THU-70250043-0, Pattern Recognition (Spring 2021) **Homework: 10**

Dimensionality Reduction

Lecturer: Changshui Zhang zcs@mail.tsinghua.edu.cn

Hong Zhao vzhao@tsinghua.edu.cn

Student: Jingxuan Yang yangjx20@mails.tsinghua.edu.cn

PCA and Eigenvectors

1. Let $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n$ denote n vectors in $\mathbb{R}^D$, and we know the mean vector

$$\bar{\boldsymbol{x}} = \frac{1}{n}\sum_{i=1}^{n} \boldsymbol{x}_i = \mathbf{0} \in \mathbb{R}^D. \tag{1}$$

We project them into a lower dimensional space by performing a linear transformation

$$\boldsymbol{y}_i = \boldsymbol{W}^\top \boldsymbol{x}_i, \tag{2}$$

where $\boldsymbol{y}_i \in \mathbb{R}^d$, $\boldsymbol{W} \in \mathbb{R}^{D\times d}$, and $\boldsymbol{W}^\top \boldsymbol{W} = \boldsymbol{I} \in \mathbb{R}^{d\times d}$.

To simplify notations, we stack $\boldsymbol{x}_i$ column by column to make a data matrix $\boldsymbol{X} = [\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n] \in \mathbb{R}^{D\times n}$, and then perform the same operation on $\boldsymbol{y}_i$ to get $\boldsymbol{Y} \in \mathbb{R}^{d\times n}$. Then we can calculate the covariance matrix $\boldsymbol{\Sigma}_{\boldsymbol{X}} = \boldsymbol{X}\boldsymbol{X}^\top$, and $\boldsymbol{\Sigma}_{\boldsymbol{Y}} = \boldsymbol{Y}\boldsymbol{Y}^\top$. Please find the matrix $\boldsymbol{W}$ which maximizes the trace of $\boldsymbol{\Sigma}_{\boldsymbol{Y}}$. This problem has a closed-form solution and thus numerical solutions will not be accepted.

解: 待求优化问题为

$$\begin{aligned} \max_{\boldsymbol{W}} \ & \operatorname{Tr}(\boldsymbol{\Sigma}_{\boldsymbol{Y}}) \\ \text{s.t.} \ & \boldsymbol{W}^\top \boldsymbol{W} = \boldsymbol{I} \end{aligned} \tag{3}$$

由 $\boldsymbol{Y} = \boldsymbol{W}^\top \boldsymbol{X}$ 可得

$$\operatorname{Tr}(\boldsymbol{\Sigma}_{\boldsymbol{Y}}) = \operatorname{Tr}(\boldsymbol{Y}\boldsymbol{Y}^\top) = \operatorname{Tr}(\boldsymbol{W}^\top \boldsymbol{X}\boldsymbol{X}^\top \boldsymbol{W}) = \operatorname{Tr}(\boldsymbol{W}^\top \boldsymbol{\Sigma}_{\boldsymbol{X}} \boldsymbol{W}) \tag{4}$$

所以上述优化问题可写为

$$\begin{aligned} \max_{\boldsymbol{W}} \ & \operatorname{Tr}(\boldsymbol{W}^\top \boldsymbol{\Sigma}_{\boldsymbol{X}} \boldsymbol{W}) \\ \text{s.t.} \ & \boldsymbol{W}^\top \boldsymbol{W} = \boldsymbol{I} \end{aligned} \tag{5}$$

记 $\boldsymbol{W} = [\boldsymbol{w}_1, \boldsymbol{w}_2, \ldots, \boldsymbol{w}_d]$, $\boldsymbol{w}_i \in \mathbb{R}^D$, 则 $\boldsymbol{W}^\top \boldsymbol{W} = \boldsymbol{I}$ 之约束可写为

$$\boldsymbol{w}_i^\top \boldsymbol{w}_j = \delta_{ij}, \quad \forall\, i, j = 1, 2, \ldots, d \tag{6}$$

其中 δ_{ij} 是 Kronecker delta 函数.

