Spectral Clustering

Alessandro Ciarrocchi, s344430
Politecnico di Torino

***Abstract*—The project focuses on comparing clustering methods across different datasets using MATLAB. The code is divided into five sections, where clustering techniques are applied to 2D and 3D data, leveraging normalized Laplacian matrices and a custom implementation of the inverse power method for eigenvalues computations. The analysis includes spectral clustering and highlights the differences between manual and built-in approaches.**

I. Introduction to the datasets

The datasets used in this project consist of three files:

- *Spiral.mat*: containing 312 two-dimensional points along with a column indicating cluster labels; the dataset span from 3 to 31.5 on the x1 axis and from 2.9 to 31.65 on the x2 axis (Figure 1).
- *Circle.mat*: containing 900 two-dimensional points; the dataset span from -4 to 10 on the x1 axis and from -10 to 6 on the x2 axis (Figure 2).
- *clustering_datasets.mat*: containing 396 three-dimensional points (Figure 3).

It is interesting to note the difference in scale between the datasets Spiral.mat and Circle.mat, as this could influence the performance and outcomes of the clustering methods applied.

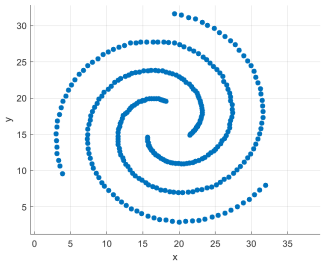

Fig. 1. Spiral dataset distribution

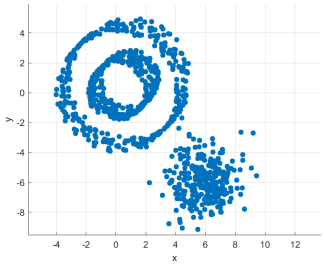

Fig. 2. Circle dataset distribution

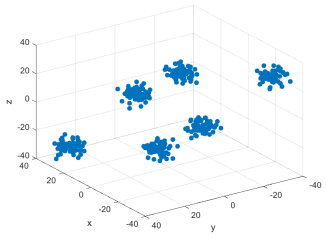

Fig. 3. 3D dataset

II. Proposed approach

As previously mentioned, the code is divided into five sections. The analysis begins with the main part of the code, where the files are uploaded, and the points from the datasets Circle.mat and Sprial.mat are plotted on a two-dimensional plane to visualize the spectral distribution. A similarity matrix is then constructed using the formula

$$s_{i,j} = exp(-\frac{||X_i - X_j||^2}{2\sigma^2}),$$

followed by the creation of an adjacency matrix W_{knn} for each $k = [10, 20, 40]$ (where k represents the number of neighbors). To achieve this, a custom function was implemented that applies the k-nearest neighbors (KNN) method, where for each point, the k nearest neighbors are identified based on the distance metric, and an adjacency matrix is constructed by linking each point to its neighbors. Then, the degree matrix D is constructed as a diagonal matrix, where each diagonal elements represents the sum of the weights of the connections for each node. Laplacian matrix L is then computed as the difference between D and W_{knn}. These matrices are stored in sparse format to reduce memory consumption and enhance computational efficiency. The Laplcacian matrix's eigenvalues provide insight into the connectivity of a graph. Specifically, the number of eigenvalues that are close to zero corresponds to the number of connected components in the graph: they were computed for both the datasets and for all the k values. The three smallest eigenvalues and their corresponding eigenvectors were computed using MATLAB's function. The eigenvalues were stored in the vector $Lambda$, and the eigenvectors were organized in the matrix U. Subsequently, spectral clustering was performed by applying MATLAB's k-means function to the matrix U and setting a maximum of 1000 iterations. Afterward, the original data points were assigned to the corresponding clusters, and the results were visualized in a two-dimensional space. Two additional clustering techniques were then applied: k-means (this time applied on the original data points) and DBSCAN, for which several combinations of initial parameters were tested (one for each part of the code) to enable a thorough comparison of its performance on both datasets. In this part, the parameters of the column A of the Table I were used.

In the second case, the same code previously implemented was used, with the key difference being the application of the symmetrically normalized Laplacian matrix L instead of the original one. DBSCAN was applied with the parameters of the column B in Table I.

In the third part, the smallest eigenvalues were computed by implementing the inverse power method manually, which utilizes power iteration to approximate the smallest eigenvalues and their corresponding eigenvectors of the Laplacian matrix. The clusters identified vary with each execution of the code due to the random choice of the initial vector used inside

the inverse power method function. Also here, DBSCAN was applied using other parameters (column C in Table I)..

In the fourth part, the code from the previous section was reapplied using the symmetrically normalized Laplacian matrix and updated DBSCAN parameters (columns D in Table I).

