Homework 4 - Report

Chengyang Huang, Ziheng Zhang, Group 25

Implementation

We implemented algorithm described in the paper “On Spectral Clustering: Analysis and an algorithm” by the following steps.

- Given a set of points $S = s_1, \ldots, s_n$ that we want to cluster into k subsets and form A.

  We used `networkx` package to read and draw graphs and converted them into `np.array` data type.

  The results of visualisation regarding dataset 1 and dataset 2 are shown below.

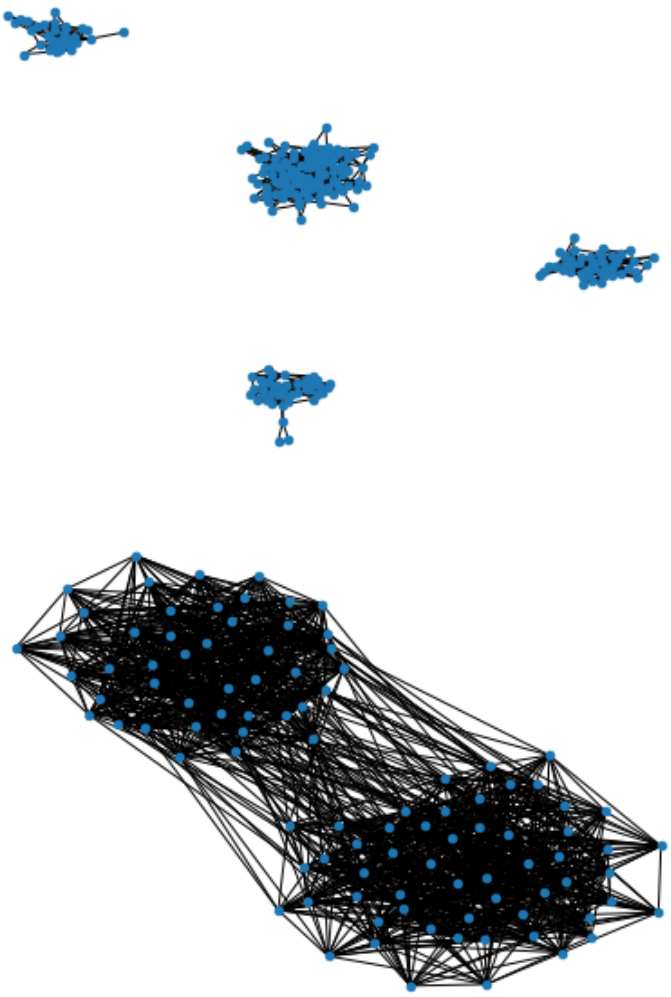


Fig.1 - Dataset 1 and 2.

Therefroe, we can for the affinity matrix (adjacent matrix) A.

```
A = nx.to_numpy_array(graph)
```

- Define D and L.

  D is a diagonal matrix generated by making a sum of each line in A.

$$D = \begin{pmatrix} \sum_{j=1}^{n} a_{1j} & 0 & 0 & 0 & 0 & 0 \\ 0 & \sum_{j=1}^{n} a_{2j} & & & & \\ 0 & & \ddots & & & \\ 0 & & & \ddots & & \\ 0 & & & & \ddots & \\ 0 & & & & & \sum_{j=1}^{n} a_{nj} \end{pmatrix} \tag{1}$$

```
D = np.diagflat(np.sum(A, axis=1))
```

$L = D^{-1/2}AD^{-1/2}$
$X^{-1/2} = (X^{1/2})^{-1}$

```
D_inv = np.linalg.inv(np.sqrt(D))
L = D_inv @ A @ D_inv
```

- Calculate the k largest eigenvectors and form X.

```
w, v = np.linalg.eigh(L)
```

```
X = v[:, -k:]
```

  - How to find k.

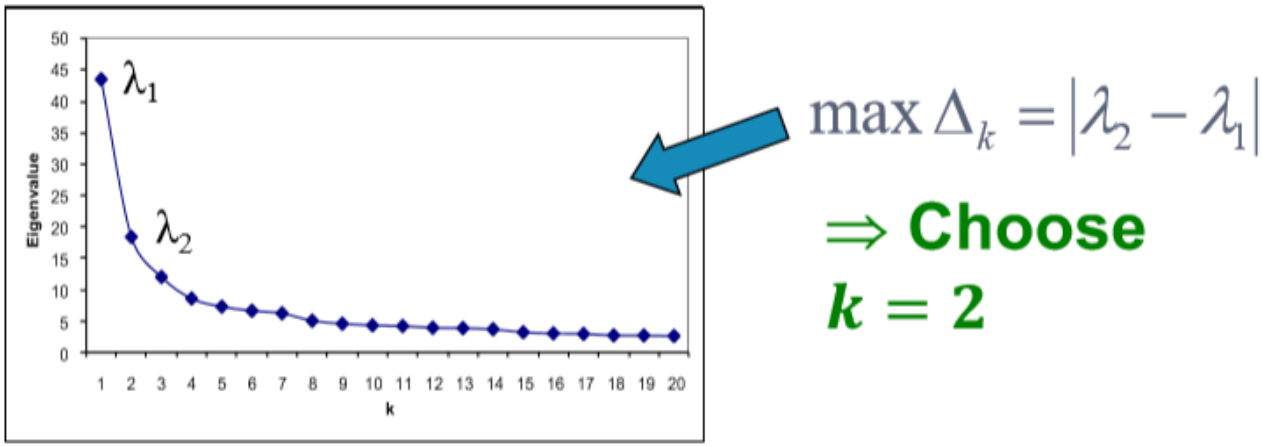


Fig.2 - Diff.

```
np.argmax(np.abs(np.ediff1d(w[::-1])))
```

`np.ediff1d` is used to calculate the differences between consecutive elements.

```
np.array([1, 2, 4, 7, 0]) => [1, 2, 3, -7]
```

- Normalise Y.
  $Y_{ij} = X_{ij}/(\sum_j X_{ij}^2)^{1/2}$

```
# Default: Frobenius Norm
Y = X / np.linalg.norm(X, axis=1, keepdims=True)
```

- Use K-means to cluster them.

```
res = KMeans(n_clusters=k).fit(Y).labels_
```

- Fiedler Value.

  the second-smallest eigenvalue of the Laplacian matrix L.
  $L = D - A$

```
values, vectors = np.linalg.eig(D-A)
```

Plot results are shown below.