引入 Lagrange 乘子 $\lambda_1, \lambda_2, \ldots, \lambda_d$, 并记 $\boldsymbol{\Lambda} = \mathrm{diag}(\lambda_1, \lambda_2, \ldots, \lambda_d)$, 可得 Lagrange 函数为

$$\begin{aligned} L(\boldsymbol{W}, \boldsymbol{\Lambda}) &= \mathrm{Tr}(\boldsymbol{W}^\top \boldsymbol{\Sigma}_{\boldsymbol{X}} \boldsymbol{W}) - \sum_{i=1}^{d} \lambda_i (\boldsymbol{w}_i^\top \boldsymbol{w}_i - 1) \\ &= \mathrm{Tr}(\boldsymbol{W}^\top \boldsymbol{\Sigma}_{\boldsymbol{X}} \boldsymbol{W}) - \mathrm{Tr}(\boldsymbol{\Lambda}(\boldsymbol{W}^\top \boldsymbol{W} - \boldsymbol{I})) \end{aligned} \tag{7}$$

注意到 $\boldsymbol{\Sigma}_{\boldsymbol{X}}$ 为对称矩阵, 可知 Lagrange 函数对 $\boldsymbol{W}$ 的偏导数为

$$\frac{\partial L(\boldsymbol{W}, \boldsymbol{\Lambda})}{\partial \boldsymbol{W}} = 2\boldsymbol{\Sigma}_{\boldsymbol{X}} \boldsymbol{W} - 2\boldsymbol{W}\boldsymbol{\Lambda} \tag{8}$$

令此偏导数为 $\mathbf{0}$ 可得

$$\boldsymbol{\Sigma}_{\boldsymbol{X}} \boldsymbol{W} = \boldsymbol{W}\boldsymbol{\Lambda} \tag{9}$$

所以, $\boldsymbol{w}_i$ 为矩阵 $\boldsymbol{\Sigma}_{\boldsymbol{X}}$ 的特征向量, 且 λ_i 为对应的特征值.

令 Lagrange 函数对 $\boldsymbol{\Lambda}$ 的偏导数为 $\mathbf{0}$ 可得

$$\boldsymbol{W}^\top \boldsymbol{W} = \boldsymbol{I} \tag{10}$$

此时

$$\mathrm{Tr}(\boldsymbol{W}^\top \boldsymbol{\Sigma}_{\boldsymbol{X}} \boldsymbol{W}) = \mathrm{Tr}(\boldsymbol{W}^\top \boldsymbol{W}\boldsymbol{\Lambda}) = \mathrm{Tr}(\boldsymbol{I}\boldsymbol{\Lambda}) = \mathrm{Tr}(\boldsymbol{\Lambda}) = \sum_{i=1}^{d} \lambda_i \tag{11}$$

则最大化 $\mathrm{Tr}(\boldsymbol{\Sigma}_{\boldsymbol{Y}}) = \mathrm{Tr}(\boldsymbol{\Lambda})$ 要求取 $\lambda_1, \lambda_2, \ldots, \lambda_d$ 为矩阵 $\boldsymbol{\Sigma}_{\boldsymbol{X}}$ 前 d 个最大的特征值. 令 $\tilde{\boldsymbol{w}}_i$ 为 λ_i 对应的特征向量, 由矩阵 $\boldsymbol{\Sigma}_{\boldsymbol{X}}$ 为实对称矩阵可知 $\{\tilde{\boldsymbol{w}}_1, \tilde{\boldsymbol{w}}_2, \ldots, \tilde{\boldsymbol{w}}_d\}$ 互相正交, 取

$$\boldsymbol{w}_i = \frac{\tilde{\boldsymbol{w}}_i}{\|\tilde{\boldsymbol{w}}_i\|}, \quad \forall\, i = 1, 2, \ldots, d \tag{12}$$

则 $\boldsymbol{W} = [\boldsymbol{w}_1, \boldsymbol{w}_2, \ldots, \boldsymbol{w}_d]$ 亦满足 $\boldsymbol{W}^\top \boldsymbol{W} = \boldsymbol{I}$, 所以此 $\boldsymbol{W}$ 可使得 $\mathrm{Tr}(\boldsymbol{\Sigma}_{\boldsymbol{Y}})$ 达到最大.

