Exercise 4 - Clustering

Lirane Bitton - 200024677

20/12/18

1 Theoretical Questions

1.1 Convergence of k-means

K-means algorithm is mainly composed of two parts: the clustering step and the centroid step.

In the first step, each point is clustered to the nearest centroid such that $C(k) = argmin_k(||x - \mu_k||^2)$. Since the algorithm is iterative the cost function can only be minimized or not change at this step.

In the second step we chose a centroid will minimise the cost for its cluster ($\sum_{x \in C_i} ||x - \mu_i||^2$). Here also at each iteration the cost function can only decrease

It's important to state that k-means algorithm run over a space of possible clustering which is finit since in order to cluster n point into k cluster there are at most k^n possibilities. Hence since the space of possible cluster is finite and the iterative algorithm is minimizing it's cost function at each step then the algorithm must converges in a finie number of steps.

1.2 Suboptimality of k-means

1. Let's define our dataset as point in 2d in the euclidean space forming a rectangle where the initialisation centroid are on the middle of the longer sides.

DataSet: {(0,0), (0,2), (1,0), (1,2)} and centroids initialisation: {(0,1),(1,1)} where the optimal solution is : {(0.5,0), (0.5,2)}

2. In the same way let's take the 2D unit square where different optimal clustering can be obtained depending on which pairs of vertex.

DataSet: {(0,0), (0,1), (1,0), (1,1)} and the following cluster: C1={(0,0), (0,1)} and C2={(1,0), (1,1)} or C1={(0,0), (1,0)} and C2={(0,1), (1,1)}

1.3 Initializations of k-means

1.

$$p_{good} = k! \prod \alpha_i$$

$$\mathbb{E}[\#trials] = \frac{1}{p_{good}} = \frac{1}{k! \prod \alpha_i}$$

2.

$$\mathbb{E}[\#trials] = \frac{1}{k! \prod \alpha_i} \geq min_{\alpha_i} \frac{1}{k! \prod \alpha_i} \geq \frac{1}{k! max_{\alpha_i}(\prod \alpha_i)}$$

Now we have an optimization problem where $\prod \alpha_i$ is our objective subject to $\sum \alpha_i = 1$. Using the lagrangian to solve:

Formalized:

$$argmax_{\alpha_i} \prod \alpha_i$$

$$s.t. \sum \alpha_i = 1$$

$$\rightarrow argmax_{\alpha_i} log(\prod \alpha_i) = argmax_{\alpha_i} \sum log(\alpha_i)$$

$$\rightarrow L(\alpha, \lambda) = \sum log(\alpha_i) - \lambda(\sum \alpha_i - 1)$$

$$\frac{\partial L}{\partial \alpha_i} = \frac{1}{\alpha_i} - \lambda \rightarrow \alpha_i = \frac{1}{\lambda}$$

Hence all the α_i are uniformly distributed then $\lambda = k$ and the lower bound is:

$$\mathbb{E}[\#trials] \geq \frac{1}{k! max_{\alpha_i}(\prod \alpha_i)} = \frac{1}{k! \prod \frac{1}{k}} = \frac{k^k}{k!}$$

3.It's not a plausible number of start trials. K-Means actually often works because we use different technique for initialization which reduce this number of start trials, such as K-Means ++ learned in class.

2 Practical Exercise

For the practical exercise I used the following dataset:

As synthetic datasets, I used the provided APML and circles datasets and also added one generated using make_blobs function from scikit which generate gaussians (for this purpose I configured it to have 4 centers in 2D with 200 samples)

Then I used the biological dataset provided to show the performance of my implementation on higher dimension dataset.

Finally in the t-SNE part I followed the recommendation to play with MNIST dataset and as toy dataset I generated a high dimensional dataset using make_blobs function parameterized with 30 feature having 5 centers and generating 700 samples.

Attached to this document you cna find the code with a main function well separated and documented to answer each part of the exercise with visualisations.

2.1 k-means - synthetic data

k-means seems to cluster the gaussian data perfectly but for the other synthetic dataset where the 2D euclidean proximity is not enough to cluster point together we can observe how badly the classification is performed even if the algorithm converge (but with very high values of MSE).

