Advanced Machine Learning -Practical 2- Projection and Clustering Part 2

Professor: Aude Billard
Assistants: Guillaume de Chambrier, Nadia, Denys

Spring Semester 2016

1 Introduction

Another alternative to find the number of clusters, K, is to use the eigenvalues given by Laplacian Eigenmaps. Indeed, or the similarity graph is not fully connected and so the multiplicity of the eigenvalue $\lambda = 0$ gives us an estimation of K, or looking at the smallest eigenvalues we can have this estimation (keeping number of eigenvalues which are almost zero). In a first part of the tutorial, we will look into the following :

- Evaluate Laplacian Eigenmaps as means of deducing the number of clusters present in a dataset.
- Compare Laplacian Eigenmaps with kPCA solution seen yesterday on the same dataset.

Another difficulty is that the struture of the data can be a manifold and so applying clustering methods to the data can lead to get clusters which are not representative of this structure. Here is another point where dimensionality reduction techniques are useful since they enable to obtain in a low-dimension space a projection of the data with possibly a linear separation between the different clusters. Applying simple clustering algorithm, like k-means, can at this moment be used to find the clusters. In a second part, we will so look into :

- Dimensionality Reduction Technique as means of discovering manifolds in data.
- Apply clustering algorithm to find clusters.
- Evaluation of the performance after different projections using F-measure for semi-supervised clustering

2 ML_toolbox

ML_toolbobx contains a set of matlab methods and examples for learning about machine learning methods. You can download ML_toolbobx from here: [link] and the matlab scripts for this tutorial from here [link]. The matlab scripts will make use of the toolbox.

Before proceeding make sure that all the sub-directories of the ML_toolbox including the files in **TP2** have been added to your matlab search path. This can be done as follows in the matlab command window:

```
>> addpath(genpath('path_to_ML_toolbox'))
>> addpath(genpath('path_to_TP2'))
```

To test that all is working properly you can try out some examples of the toolbox; look in the **examples** sub-directory.

3 Laplacian Eigenmaps to determine the number of clusters

3.1 Laplacian Eigenmaps

The Laplacian Eigenmaps is a non-linear projection technique which is based on the eigenvalue decomposition of a scaled similarity matrix $L = D - S$. If we perform an eigenvalue decomposition of this matrix $L = V\Lambda V^{\mathrm{T}}$, where $V \in (N \times N)$ are the eigenvectors α and $\Lambda \in (N \times N)$ is a diagonal matrix containing the eigenvalues, we can then order the eigenvalues by increasing order : $\lambda_1 = 0 \leq \lambda_2 \leq ... \leq \lambda_N$, where N is the number of datapoints. Looking at the multiplicity of eigenvalue 0 or more generally at the eigenvalues which are almost 0 provides information about the partioning of the similarity graph built on the data and so indication about the number of clusters we could find.

3.2 Questions

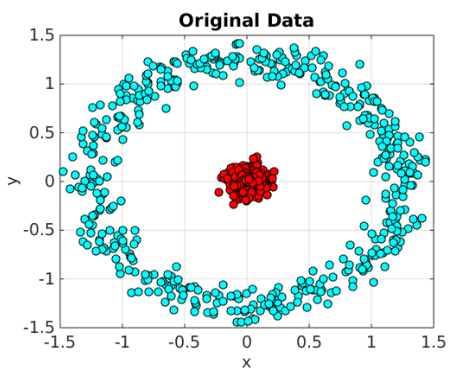
Your task is to find the number of clusters present in the following datasets:

Circles	:	2D data set of non linearly separable clusters.
Breast-cancer-Wisconsin	:	Medical dataset taken from the UCI database.
House-votes	:	Voting patterns between republicans and democrats, also UCI database.
Digits	:	8×8 digit images.