MDS and Strain

2. In MDS, we have the distance matrix $\boldsymbol{D} \in \mathbb{R}^{n\times n}$ for n data points, where $\boldsymbol{D}_{i,j} = (\boldsymbol{x}_i - \boldsymbol{x}_j)^\top(\boldsymbol{x}_i - \boldsymbol{x}_j)$. We first get the inner product matrix $\boldsymbol{B}$ by

$$\boldsymbol{B} = -\frac{1}{2}\boldsymbol{H}\boldsymbol{D}\boldsymbol{H}, \tag{13}$$

where $\boldsymbol{H}$ is defined as $\boldsymbol{H} \triangleq \boldsymbol{I} - \frac{1}{n}\mathbf{1}\mathbf{1}^\top$, $\mathbf{1} = (1, 1, \ldots, 1)^\top \in \mathbb{R}^{n\times 1}$ and $\boldsymbol{I} \in \mathbb{R}^{n\times n}$ is the identity matrix.

Suppose the desired number of dimensions for output is m. In the next step of MDS we should find the m largest eigenvalues values $\lambda_1, \lambda_2, \ldots, \lambda_m$ and corresponding eigenvectors $\boldsymbol{u}_1, \boldsymbol{u}_2, \ldots, \boldsymbol{u}_m \in \mathbb{R}^n$ of matrix $\boldsymbol{B}$ and the final output of MDS should be $\boldsymbol{X} = [\boldsymbol{u}_1, \boldsymbol{u}_2, \ldots, \boldsymbol{u}_m] \cdot \mathrm{diag}(\sqrt{\lambda_1}, \sqrt{\lambda_2}, \ldots, \sqrt{\lambda_m})$. Please prove that this procedure is equivalent to find $\boldsymbol{X}$ to minimize the strain, which is defined by

$$\mathrm{Strain}(\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n) = \sqrt{\frac{\sum_{i,j}(\boldsymbol{B}_{i,j} - \boldsymbol{x}_i^\top \boldsymbol{x}_j)^2}{\sum_{i,j}\boldsymbol{B}_{i,j}}}. \tag{14}$$

解: 记 $\boldsymbol{U} = [\boldsymbol{u}_1, \boldsymbol{u}_2, \ldots, \boldsymbol{u}_m]$, $\boldsymbol{\Lambda} = \mathrm{diag}(\lambda_1, \lambda_2, \ldots, \lambda_m)$, 则 $\boldsymbol{B} = \boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{U}^\top$, MDS 得到

$$\boldsymbol{X} = [\boldsymbol{u}_1, \boldsymbol{u}_2, \ldots, \boldsymbol{u}_m] \cdot \mathrm{diag}(\sqrt{\lambda_1}, \sqrt{\lambda_2}, \ldots, \sqrt{\lambda_m}) = \boldsymbol{U}\boldsymbol{\Lambda}^{\frac{1}{2}} \tag{15}$$

关于最小化 strain, 易知

$$\min_{\boldsymbol{X}\in\mathbb{R}^{n\times m}} \sqrt{\frac{\sum_{i,j}(\boldsymbol{B}_{i,j} - \boldsymbol{x}_i^\top \boldsymbol{x}_j)^2}{\sum_{i,j}\boldsymbol{B}_{i,j}}} \Longleftrightarrow \min_{\boldsymbol{X}\in\mathbb{R}^{n\times m}} \sum_{i=1}^{n}\sum_{j=1}^{n}(\boldsymbol{B}_{i,j} - \boldsymbol{x}_i^\top \boldsymbol{x}_j)^2 \tag{16}$$

由于

$$(\boldsymbol{B}_{i,j} - \boldsymbol{x}_i^\top \boldsymbol{x}_j)^2 \geqslant 0, \quad \forall\, i, j = 1, 2, \ldots, n \tag{17}$$

则 strain 最小值为 0, 且等号能够取到当且仅当

$$\boldsymbol{B}_{i,j} - \boldsymbol{x}_i^\top \boldsymbol{x}_j = 0, \quad \forall\, i, j = 1, 2, \ldots, n \tag{18}$$