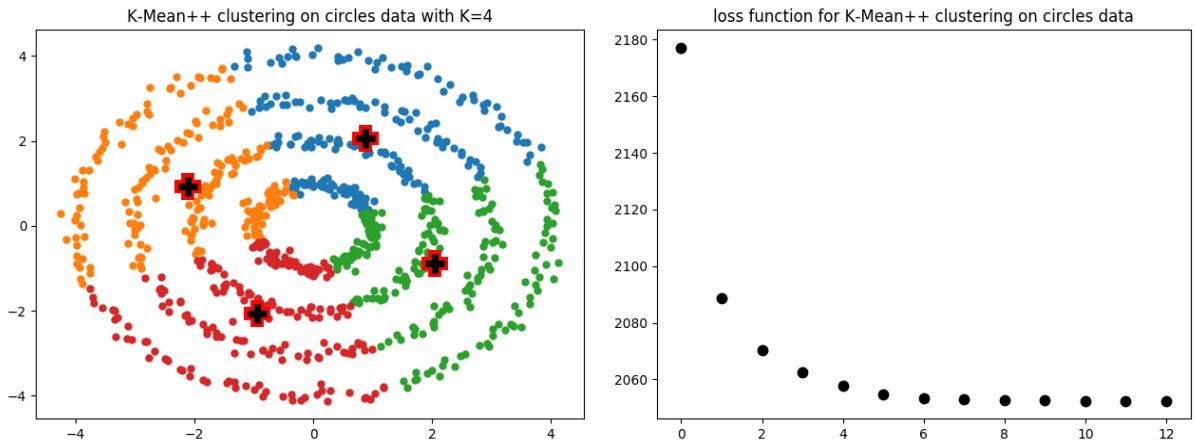

Figure 1: k-means ++ on circles dataset

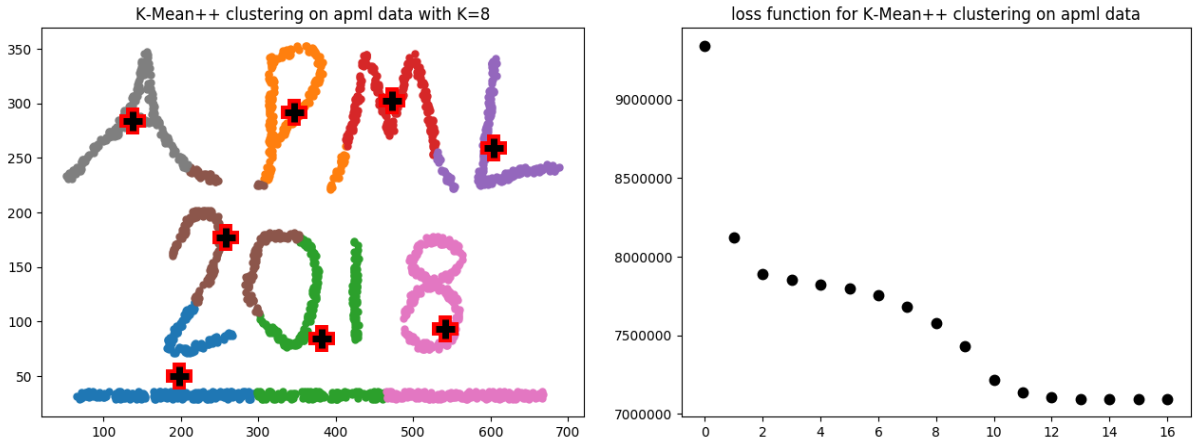

Figure 2: k-means ++ on APML dataset

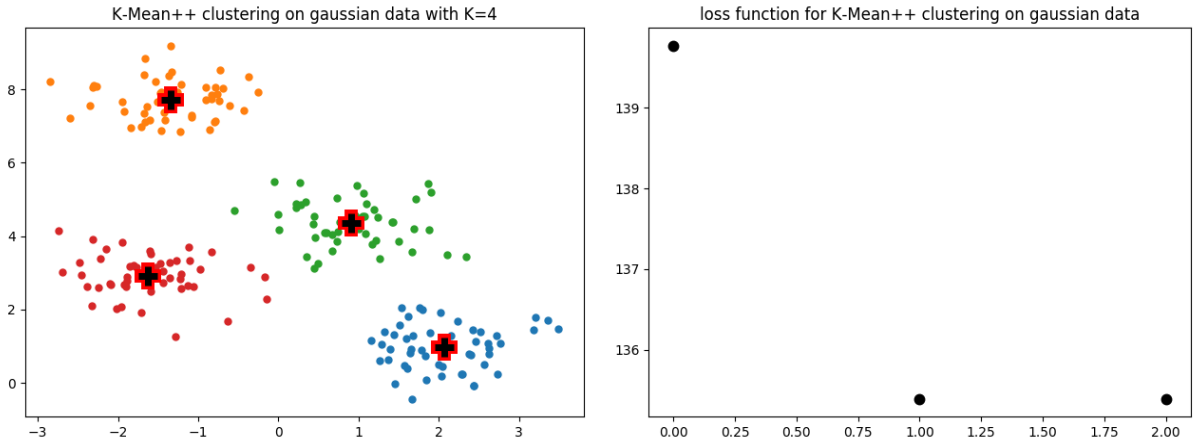

Figure 3: k-means ++ on gaussian dataset

2.2 spectral -synthetic data

Using the spectral clustering technique on the synthetic data we can see that the algorithm perform well in clustering for each dataset. Note that since the clustering is performed on the sorted eigenvectors hence the loss function reach its convergence point very fast after two steps. In the same way since the clustering is no performed on distances we can denote the strange points got as centroid which are almost the same for each cluster.

2.2.1 MNN

The Mnn parameter was tuned manually and found that for 10 neighbors the best clusters was obtained.

