先端データ解析論
第十二回　レポート

37-196360　森田涼介

2019 年 7 月 16 日

宿題 1

$$\boldsymbol{W} \in \mathbb{R}^{n\times n},\ W_{i,i'} = W_{i',i} \tag{1}$$

$$\boldsymbol{D} = \mathrm{diag}\left(\sum_{i'=1}^{n} W_{i,i'}\right) \tag{2}$$

$$\boldsymbol{L} = \boldsymbol{D} - \boldsymbol{W} \tag{3}$$

についての固有値問題

$$\boldsymbol{L\psi} = \gamma \boldsymbol{D\psi} \tag{4}$$

の最小固有値 $\gamma_n = 0$ であり，対応する固有ベクトルは $\mathbf{1}$ であることを示す。

まず，$\boldsymbol{L}$ の最小固有値が 0 であることを示す。これは，全ての固有値が非負であること，つまり $\boldsymbol{L}$ が半正定値行列であることを示せばよい。よって，次式を示す。

$$^{\forall}\boldsymbol{\alpha} \in \mathbb{R}^n : \boldsymbol{\alpha}^{\mathrm{T}}\boldsymbol{L\alpha} \geq 0 \tag{5}$$

$\boldsymbol{\alpha}$, $\boldsymbol{L}$ を要素ごとに表すと，

$$\boldsymbol{\alpha} = \begin{bmatrix}\alpha_1 & \cdots & \alpha_n\end{bmatrix}^{\mathrm{T}} \tag{6}$$

$$\boldsymbol{L} = \begin{bmatrix} \left(\sum_{i'=1}^{n} W_{1,i'} - W_{1,1}\right) & -W_{1,2} & \cdots & -W_{1,n} \\ -W_{2,1} & \ddots & & \vdots \\ \vdots & & \ddots & \vdots \\ -W_{n,1} & \cdots & -W_{n,n-1} & \left(\sum_{i'=1}^{n} W_{n,i'} - W_{n,n}\right) \end{bmatrix} \tag{7}$$

となるので，

$$\begin{aligned}
\boldsymbol{L\alpha} &= \begin{bmatrix} \left(\sum_{i'=1}^{n} W_{1,i'} - W_{1,1}\right) & -W_{1,2} & \cdots & -W_{1,n} \\ -W_{2,1} & \ddots & & \vdots \\ \vdots & & \ddots & \vdots \\ -W_{n,1} & \cdots & -W_{n,n-1} & \left(\sum_{i'=1}^{n} W_{n,i'} - W_{n,n}\right) \end{bmatrix} \cdot \begin{bmatrix}\alpha_1 \\ \vdots \\ \alpha_n\end{bmatrix} \\
&= \begin{bmatrix} \left(\sum_{i'=1}^{n} W_{1,i'} - W_{1,1}\right)\alpha_1 - W_{1,2}\alpha_2 - \cdots - W_{1,n}\alpha_n \\ \vdots \\ -W_{n,1}\alpha_1 - \cdots - W_{n,n-1}\alpha_{n-1} + \left(\sum_{i'=1}^{n} W_{n,i'} - W_{n,n}\right)\alpha_n \end{bmatrix} \\
&= \begin{bmatrix} \left(\sum_{i'=1}^{n} W_{1,i'}\right)\alpha_1 - \sum_{i'=1}^{n} W_{1,i'}\alpha_{i'} \\ \vdots \\ \left(\sum_{i'=1}^{n} W_{n,i'}\right)\alpha_n - \sum_{i'=1}^{n} W_{n,i'}\alpha_{i'} \end{bmatrix} \\
&= \begin{bmatrix} \sum_{i'=1}^{n} W_{1,i'}(\alpha_1 - \alpha_{i'}) \\ \vdots \\ \sum_{i'=1}^{n} W_{n,i'}(\alpha_n - \alpha_{i'}) \end{bmatrix}
\end{aligned}$$