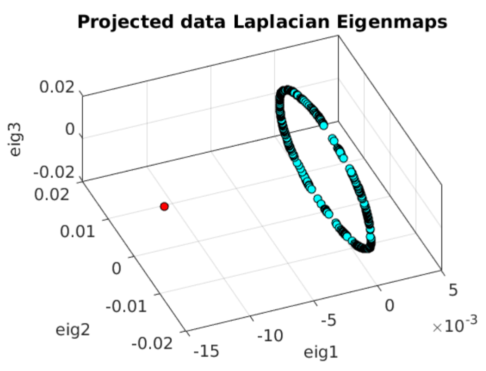
You will be compare the use of Laplacian Eigenamaps to techniques of previous practicals in order to find the number of clusters K.

3.2.1 Circle clusters

Open **TP2_Laplacian_kPCA_comparison.m** and you will find a detailed descriptions in the script of the steps you should take. You first generate a data set of circles or

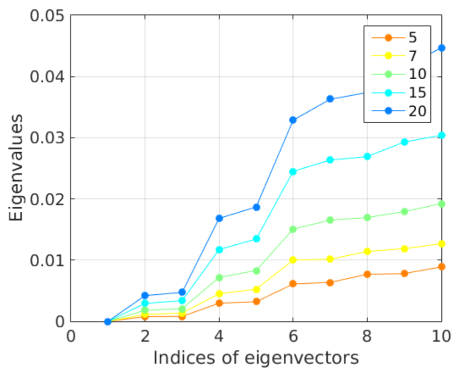


(a)

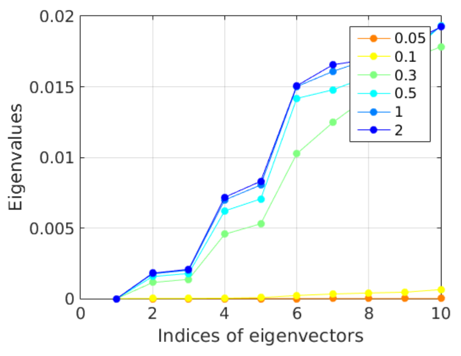


(b)

Figure 1: (a) Original circle dataset. (b) Projected data set with a Gaussian kernel function with $\sigma^2 = 0.4$.



(a)



(b)

Figure 2: (a) Eigenvalues for different numbers of neighbors : We see that this parameter doesn't have an influence on the search for good K in this case since it doesn't change the aspect of the curve but just the scale. (b) Eigenvalues for different widths of the kernel : If the variance is too small (< 0.3), many of the lowest eigenvectors are close to 0 and it's difficult to determine K

spheres depending on the dimension you choose for the original data Figure 1(a). Then you run successively kPCA and Laplacian Eigenmaps (with a chosen kernel width, neighborhood for Laplacian Eigenmaps and number of eigenvectors to retain). The result of the projection with Laplacian Eigenmaps is illustrated in Figure 1(b).

Q: How many clusters did you find with Laplacian Eigenmaps ? Is it sensitive to hyper-parameters ? The problem is still to select the appropriate variance of the kernel function and also the appropriate number of neighbors now. We did a grid search over the width and neighborhood. In the Figure 2, we can see the results. In this case the number of neighbors doesn't have an influence but we need to have a sufficiently great (≥ 0.3) variance in order to observe a gap in the curve which enables to determine the number of clusters. If not, all the eigenvalues are close to 0 and it's hard to get K. Here we can estimate that the multiplicity of the eigenvalue 0 is three and so $K = 3$.