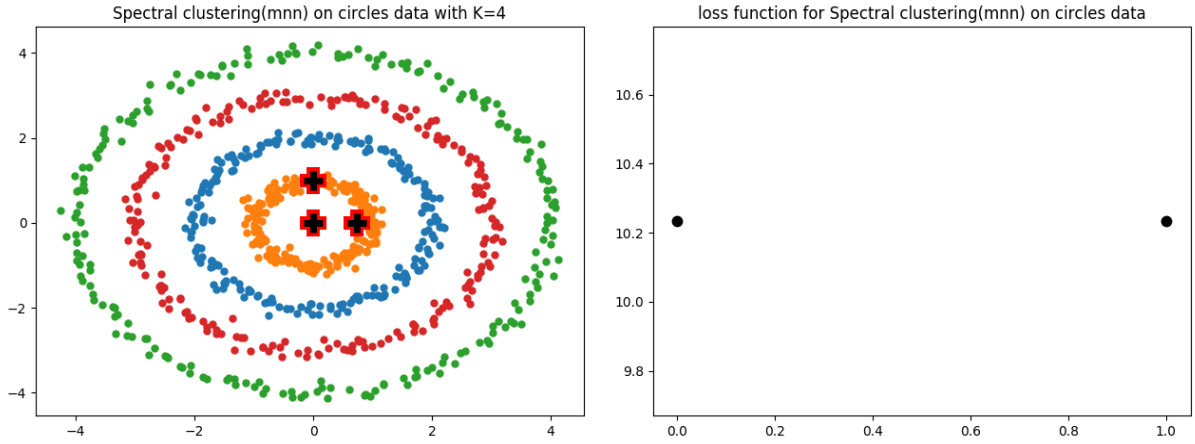

Figure 4: spectral on circles with mnn = 10

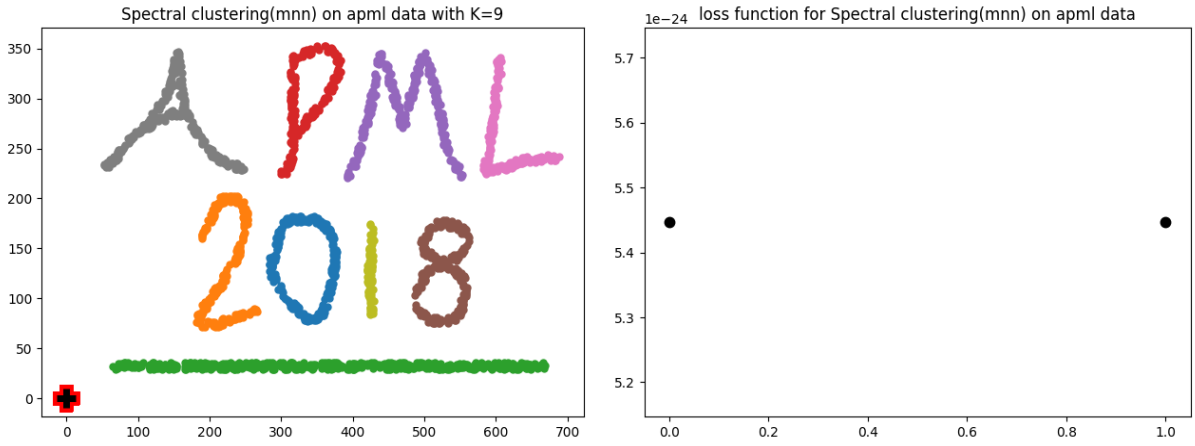

Figure 5: spectral on APML with mnn = 10

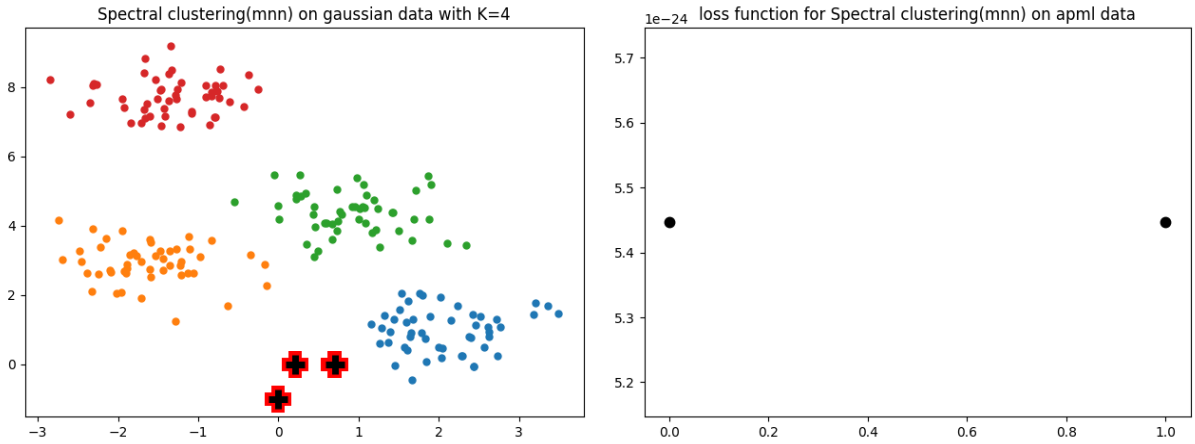

Figure 6: spectral on gaussian with mnn = 10

2.2.2 Heat Kernel

For the Heat Kernel I wrote an estimator function which actually perform the technique stated in class to compute the histogram of distances and then take of the first or second percentile as sigma value. This outperformed well on synthetic data but less on the biological data where some hand tuning was required in order to see any effective cluster.

