Advanced Machine Learning -Practical 2- Projection and Clustering

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

Spring Semester 2016

1 Introduction

The common difficulty with clustering problems is that you do not usually know before hand the number of clusters, K, present. One approach would be to do an exhaustive search over the number of possible clusters and retrain the one which performed the best. This approach is time consuming. An alternative would be to use a dimensionality reduction technique such as kernel-PCA (kPCA) to get an intuition regarding the number of clusters present in the data. In a first part of the tutorial, we will look into the following :

- Evaluate kPCA as means of deducing the number of clusters present in a dataset.
- Compare kPCA solution, in terms of the found clusters, with AIC and BIC with k-means clustering.
- Compare kernel-kmeans against k-means.

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 hidden in the manifold.

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 Projection Techniques to determine the number of clusters

3.1 Using kernel methods to determine the number of clusters

We will study how to kPCA can be used to help to determine the number of clusters present in a dataset. All kernel methods are based on the computation of the so called Gram matrix $K \in (N \times N)$,where N is the number of samples in your dataset. So be careful, if you use a kernel method on large dataset, it might take a very long time to compute this matrix. The way you can use matrix K to determine the number of clusters is through inspecting its eigenvalues.

Gram eigenvalues The Gram matrix is symmetric positive semi-definite (PSD), and as you have seen in class, it can be decomposed into its eigenvalues and eigenvectors,

$$K = V \Lambda V^{\mathrm{T}} \tag{1}$$

where $V \in (N \times N)$ are the eigenvectors α and $\Lambda \in (N \times N)$ is diagonal matrix containing the eigenvalues. In regular PCA the eigenvalues conveyed how much of the variance in the original signal is preserved in the projected space. Usually one would keep as many eigenvectors as necessary to explain at least 90% of the variance of the original data. Hopefully for you this will result in a significant dimensionality reduction. What off the eigenvalues of the Gram matrix ? They have exactly the same interpretation as standard eigenvalues (Appendix 5.1).

3.2 Clustering questions

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

Circles	:	2D data set of non linearly separable clusters.
High-dimensional clusters	:	10D synthetically generated data.
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 two techniques of finding the number of clusters K.

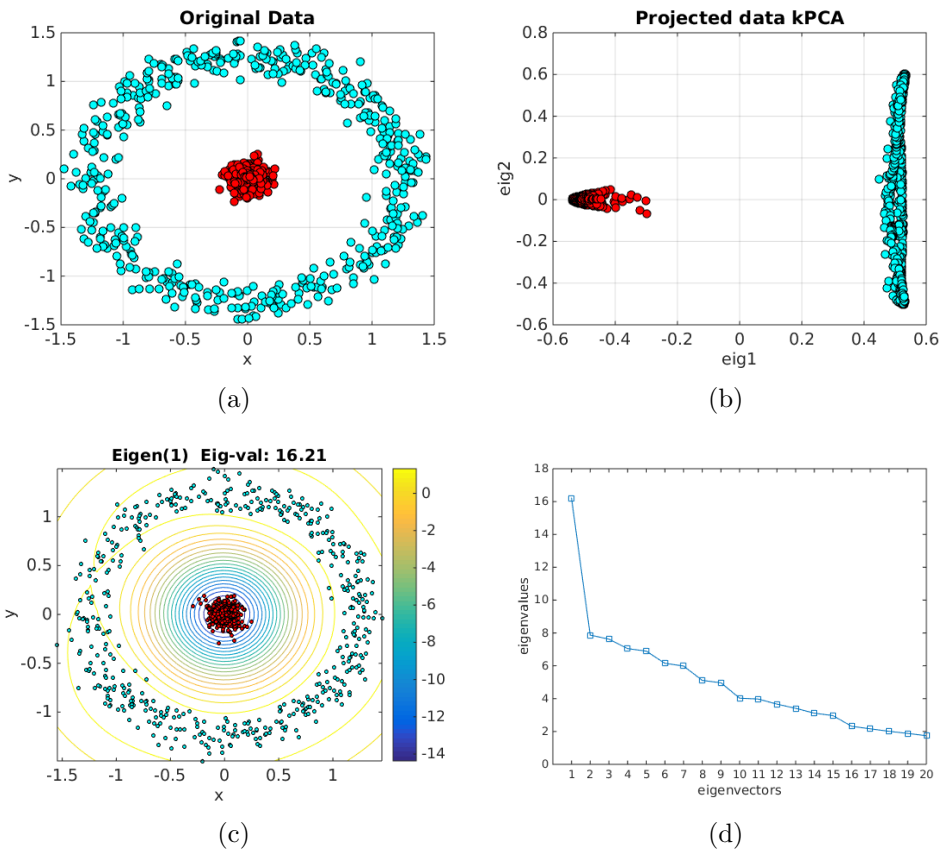


Figure 1: (a) Original circle dataset. (b) Projected data set with a Gaussian kernel function with $\sigma^2 = 0.4$. (c) Isolines of the first eigenvector draw onto the original data. (d) Eigenvalues as a function of eigenvectors.

- kPCA : the eigenvalues of kPCA can help determining the number of clusters present in your dataset.
- AIC & BIC: These information criterion are typically used to determine the number of clusters.