写成矩阵形式即

$$\boldsymbol{B} = \boldsymbol{X}\boldsymbol{X}^\top \tag{19}$$

又 $\boldsymbol{B} = \boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{U}^\top$, 则有 $\boldsymbol{X} = \boldsymbol{U}\boldsymbol{\Lambda}^{\frac{1}{2}}$, 与 MDS 得到的 $\boldsymbol{X}$ 相同, 即 MDS 由内积矩阵 $\boldsymbol{B}$ 求取 $\boldsymbol{X}$ 的步骤与最小化 strain 等价.

ISOMAP and LLE

ISOMAP, LLE 对流形的降维

3. 考虑如下的问题并实现 ISOMAP, LLE 等降维方法. 注意: 数据在产生过程中可不必严格保证形状, 大致符合要求即可, 不用在数据的产生上花费过多时间. 可以参考 scikit-learn 的官方文档, 实现类似的效果, 但是不可以直接使用已有的 LLE 和 ISOMAP 函数.

3.1. 在三维空间中产生 “Z” 形状的流形, 使用 ISOMAP 方法降维并作图, 给出数据的三维分布图和最佳参数下的降维效果图.

解: 生成 “Z” 形状的流形数据如图 1 所示, 使用 ISOMAP 方法降维, 选择 20 近邻计算距离, 得到降维后的结果如图 2 所示, ISOMAP 将三维数据降维到二维平面, 并且保持着数据点在二维流形上的位置关系.

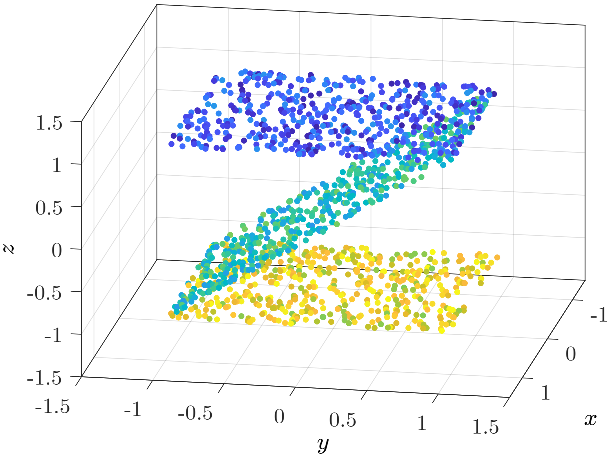

图 1: “Z” 形状流形数据分布图

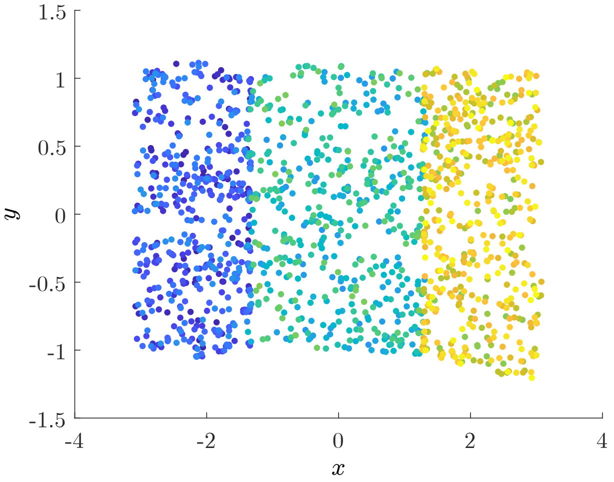

图 2: “Z” 形状流形数据 ISOMAP 降维结果

3.2. 在三维空间中产生 “W” 形状的流形, 使用 LLE 方法降维并作图, 给出数据的三维分布图和最佳参数下的降维效果图.

解: 生成 “W” 形状的流形数据如图 3 所示, 使用 LLE 方法降维, LLE 算法部分代码参考网站 [1], 选择 40 近邻计算距离, 得到降维后的结果如图 4 所示, LLE 将三维数据降维到二维平面, 并且保持了良好的可分性.