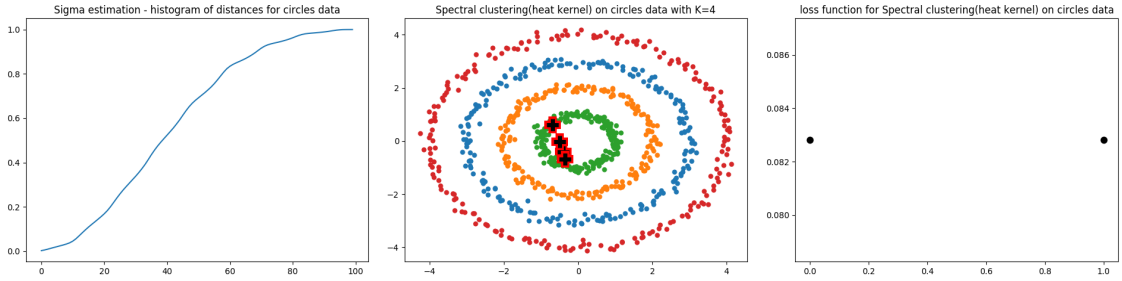

Figure 7: spectral on circles dataset with sigma estimated to first percentile

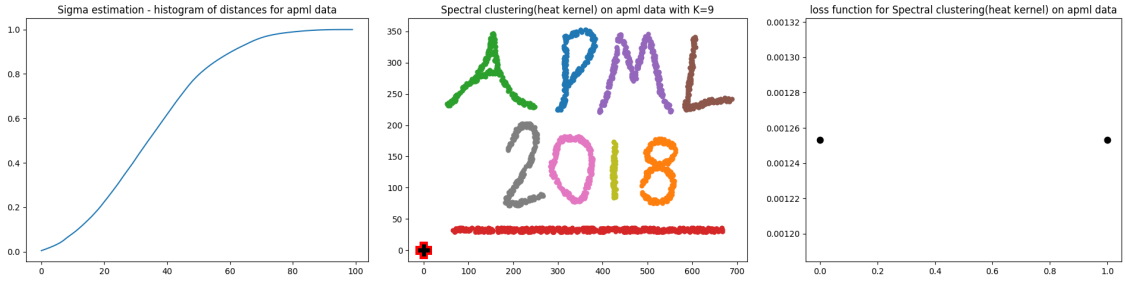

Figure 8: spectral on APML dataset with sigma estimated to first percentile

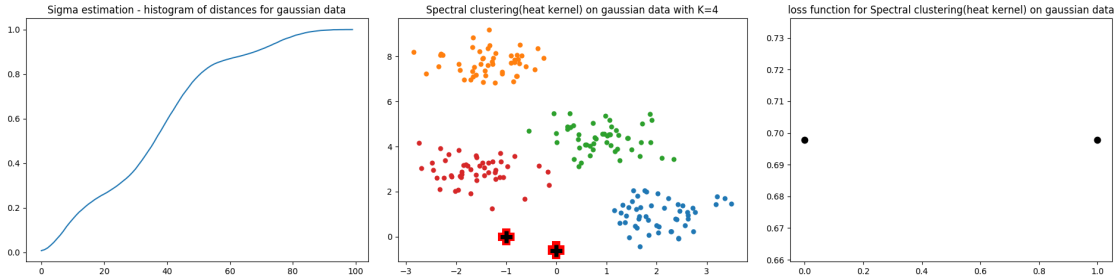

Figure 9: spectral on gaussian datasetestimated to second percentile

2.2.3 Plot similarty graph

I attached the similarity graph for the APML dataset and for the gaussian one where we can see pretty results from the uncluster data to the clustered one. We can see more structures appearing on the diagonal. The circle similarity matrix didn't actually show any structure after clustering and I'm assuming that is due to the circular and symmetric for of the dataset.

