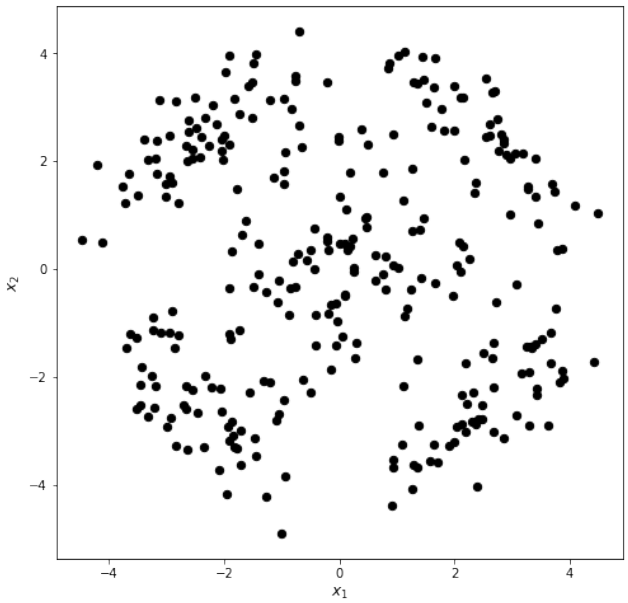
Homework 08 – Spectral Clustering

Importing Data Set

- After importing NumPy and Matplotlib libraries, and spatial function from the SciPy library, I've imported the data set and NumPy's genfromtxt() function.
- I've also visualized the initial, unclustered data set.



Euclidean Distance & Connectivity Matrix

- I've calculated the Euclidean distances between the pairs of data points using the function I wrote called "euclidean_distance". Using this function, I've first created "euc" matrix, which gave the Euclidean distances between each pair of data points, and then I created B matrix (connectivity/adjacency matrix) using the condition given in the homework description.

```
def euclidean_distance(x1, x2):
    return np.sqrt((x2[0] - x1[0])**2 + (x2[1] - x1[1])**2)
```

```
euc = np.array([[euclidean_distance(X[i], X[k]) for k in range(N)] for i in range(N)])

threshold = 1.25
B = np.array([[1 if euc[i, k] < threshold and i != k else 0 for k in range(N)] for i in range(N)])
```

- Then, I have visualized the connections between each data point, if they are connected, using the connectivity matrix.

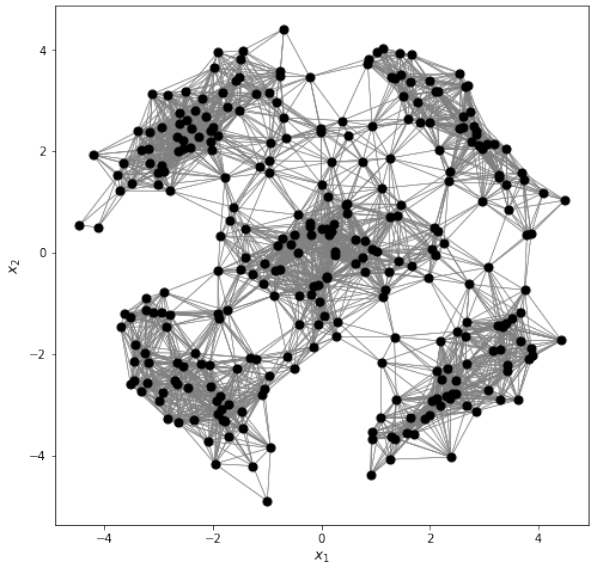
```
plt.figure(figsize=(8, 8))

for i in range(N):
    for k in range(N):
        if B[i, k] == 1 and i != k:
            plt.plot(np.array([X[i, 0], X[k, 0]]), np.array([X[i, 1], X[k, 1]]), linewidth=0.5, c="grey")

plt.scatter(X[:, 0], X[:, 1], s=50, color="k", zorder=3)

plt.xlabel("$x_1$", fontsize=12)
plt.ylabel("$x_2$", fontsize=12)

plt.show()
```



Degree, Laplacian & Normalized Laplacian Matrices

- Using the following formulas given in lecture notes, I have created degree, Laplacian and normalized (symmetric) Laplacian matrices.

$$d_{ii} = \sum_{j \neq i} b_{ij} \forall i$$

$$L = D - B$$

$$L_{SYMMETRIC} = I - D^{-\frac{1}{2}} * B * D^{-\frac{1}{2}}$$

```
D = np.zeros((300, 300), dtype=int)

for i in range(300):
    D[i, i] = B[i].sum()

L = D - B

D_new = np.zeros((300, 300))

for i in range(300):
    D_new[i, i] = D[i, i]**(-1/2)

L_symmetric = np.eye(300) - (D_new @ B @ D_new)
```

Eigenvectors of Normalized Laplacian Matrix

- Using NumPy's eig function from the linalg module, I have calculated the eigenvectors and eigenvalues of the normalized Laplacian matrix, and then chose the 5 smallest eigenvectors, putting them into the Z matrix.

```
R = 5
eig_vals, eig_vecs = np.linalg.eig(L_symmetric)
Z = eig_vecs[:,np.argsort(eig_vals)[1:R+1]]
```

- I've also created the initial centroids using 29th, 143rd, 204th, 271st, and 277th rows of the Z matrix.

```
centroids = np.vstack([Z[28], Z[142], Z[203], Z[270], Z[276]])
print(f"Initial Centroids:\n{centroids}")
```

```
Initial Centroids:
[[-0.00325886 -0.03070096  0.12747516  0.05551091  0.03543487]
 [ 0.0342538   0.11685073 -0.02269928  0.06127445 -0.00404365]
 [ 0.01188869  0.03876559 -0.00024575 -0.04717368  0.01128642]
 [ 0.01512635  0.03295704  0.00463725 -0.08546651  0.01848106]
 [-0.00092625  0.02155608  0.00332934 -0.07374296  0.01282762]]
```

K-Means Algorithm

- Then, using a slightly modified version of the k-means algorithm from the lab11, I've ran the k-means algorithm on Z matrix with the initialized centroids.

```
def update_memberships(centroids, X):
    D = spa.distance_matrix(centroids, X)
    memberships = np.argmin(D, axis = 0)
    return(memberships)

def update_centroids(memberships, X):
    centroids = np.vstack([np.mean(X[memberships == k,:], axis = 0) for k in range(K)])
    return(centroids)
```

```
memberships = update_memberships(centroids, Z)
iteration = 1
K = 5

while True:
    print(f"Iteration #{iteration}")
    print(f"Centroids:\n{centroids}")
    print(f"Memberships:\n{memberships}\n")
    old_centroids = centroids
    centroids = update_centroids(memberships, Z)
    if np.alltrue(centroids == old_centroids):
        break
    old_memberships = memberships
    memberships = update_memberships(centroids, Z)
    iteration += 1

centroids = update_centroids(memberships, X)
```

Visualization of Clustering Results

- Lastly, I've visualized the clustering results obtained by the k-means algorithm ran on the Z matrix.