よって，

$$
\begin{aligned}
\boldsymbol{\alpha}^{\mathrm{T}}\boldsymbol{L}\boldsymbol{\alpha} &= \begin{bmatrix}\alpha_1 & \cdots & \alpha_n\end{bmatrix} \cdot \begin{bmatrix}\sum_{i'=1}^{n} W_{1,i'}(\alpha_1 - \alpha_{i'}) \\ \vdots \\ \sum_{i'=1}^{n} W_{n,i'}(\alpha_n - \alpha_{i'})\end{bmatrix} \\
&= \sum_{i=1}^{n} \alpha_i W_{i,i'}(\alpha_i - \alpha_{i'}) \\
&= \sum_{i,i'=1}^{n} W_{i,i'}\left(\alpha_i{}^2 - \alpha_i\alpha_{i'}\right) \\
&= \frac{1}{2}\sum_{i,i'=1}^{n} W_{i,i'}\left(2\alpha_i{}^2 - 2\alpha_i\alpha_{i'}\right) \\
&= \frac{1}{2}\sum_{i,i'=1}^{n} W_{i,i'}\left(\alpha_i{}^2 + \alpha_{i'}{}^2 - 2\alpha_i\alpha_{i'}\right) \\
&= \frac{1}{2}\sum_{i,i'=1}^{n} W_{i,i'}(\alpha_i - \alpha_{i'})^2 \\
&\geq 0
\end{aligned}
$$

となり，$\boldsymbol{\alpha}^{\mathrm{T}}\boldsymbol{L}\boldsymbol{\alpha} \geq 0$ が示された。これより，$\boldsymbol{L}$ の最小固有値が 0 であることがわかる。

次に固有値 0 に対応する固有ベクトルが $\mathbf{1}$ であることを示す。

$$
\boldsymbol{L}\boldsymbol{\psi} = \gamma\boldsymbol{D}\boldsymbol{\psi} = \mathbf{0} \tag{8}
$$

から，

$$
\begin{bmatrix}\sum_{i'=1}^{n} W_{1,i'}(\psi_1 - \psi_{i'}) \\ \vdots \\ \sum_{i'=1}^{n} W_{n,i'}(\psi_n - \psi_{i'})\end{bmatrix} = \begin{bmatrix}0 \\ \vdots \\ 0\end{bmatrix} \tag{9}
$$

となる。また，

$$
\boldsymbol{\psi}^{\mathrm{T}}\boldsymbol{L}\boldsymbol{\psi} = \frac{1}{2}\sum_{i,i'=1}^{n} W_{i,i'}(\psi_i - \psi_{i'})^2 = 0 \tag{10}
$$

である。これらと $W_{i,i'} \geq 0$ から，

$$
\psi_i = \psi_{i'} \qquad (i,\, i' = 1,\, \cdots,\, n) \tag{11}
$$

となる。これより，

$$
\boldsymbol{\psi} = \begin{bmatrix}\psi_1 & \cdots & \psi_n\end{bmatrix} = \psi_1\begin{bmatrix}1 & \cdots & 1\end{bmatrix} = \psi_1\mathbf{1} \tag{12}
$$

となる。よって固有値 0 に対応する固有ベクトルが $\mathbf{1}$ であることがわかる。

宿題 2

最近傍類似度に対するラプラス固有写像を実装する。

類似度行列を次のように定義する。

$$W_{i,i'} = \begin{cases} 1 & \text{if } \boldsymbol{x}_i \in \mathrm{kNN}(\boldsymbol{x}_{i'}) \text{ or } \boldsymbol{x}_{i'} \in \mathrm{kNN}(\boldsymbol{x}_i) \\ 0 & \text{otherwise} \end{cases} \tag{13}$$

なお，最近傍類似度を考えるので，$k = 1$ とする。ラプラス固有写像の埋め込み先 $\boldsymbol{\Psi}^{\mathrm{T}}$ は，

$$\boldsymbol{D} = \mathrm{diag}\left(\sum_{i'=1}^{n} W_{i,i'}\right) \tag{14}$$

$$\boldsymbol{L} = \boldsymbol{D} - \boldsymbol{W} \tag{15}$$

についての固有値問題

$$\boldsymbol{L}\boldsymbol{\psi} = \gamma \boldsymbol{D}\boldsymbol{\psi} \tag{16}$$

を解き，

$$\boldsymbol{\Psi}^{\mathrm{T}} = \begin{bmatrix} \psi_{n-1} & \psi_{n-2} & \psi_{n-m} \end{bmatrix}^{\mathrm{T}} \tag{17}$$

$$(\gamma_1 \geq \cdots \geq \gamma_n) \qquad \boldsymbol{\psi}_i^{\mathrm{T}} \boldsymbol{D} \boldsymbol{\psi}_i = 1 \tag{18}$$

とすればよい。

実験では 3 次元データを 2 次元に次元削減する。4 ページの Listing 1 にプログラムを示した。結果は図 1 に示した通りで，次元削減後の点群が 3 点に固まってしまうという形になった。原因としては，実装ミス以外で挙げるとすれば，最近傍類似度を 1,000 データ点に対して考えているから，$\boldsymbol{W}$ がスパースになりすぎているのがよくないのかもしれないと思ったが，解決には至らなかった。