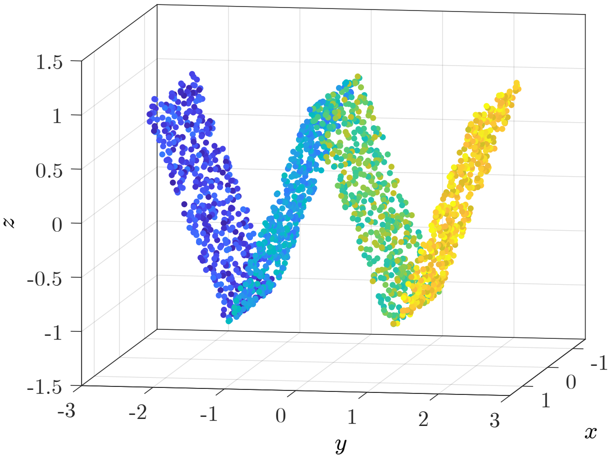

图 3: “W” 形状流形数据分布图

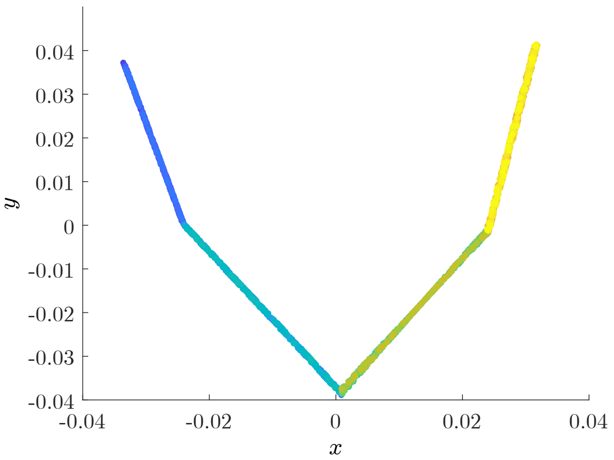

图 4: “W” 形状流形数据 40 近邻 LLE 降维结果

选择不同的近邻参数进行实验, 当近邻参数 k 较小时, 降维结果均为与图 4 类似的四条线段组成的折线, 而当近邻参数 k 约等于一个平面的点数时, 降维结果为四个平行四边形组成的平面图形. “W” 的四个面均由 500 个点构成, 若取 500 近邻, 则 LLE 降维结果如图 5 所示, 此时 LLE 将三维数据降维到二维平面, 并且保持着数据点在二维流形上的位置关系.

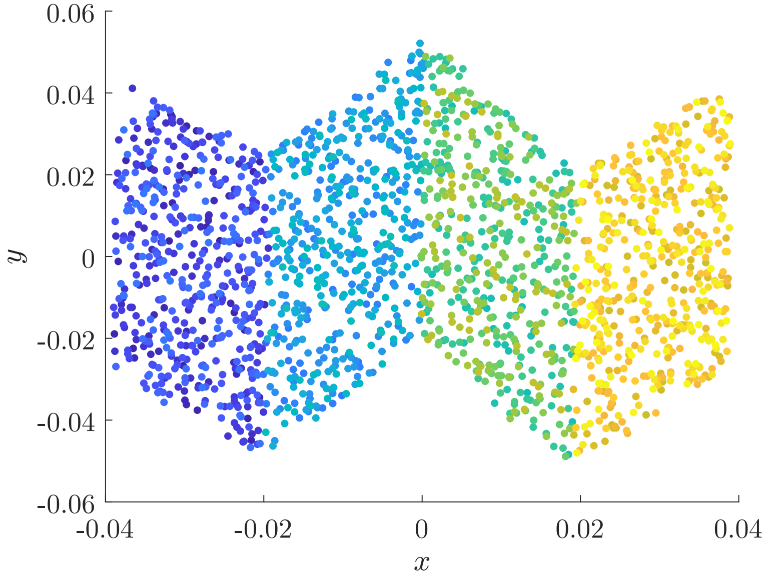

图 5: “W” 形状流形数据 500 近邻 LLE 降维结果

Further Reading

Whitening with PCA and ZCA

“A whitening transformation or sphering transformation is a linear transformation that transforms a vector of random variables with a known covariance matrix into a set of new variables whose covariance is the identity matrix, meaning that they are uncorrelated and each have variance 1. The transformation is called ‘whitening’ because it changes the input vector into a white noise vector.” [2]

