R}^N : \|\boldsymbol{x}\|_0 \leq S\}$, we have

$$(1-\delta) \leq \frac{\|\boldsymbol{\Phi x_1} - \boldsymbol{\Phi x_2}\|_2^2}{\|\boldsymbol{x_1} - \boldsymbol{x_2}\|_2^2} \leq (1+\delta)$$

we have

- **Stable Recovery:** When the rows of sensing matrix $M \geq \frac{1}{C} \cdot S \cdot \log \frac{N}{S}$ leads to an exact reconstruction with probability $1 - \mathcal{O}(N^{-M})$.
- Solving $\min_{\boldsymbol{s}} \|\boldsymbol{y} - \boldsymbol{\Theta s}\|_2^2 + \lambda \|\boldsymbol{s}\|_1$ is equivalent to solving $\min_{\boldsymbol{s}} \|\boldsymbol{y} - \boldsymbol{\Theta s}\|_2^2 + \lambda \|\boldsymbol{s}\|_0$

Commonly used sensing matrices $\boldsymbol{\Theta}$ include **Sub-Gaussian matrices(Gaussian, Bernoulli) and Fourier** matrices.