The last part was dedicated to the analysis of the three-dimensional dataset: after applying inverse power method to the Laplacian matrix, six clusters were identified using spectral clustering, DBSCAN (column E in Table I) and k-means clustering.

Parameter	A	B	C	D	E
epsilon	3	1	1	0.5	5
minpts	5	5	10	5	10

TABLE I
DBSCAN PARAMETERS FOR DIFFERENT CASES

III. RESULTS

This section is dedicated to presenting and analyzing the obtained results.

1) *First part*: for the spiral dataset, spectral clustering performed excellently for $k = 10$ (Figure 6) and $k = 40$, while for $k = 20$ the results were generally accurate, although in some cases, a few points were misclassified. In contrast, for the Circle dataset, spectral clustering yielded good results only for $k = 10$ and $k = 20$ (Figure 7), but failed to produce meaningful clusters for $k = 40$. Overall, the method demonstrated relatively stable performance, with minor exceptions. Regarding DBSCAN, using the parameters from column A of Table I, the algorithm performed exceptionally well on the Spiral dataset but proved ineffective for Circle (where all the points are considered core points). Finally, k-means was also applied, but it yielded poor clustering results in both cases (Figure 8 and Figure 9). In addition, for the Spiral dataset, the connected componets identified through the eigenvalues of the Laplacian matrix are 2 for $k = 10$ and 1 for both $k = 20$ and $k = 40$. In contrast, for the Circle dataset, 3 connected components are found for $k = 10$, while only 1 component is identified for $k = 20$ and $k = 40$.

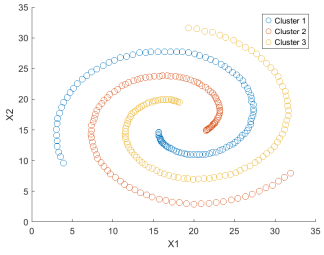

Fig. 4. Perfect spiral clustering

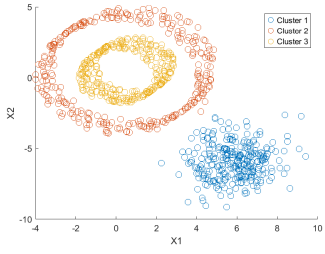

Fig. 5. Perfect circle clustering

2) *Second part*: the spectral clustering applied to the Spiral dataset demonstrated excellent performance for all values of k, achieving a consistently accurate clustering with high stability. Conversely, the DBSCAN algorithm applied with the parameters of the column B in the Table (I) did not perform well on Spiral (Figure 10). For the Circle dataset, the results for the spectral clustering remained unchanged compared to the previous analysis; the same applies to DBSCAN, with the only difference being the addition of a few noise points (Figure 11). Additionally, for the Spiral dataset, three main components are correctly identified for each value of k. In contrast, for the Circle dataset, the number of connected components varies: 3 are found for $k = 40$, 2 for $k = 20$, and only 1 for $k = 10$.

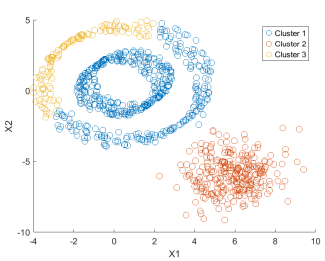

Fig. 6. Spectral Clustering for circle with k = 40

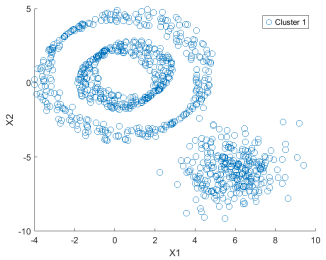

Fig. 7. DBSCAN for circle with Parameters A

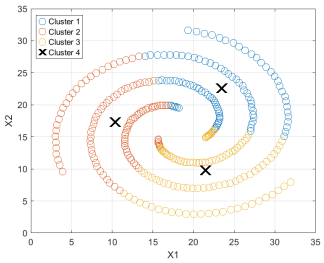

Fig. 8. Spiral k-means

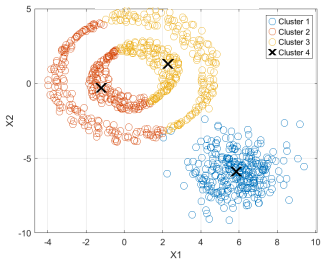

Fig. 9. Circle k-means

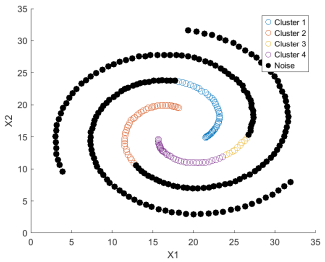

Fig. 10. DBSCAN spiral Parameters B

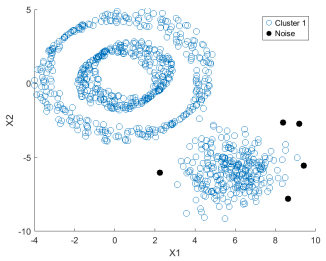

Fig. 11. DBSCAN circle Parameters B

3) *Third part*: the spectral clustering on the Spiral dataset perform perfectly for $k = 10$, with only occasional minor misclassifications; however, is is consistently inaccurate for $k = 20$ (Figure 12) and highly variable for $k = 40$, where many points are classified correctly, but others are misclassified (Figure 13). The DBSCAN algorithm on Spiral shows worse results compared to the previous part, failing to capture the correct structure effectively (Figure 14). For the Circle dataset, spectral clustering performs flawlessly for $k = 20$, while for