Actually it can be a good representation for clustering unless the data is symetric and then the information is quite inexistant.

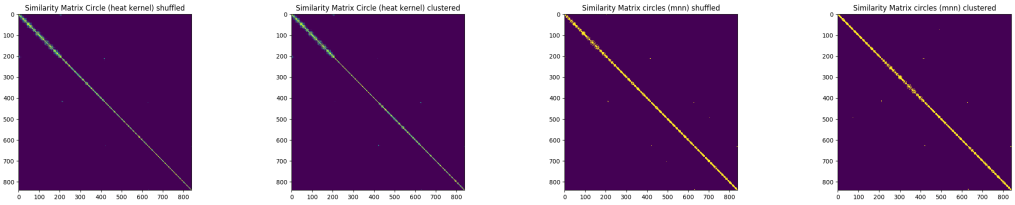

Figure 10: Similarity Matrix circles dataset

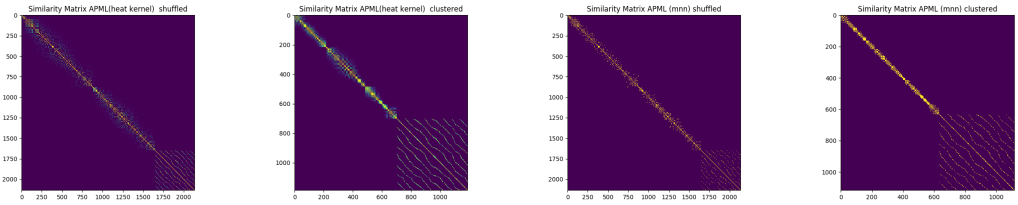

Figure 11: Similarity Matrix APML dataset

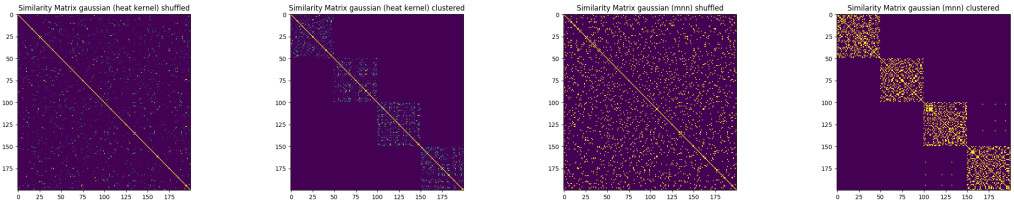

Figure 12: Similarity Matrix Gaussian dataset

2.3 Elbow and Silhouette for k choosing

Here are the elbow plottings for each one of the synthetic dataset and we can observe the elbow form obtained. But I might stipulate here that it's not really easy to understand from those graph or even by coding which is the right K to chose. For example for circles 4 or 5 can be chosen and will give almost same MSE loss function but th only way to distinguish between them is by visualizing the cluster which is wrong with k=5.