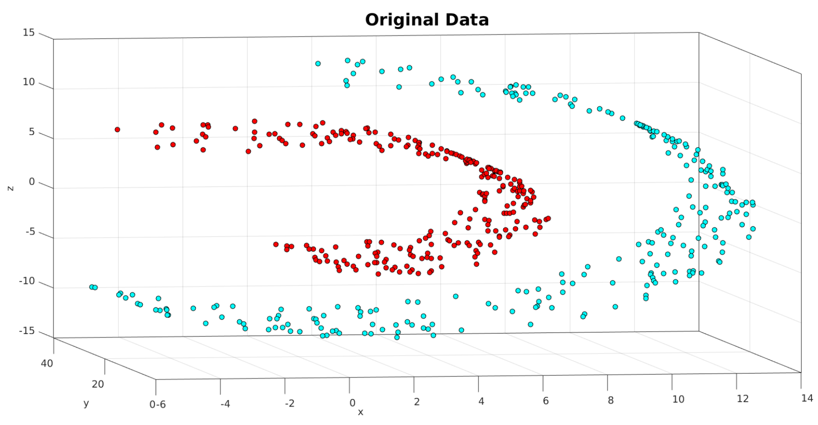


Figure 3: 3D view of the original dataset with two clusters

Q: How many clusters had you found with kPCA, with AIC and BIC with k-means ? We ran AIC and BIC on the original dataset for an optimal number of clusters of 6-7, and on the kPCA projected data for an optimal number of clusters of 3-4. The information given by kPCA was that there is not much more than two clusters present. In this toy example the K estimated looking at the eigenvectors of kPCA gives us the best prediction. What happens if we use real dataset ?

3.2.2 Datasets clusters

You will be trying to do the same now for the following three real datasets: (1) *Breast-cancer-Wisconsin*, (2) *House-votes*, (3) *Digits*.

How many clusters did you find with Laplacian Eigenmaps ?

Compare with the result given by AIC, BIC and kPCA

4 Projection techniques for clustering

4.1 Using projection techniques to cluster data

We will study how projection techniques can be used to project the data so that we could apply clustring algorithm (kmeans) to fingd clusters. For this purpose, we will use a dataset called Swiss Roll (for the typical rolled cake) which has been sampled so that we could have two different clusters we want to find.

4.2 Tasks

Your task will be to find a good projection technique for the swiss roll example in Figure 3 in order to obtain a separation of the dataset we can easily cluster. You will compare different projection techniques for this and then apply the k-means algorithm to try clustering the data.

The projection techniques you will use are :

- PCA
- kPCA
- Isomap
- Laplacian Eigenmaps

4.2.1 Projection

Open **TP2_projection_comparison.m** which contains all the steps you will have to follow. You first generate a data set of the two-clusters Swiss roll as in Figure 3 and plot it. Then you can first run PCA and display the result and then each projection technique one after another. For kernel PCA, you have to choose the kernel and the hyper-parameters. For Isomap and Laplacian Eigenmaps you have to choose the number of neighbors used to compute the graph or put *'adaptative'* to get an automatic selection. The results of each projection with good parameters is illustrated in Figure 4. Tuning the parameters of each projection technique you will have a look at how they influence the projection of the data.

Which projection enables to have a separation of the clusters ? See Figure 4
PCA and Kernel PCA doesn't work to get a projection of the two clusters because they don't find the manifold hidden in the structure of the data.

Using Isomap and Laplacian Eigenmaps which build a graph for the neighborhood of each are here very useful since points in a manifold have a neighborhood which is homeomorphic to the euclidian space.

4.2.2 Clustering

Once you will have found a good projection of the data, you will try to apply the KMeans algorithm for each projection. You can see the function implemented for each projection techniques. Try to repeat the algorithm several times on the same projection to see the effect of the random selection of centers at the beginning for some of them. At the end, there are different sections where you can estimate the F1-score for different parameters of the projections. Try to find a good range and see the influence of these hyperparameters.

Compare the average and standard deviation of the F1-score after different projections (Think about changing the hyper parameters too) See Figure 5

Can you explain the difference observed in the F-measure between Laplacian Eigenmaps and Isomap ? The difference in precision is due to the random initialization of the centers of the clusters. Indeed, for Laplacian Eigenmaps, if the initial centers belongs to the same part of the plan (X positive or X negative), k-means will tend to separate the plan along the Y axis because each cluster is far from the other along the X axis and is sparse along the y axis. That's why the F-measure is not very high in mean.