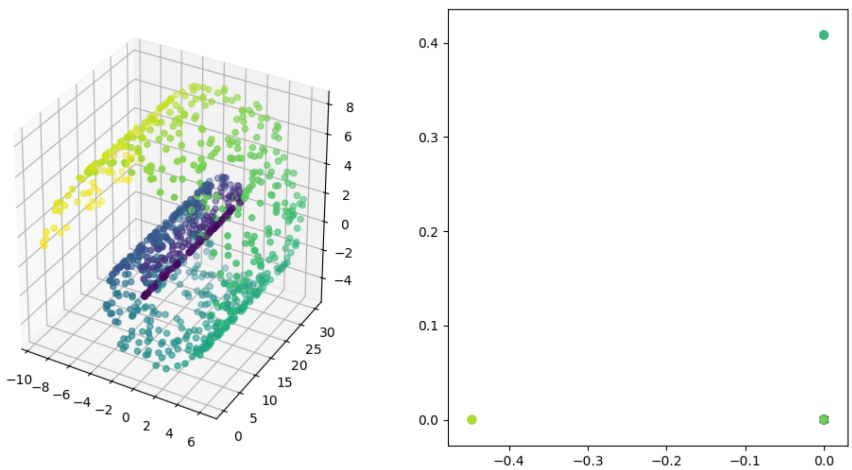

図 1: 結果

プログラム

実行環境と用いた言語・ライブラリを以下の表 1 に示す。

表 1: プログラムの実行環境

OS	:	Microsoft Windows 10 Pro (64bit)
CPU	:	Intel(R) Core(TM) i5-4300U
RAM	:	4.00 GB
使用言語	:	Python3.6
可視化	:	matplotlib ライブラリ

Listings 1: assignment2.py

```
# -*- coding: utf-8 -*-


import numpy as np
import scipy.linalg

import matplotlib
matplotlib.use('TkAgg')
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D


def generate_data(n=1000):
    a = 3. * np.pi * np.random.rand(n)
    x = np.stack([
        a*np.cos(a),
        30*np.random.random(n),
        a*np.sin(a),
        ], axis=1)
    return a, x


def similarity_matrix(x, k=1):
    n = x.shape[0]
    W = np.zeros((n, n))
    for i in range(n):
        x_i = x[i, :]
        W[i, i] = 1

        norms_i = np.sum((x_i - x)**2, axis=1)
        idx_nearest = np.argsort(norms_i)[1:1+k]

```

```
            W[i, idx_nearest] = 1
            W[idx_nearest, i] = 1
    return W


def train(x, d, k=1, eps=1e-8):
    W = similarity_matrix(x)
    D_diag = W.sum(axis=0)
    D = np.diag(D_diag)
    L = D - W

    eigen_values, eigen_vectors = scipy.linalg.eig(L, D)
    eigen_vectors = eigen_vectors.T

    indices = np.argsort(eigen_values)[::-1]
    eigen_values = eigen_values[indices]
    eigen_vectors = eigen_vectors[indices]
    #eigen_values[(-eps < eigen_values) & (eigen_values < eps)] = 0

    #eigen_vectors_reduced = eigen_vectors[(eigen_values>0), :][::-1]
    eigen_vectors_reduced = eigen_vectors[::-1][1:]

    Psi_T = eigen_vectors_reduced[:d]
    Psi = Psi_T.T
    return Psi


def visualize(x, z, a, path=None):
    fig = plt.figure(figsize=(12, 6))

    ax = fig.add_subplot(1, 2, 1, projection='3d')
    ax.scatter3D(x[:, 0], x[:, 1], x[:, 2], c=a, marker='o')

    ax = fig.add_subplot(1, 2, 2)
    ax.scatter(z[:, 1], z[:, 0], c=a, marker='o')

    if path:
        plt.savefig(str(path))
    plt.show()


def main():
    # settings
    n = 1000
    k = 1
    n_components = 2
    fig_path = f'../figures/assignment2_result.png'
```

```
    np.random.seed(1)


    # generate data
    a, x = generate_data(n)
    #print(a.shape, x.shape)


    # preprocess
    #mu = x.mean(axis=0)
    #x = x - mu


    # train
    z = train(x, d=n_components, k=k)


    # result
    print(f'#Data: {n}')
    print(f'z shape: {z.shape}')
    print(f'z = \n{z}')

    visualize(x, z, a, path=fig_path)


if __name__ == '__main__':
    main()
```