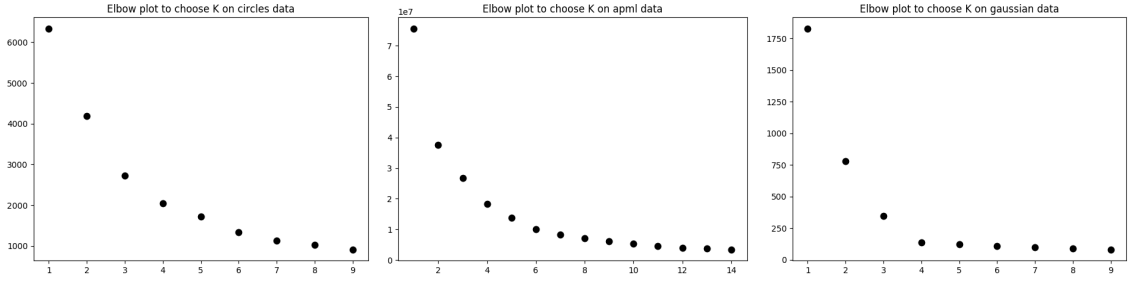

Figure 13: Elbow Method

I chose to plot the silhouette of the right k for each cluster. I could have had other option to show the differences(but there are already too much graphs). Here we can see that the clusters are

well separated in sizes and the mean of these silhouette are getting the best result hence they were chosed.

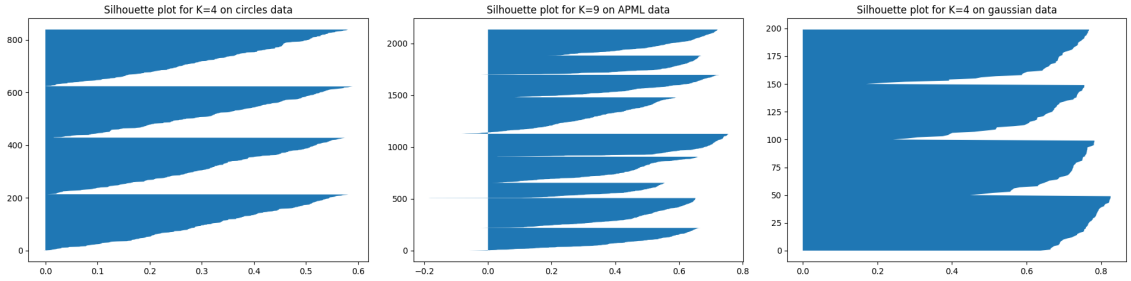

Figure 14: Silhouette Method

2.4 Eigen Gap

It was hard to define the first gap using code. This was done using a heuristic that the first difference between two eigenvalues over 50 will be the right k. Performed somehow well on the synthetic data but completely failed on bio data where it reach 500 as first k (then I just limited it to 20)

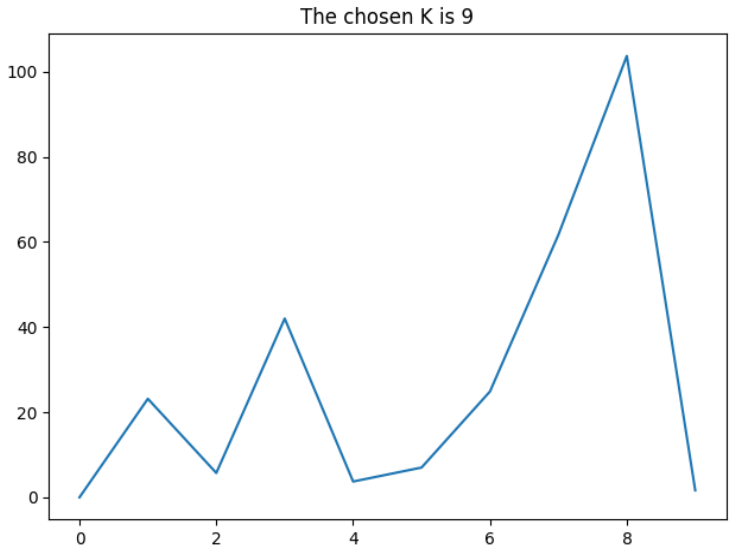

Figure 15: Eigen gap plotting values for APML

2.5 Biological Data

2.5.1 k-means++

k-means gave outstanding results on the biolgical data since the methods to estimate k performed as expected as well as the kmeans ++ itself. We can clearly distinguish 4 cluster in the plotted data where the first cluster has some yellowish line then the second are more darker in the same part. The third cluster is completely highlighted and finaly the fourth one comprises black parts in its lines. I decided to chose 4 clusters by visualizing the data and also the silhouette plots.