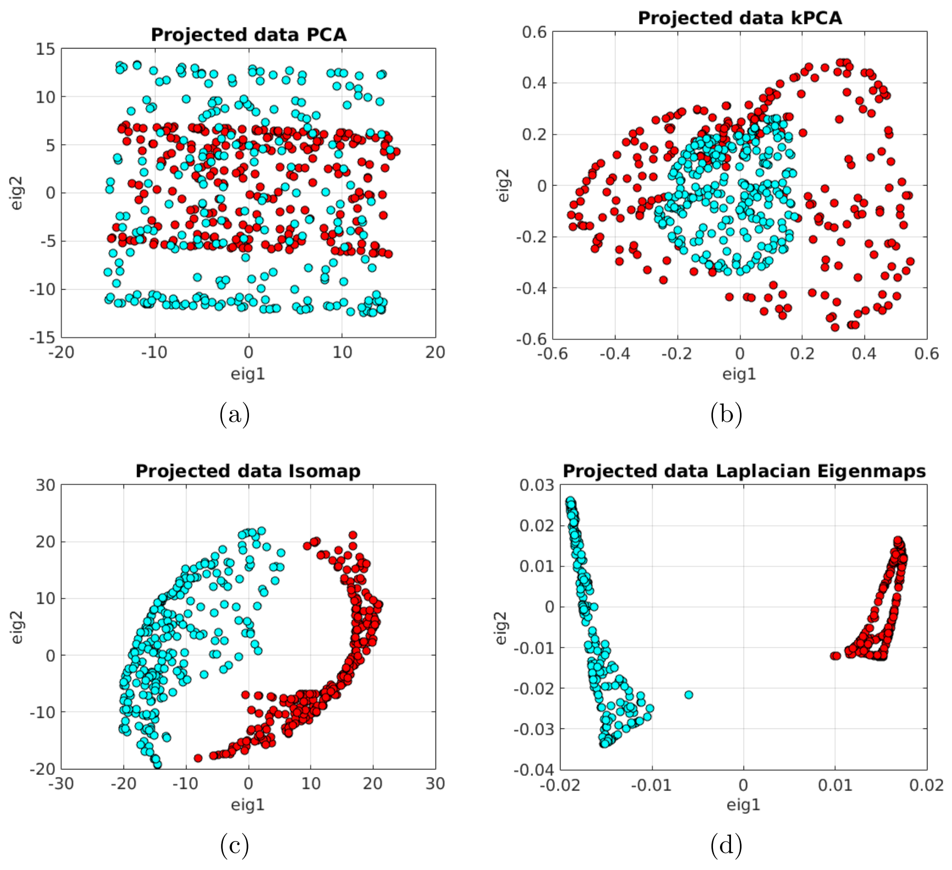


Figure 4: (a) Projected Dataset with PCA (b) Projected Dataset with kernel PCA (c) Projected Dataset with Isomap (d) Projected Dataset with Laplacian Eigenmaps