$k = 40$ the results are similarly erroneous as in previous runs, and for $k = 10$, the clustering is entirely incorrect (Figure 16). The DBSCAN algorithm identifies only two clusters (Figure 15). Overall, the algorithm for Circle demonstrates reasonable stability, in contrast to the significant variability observed for Spiral. Connected components are the same identified in the first part.

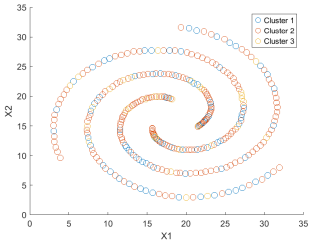

Fig. 12. Spectral clustering for k = 20

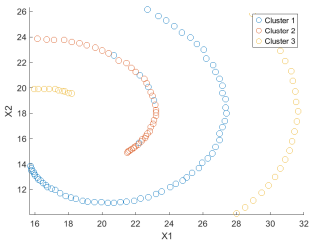

Fig. 13. Zoom of spectral clustering for k = 40

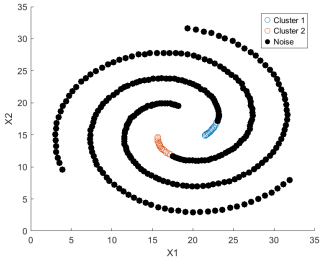

Fig. 14. DBSCAN for Parameters C

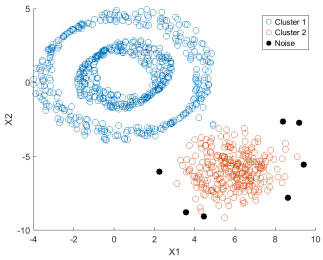

Fig. 15. DBSCAN for Parameters C

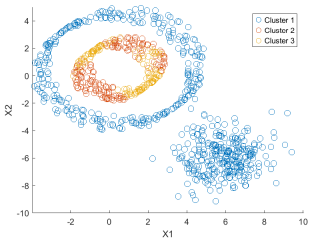

Fig. 16. Spectral clustering for k = 10

4) *Fourth part*: spectral clustering on the Spiral dataset remains perfect and highly stable; however, DBSCAN continues to perform similarly to previous tests, showing limited success. For the Circle dataset, the stability observed in earlier parts diminishes, leading to erroneous clusters for $k = 40$ and $k = 20$ (Figure 17) . Nevertheless, the clustering results for $k = 10$ are generally good, with only a few misclassifications. The DBSCAN algorithm, on the other hand, identifies four clusters along with a few noise points, marking an improvement over previous result (Figure 18. Connected components are the same identified in the second part.
5) *Fifth part*: for the three-dimensional dataset, the k-means algorithm performed flawlessly, producing accurate clustering results (Figure 19). Spectral clustering was precise

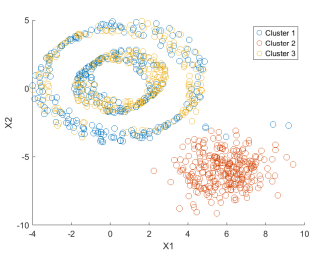

Fig. 17. Spectral clustering for k = 20, 40

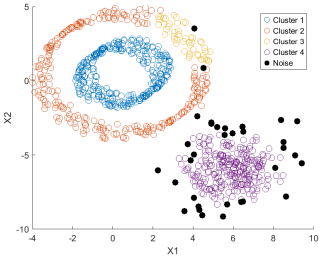

Fig. 18. DBSCAN with Parameters D

for $k = 40$ (Figure 22), but showed more variability for $k = 20$ and $k = 10$ (Figure: 21), with results oscillating between correct and incorrect classifications. DBSCAN yielded excellent results, with the exception of a few noise points identified, but overall, it contributed to reliable clustering performance (Figure 20).