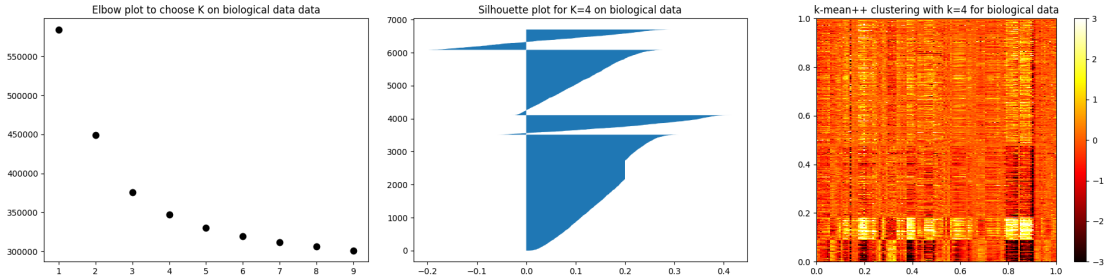

Figure 16: k-means ++ on biological data

2.5.2 spectral

Compare to the k-mean algorithm which performed well as expected in theory, here som hand tuning were required in order to get a proper clustering. The sigma estimation based on distance actually failed and can be explained by the fact that euclidean distance doesn't provide a right metric to compare this data. For the k value I just picked the same as for the k-means since I knew that it work well.

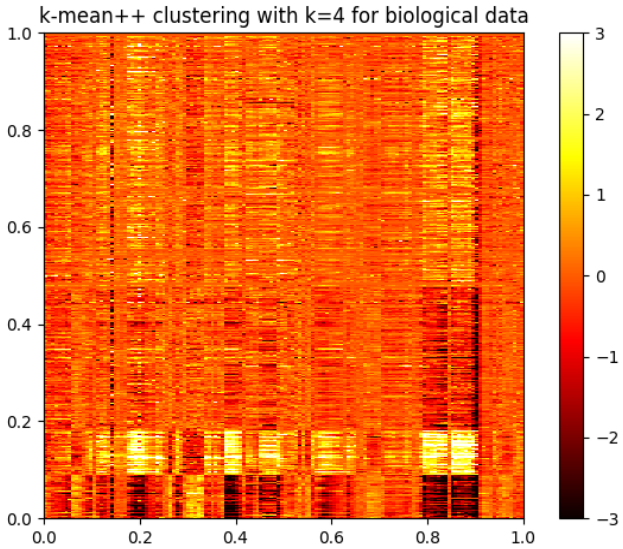

Figure 17: spectral clustering with k=4 and sigma=15

2.6 t-SNE

I played here with the t-SNE and pca projections method and definitely t-SNE capture local structure better that PCA. In the last example showed here in 3D we can think that PCA representation is better that t-SNE but actually I plots in debugger the values and they seem to be overlapping but they are separated on z-axis. Here it's only a 2D picture of 3D projection hence this looks not well seprated but in fact t-SNE gives outstandings results.