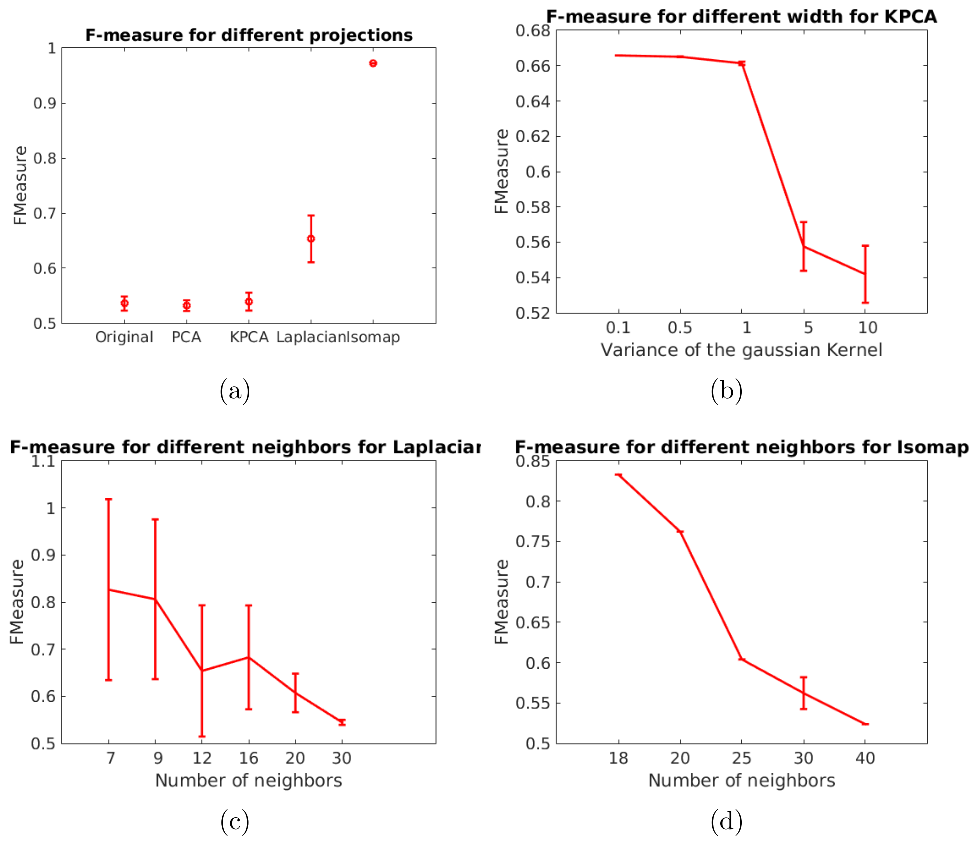


Figure 5: (a) F-measure for diferent projection techniques (b) F-measure for different width of the gaussian kernel (KPCA) (c) F-measure for different neighborhood (Laplacian Eigenmaps) (d) F-measure for different neighborhood (Isomap)