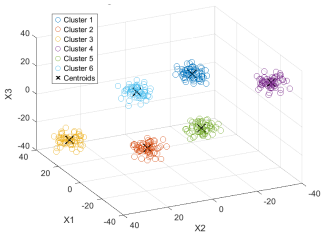

Fig. 19. 3D k-means

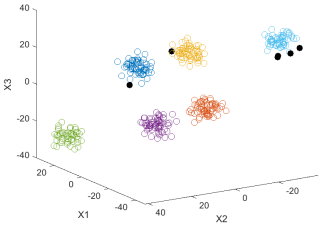

Fig. 20. DBSCAN with Parameters E

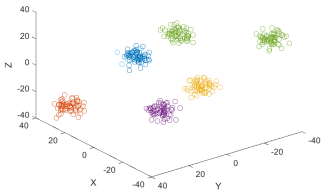

Fig. 21. Spectral clustering k = 10, 20

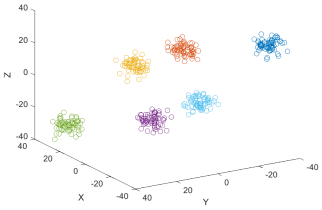

Fig. 22. Spectral clustering k = 40

IV. Discussion and Summary

The presented results are primarily influenced by two sources of randomness: the inverse power method, which involves the random execution of the code, and the k-means algorithm, which relies on a random inizialization of centroids. It is important to note that, although the MATLAB function uses the 'plus' option by default to efficiently select centroids, a random component, albeit not excessively large, remains present and should be taken into account. The subsequent analysis focuses on assessing these variations and examining how different parameters and approaches either mitigate or amplify these fluctuations.

Regarding the spiral dataset, since the clusters have an irregular shape, in the third part of the code we observe that the results are notably more satisfactory for a value of $k = 10$: his

configuration better preserves the local connections between nearby points, without interfering with points from other clusters, as seen in the cases with $k = 20$ and $k = 40$. In that latter case, a more global interpretation emerges, but it becomes more challenging to accurately identify the membership of central points, which are closer to one another. Good results are clearly observed in the first part of the code, where the randomness introduced by $k = 40$ is offset by the absence of the inverse power methos. It is also noteworthy that the use of the Laplacian matrix significantly improves outcomes, consistently yielding perfect clusters. This can be explained by the fact that the normalized Laplacian matrix, in contrast to the unnormalized version, performs better in handling clusters of varying densities (such as those in the spiral dataset), which are denser at the center. Furthermore, with the Laplacian normalized matrix were provided much more stable results.

In the Circle dataset, setting $k = 40$ allows for a clear distinction between the dense cluster of points in the bottom-right and the two concentric circular clusters. However, this choice of k proves insufficient for accurately separating the points within the circular clusters, as they remain incorrectly partitioned. For $k = 10$ and $k = 20$ the first two part of the code yield excellent results. However, the same cannot be said for the sections where the inverse power method is applied. When using the inverse power method with the unnormalized Laplacian matrix, the results are accurate for $k = 20$ but incorrect for $k = 10$; conversely, with the normalized matrix, the opposite occurs. This phenomenon can be explained by the idea that normalization enchances clustering performance by providing a more localized prospective. However, as the algorithm shifts towards a more global view, increasing the conncetivity between points, it struggles to accurately distinguish the clusters within the two concentric circle, leading to a loss of segmentation quality.

For the DBSCAN algorithm, the best results in the Spiral dataset are achieved using the parameters from column A in the Table I, which provide an optimal balance. Decreasing the radius or increasing minPoints leads to a scenario where only the densest central regions are identified as clusters, while all other points are considered as noise. In contrast, for the Circle dataset, the parameters from columns A and B cause DBSCAN to classify all points as core points, resulting in a single large cluster. By relaxing the parameters, the number of core points reduces, while the number of border points increase: this adjustment allows for the formation of more distinct and well-separated clusters, albeit at the cost of introducing a small amount of noise, which remains within an acceptable range.

K-means does not yield satisfactory results for the Spiral and Circle datasets, as it is not well-suited for identifying clusters with non-globular shapes. This is evident from the fact that the only well-predicted cluster lies in Circle dataset and it is the one separated from the two concentric rings, as it is the only region with a roughly circular shape. In contrast, K-means performs effectively on the three-dimensional dataset, where it successfully identifies all six clusters without difficulty. A similar conclusion can be drawn for the DBSCAN method, with the exception of a few points classified as noise. However, this noise remains minimal and does not significantly impact the clustering performance. For the spectral clustering of the three-dimensional dataset, excellent performance with $k = 40$ are observed, whereas the predictions for $k = 10$ and $k = 20$ are highly variable and often incorrect. The likely explanation for this behavior is similar to what was discussed with the Circle dataset, where a lower number of clusters complicates the correct identification of the data's underlying structure.

Additionally, for both datasets, when the unnormalized Laplacian matrix was used, the identified connected components were generally incorrect, except for the spiral dataset with $k = 10$. In this case, the situation improves with the normalized Laplacian matrix, where the connected components are consistently identified as three for all values of k. However, for the Circle dataset, a satisfactory result is achieved only for $k = 10$.