Suppose we have n d-dimensional data points stored in $\boldsymbol{x} \in \mathbb{R}^{n\times d}$. The covariance matrix is $C(\boldsymbol{X}) = \frac{1}{n}\boldsymbol{X}^T\boldsymbol{X}$ and a whitening transformation is $\boldsymbol{Y} = \boldsymbol{W}\boldsymbol{X}$ where $\boldsymbol{W} \in \mathbb{R}^{d\times d}$ is the whitening matrix and $\boldsymbol{Y}$ is the transformed data with $C(\boldsymbol{Y}) = \boldsymbol{I}$. Theoretically, whitening transformation is not unique because a rotated whitening matrix $\boldsymbol{W}_2 = \boldsymbol{R}\boldsymbol{W}_1$ ($\boldsymbol{R}$ is an orthogonal matrix) is also a whitening matrix.

Suppose the eigenvalue decomposition for $C(\boldsymbol{X})$ is given by $C(\boldsymbol{X}) = \boldsymbol{E}\boldsymbol{D}\boldsymbol{E}^\top$ with eigenvectors in columns of $\boldsymbol{E}$ and eigenvalues on the diagonal of $\boldsymbol{D}$. For principal component analysis (PCA), the whitening matrix is calculated by $\boldsymbol{W}_{\text{PCA}} = \boldsymbol{D}^{-\frac{1}{2}}\boldsymbol{E}^\top$. For zero-phase component analysis (ZCA), the whitening matrix is $\boldsymbol{W}_{\text{ZCA}} = \boldsymbol{E}\boldsymbol{D}^{-\frac{1}{2}}\boldsymbol{E}^\top$. Multiplication by an orthogonal matrix can be seen as rotation and multiplication by a diagonal matrix can be seen as scaling. We can see that ZCA rotates the transformed vectors of PCA back to the original data space with the orthogonal matrix $\boldsymbol{E}$.

In deep learning, we know that batch normalization (BN) is a powerful trick to accelerate and stabilize the training of deep models. BN simply performs standardization for input feature maps. However, it has been shown that batch whitening (transform the input feature maps with a whitening transformation) further improves BN's optimization efficiency and generalization ability [3]. In batch whitening, the ZCA whitening is much better than the PCA whitening, read the work [3] for further reference of why this happens.

Optional: Construct a toy example, calculate the PCA results and the ZCA results and compare them to illustrate why ZCA is preferred.

Non-classical MDS

The classical derivation of MDS in the class assumed that the distance matrix is calculated by Euclidean distances of paired data points. However, in real applications, this matrix represents a set of dissimilarities which might not be Euclidean distances or not even distances at all. The MDS problem without the Euclidean assumption of distance matrix is called non-classical MDS. This is a generalization of the classical MDS and implemented as default MDS algorithm in Python library scikit-learn. Read the book [4] for solutions in this situation.

In addition, there is a more general form of MDS algorithm called non-metric MDS which aims to preserve the *rank-order* of the distances in the embedding space rather than their *values*. You can also find solutions for non-metric MDS in the same book [4] if you are interested.

参考文献

[1] Sam T. Roweis, lle.m - A simple matlab routine to perform LLE, Locally Linear Embedding (LLE) Code Page, `https://cs.nyu.edu/~roweis/lle/code.html`.

[2] Wikipedia contributors. ”Whitening transformation.” Wikipedia, The Free Encyclopedia. Wikipedia, The Free Encyclopedia, 15 Dec. 2020. Web. 27 Apr. 2021.

[3] Huang L, Yang D, Lang B, et al. Decorrelated batch normalization [C]// Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2018: 791-800.

[4] Boyarski A., Bronstein A. Multidimensional Scaling. In: Ikeuchi K. (eds) Computer Vision. Springer, Cham, 2020.