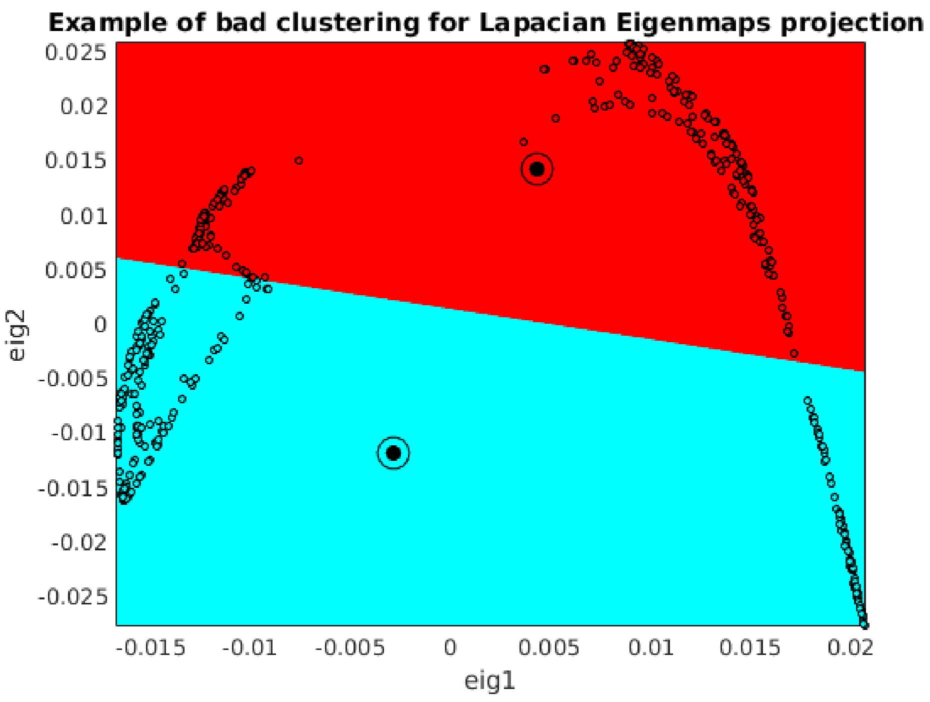


Figure 6: Example of a clustering not representative of the data

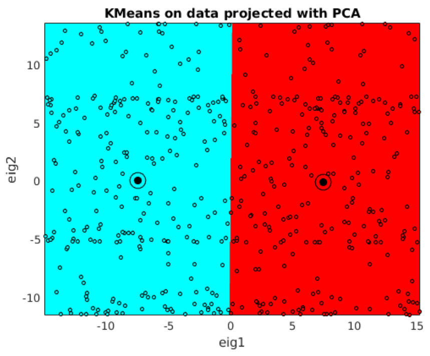
It can be very good or very bad depending on the initialization.
See Figure 6

The projection with Isomap leads to a more dense (scaled) distribution of the data and even if the clusters are not visually well separated, k-means performs better on it. The value of F-measure is almost constant.

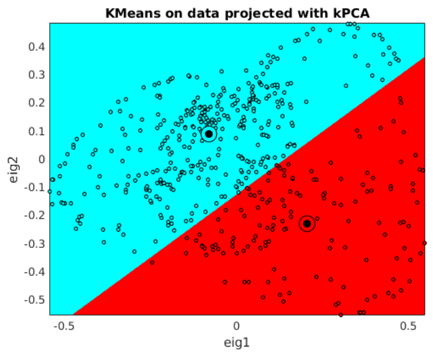
Is it possible to avoid the effect of selecting randomly the initial centers for this particuar case ? Having a 2D representation of the data with the two clusters can enable to select the two first centers by for example having one with positive X and one with negative X (and choosing the mean of Y coordinate over the data for both centers). You can do this by changing the *Start* parameter of the kmeans function.

4.3 Datasets Clusters

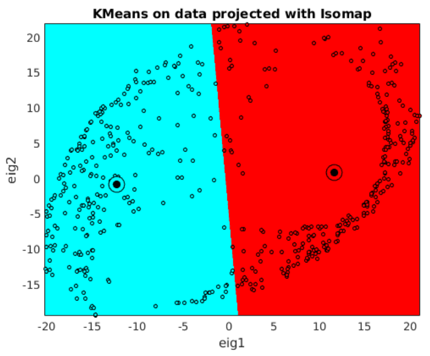
You will be trying to compare the four projection techniques used before (PCA ,kPCA, Laplacian Eigenmaps, Isomap) on the three real datsets : (1) *Breast-cancer-Wisconsin*, (2) *House-votes*, (3) *Digits*. As before, first project the data, keep a lower dimension and then apply k-means to try finding clusters. Use the value found in the first part of the practical for K or try different to see the influence of finding the good K number. Don't forget for the comparison to change not only the hyperparameters of the projection techniques but also the dimension and components after projecting.



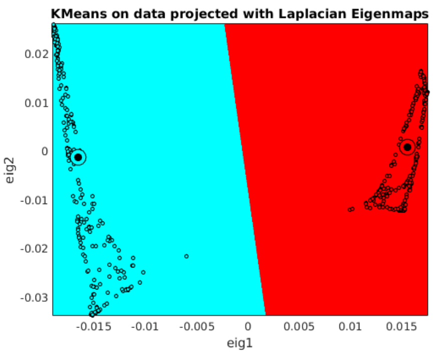
(a)



(b)



(c)



(d)

Figure 7: Examples of KMeans applied to each of the projected data