2.6.1 MNIST

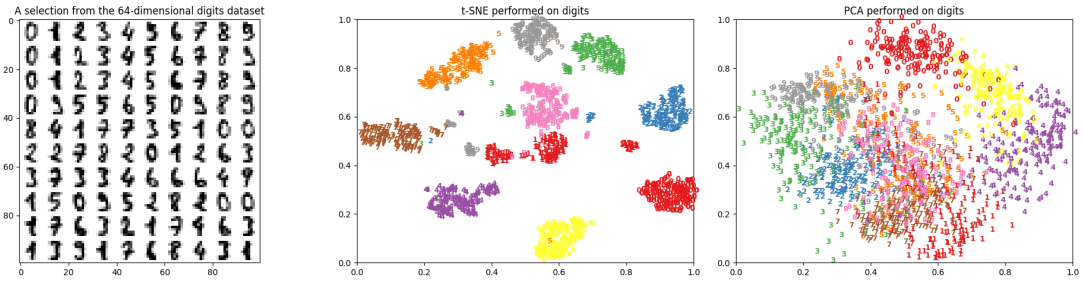

Figure 18: MNIST Dataset 2D projection

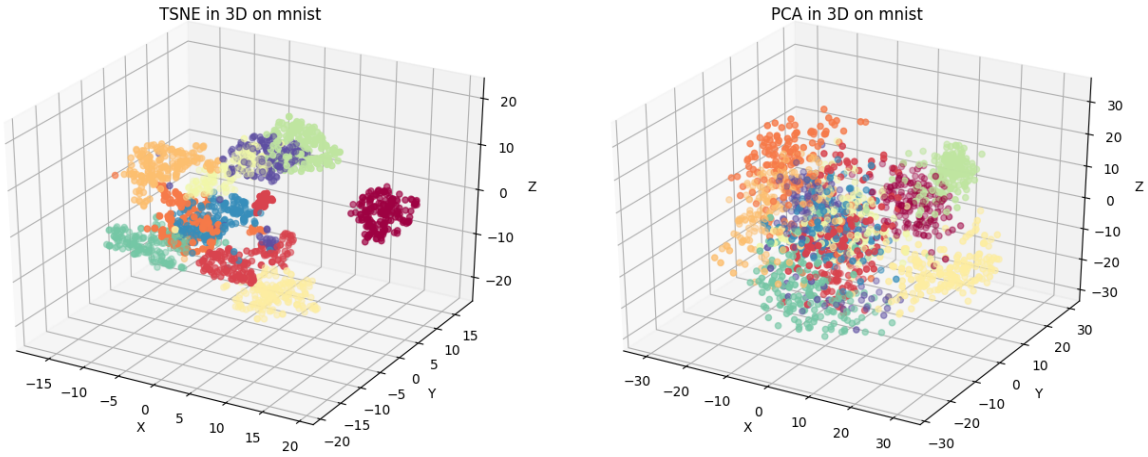

Figure 19: MNIST Dataset 3D projection

2.6.2 toy dataset

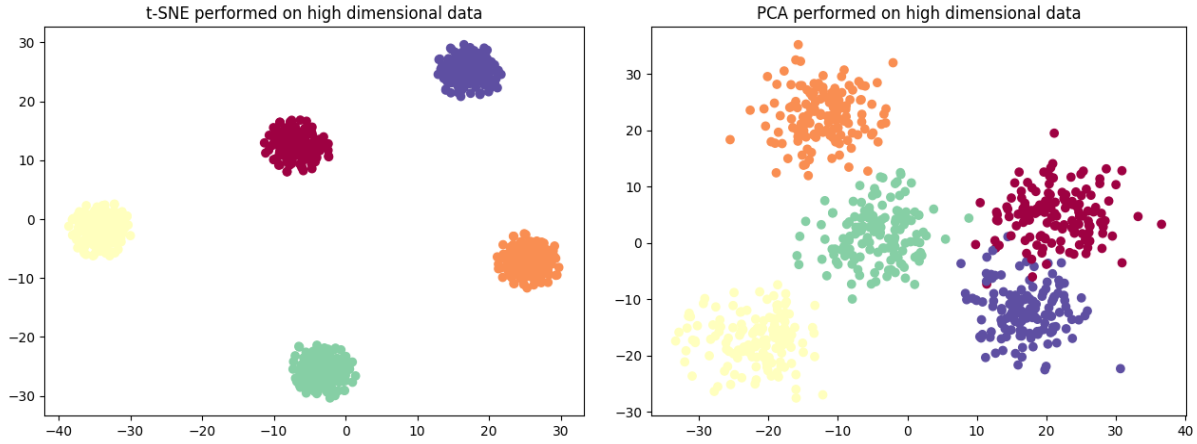

Figure 20: toy dataset 2D projection (from 30 features)

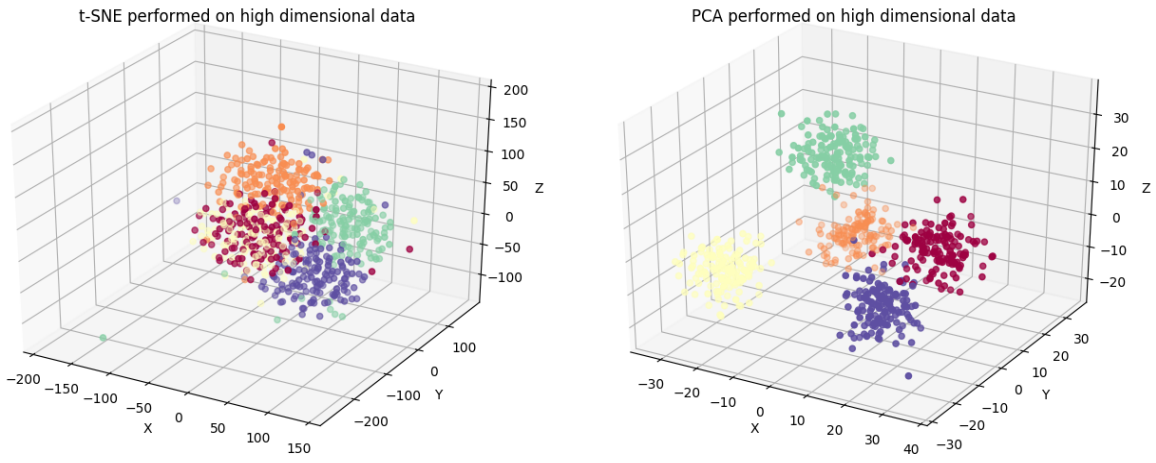

Figure 21: toy dataset 3D projection (from 30 features)