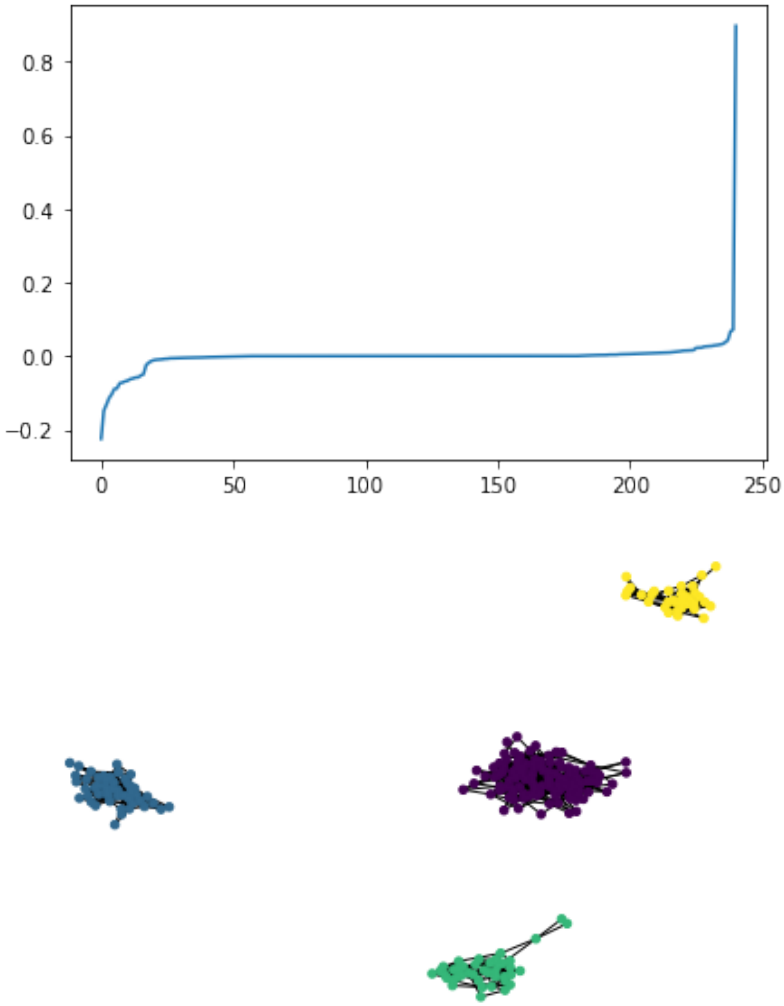
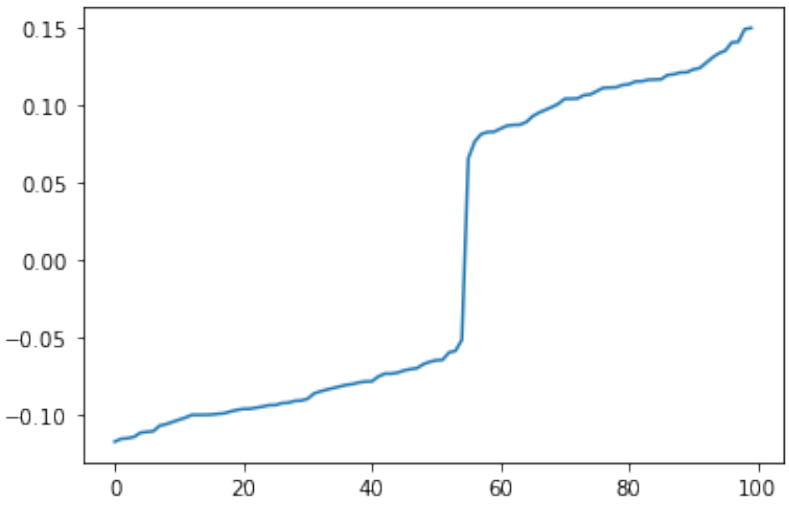


Fig.3 - Clustering Result 1.



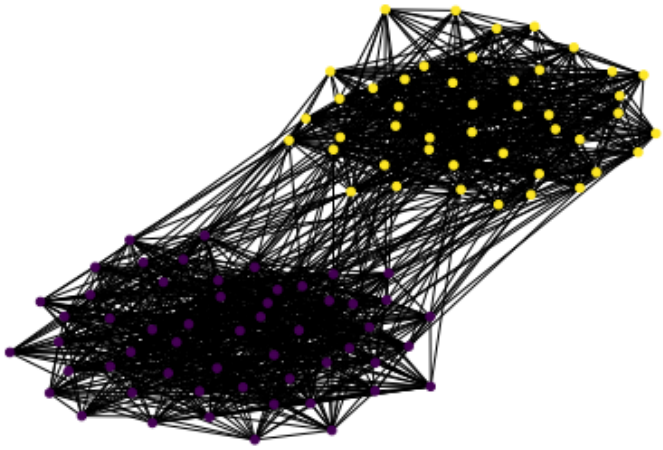


Fig.4 - Clustering Result 2.