3.2.1 Circle clusters

Open **TP2_kPCA_circle.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 spheres depending on the dimension you choose for the original data Figure 1(a). Then you run kPCA with a chosen kernel, hyper-parameters and number of eigenvectors to retain. The result of the projection is illustrated in Figure 1(b). You can then inspect the eigenvalues and eigenvectors 1 (c)-(d). If kPCA manages to find the appropriate number of clusters, there should then be exactly one eigenvector per cluster. To see if you are setting the hyperparameters correctly you can look at the isoline plots of different eigenvectors to see if the clusters are getting encoded by an eigenvector.

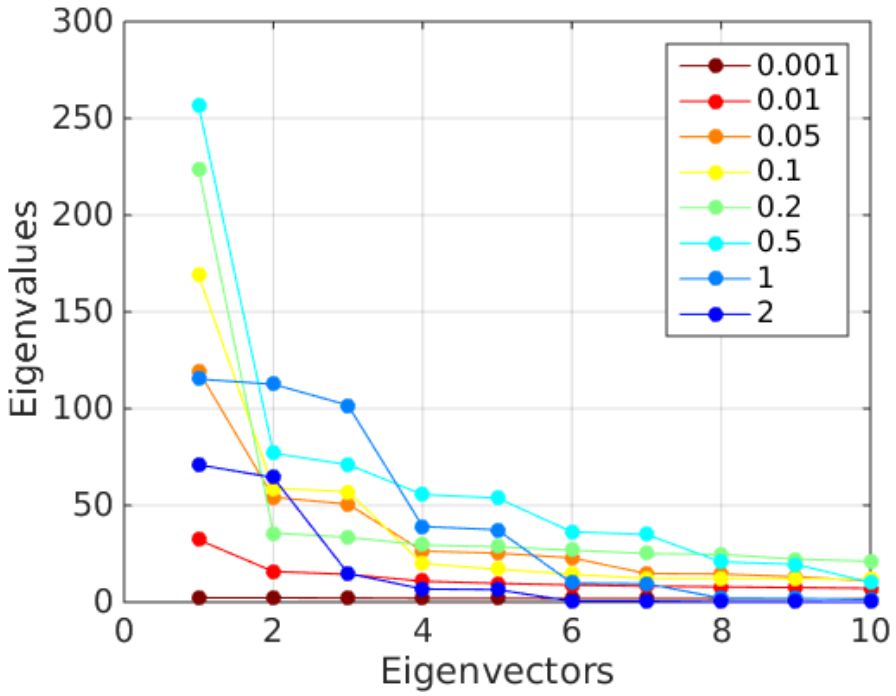


Figure 2: We can see that if the variance is too small or too large there is no dominant set of eigenvectors. In this case all of the eigenvectors have an equal amount of variance associated to them.

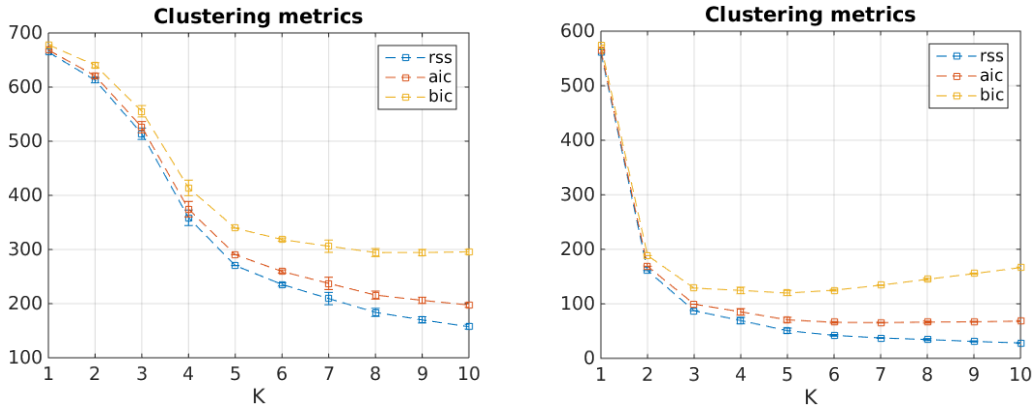
Q: How many clusters did you find with kPCA ? The result of kPCA will depend strongly on the choice of the kernel and its hyperparameters. For simplicity we only used the Gaussian kernel function. Now the problem comes to selecting the appropriate variance of the Gaussian kernel function. We did a grid search over the variance of the Gaussian function, see Figure 2. From this plot and knowing the original scale of the data we can identify the suitable σ^2 range as $[0.05 - 0.5]$. In this range it is quite clear that a dip always occurs after the second eigenvector. From this information we can gather that there is not much more than two clusters present.

Q: How many clusters did you find with AIC and BIC with k-means ? We ran AIC and BIC one time on the original data and a second time on the projected data. See Figure 3 for the results. On the original dataset the optimal number of clusters is around 6-7, but on the kPCA projected data it is around 3-4. This should give you a validation of what what we predicted by using kPCA to determine the number of clusters.

3.2.2 High dimensional clusters

You will how repeat the same steps and try to determine the optimal number of clusters for the synthetic high-dimensional dataset. Open the matlab script: **TP2_kPCA_highD.m**, follow the instructions and answer the following two questions:

Q: How many clusters did you find with kPCA ?



(a) (b)

Figure 3: Results of AIC and BIC for the circle data set. (a) Results for the **original** data. (b) Results for the projected data.

Q: How many clusters did you find with AIC and BIC with k-means ? The solutions is in Figure 4.

3.2.3 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 kPCA ?

How many clusters did you find with AIC and BIC with k-means ?

4 Projection techniques for clustering

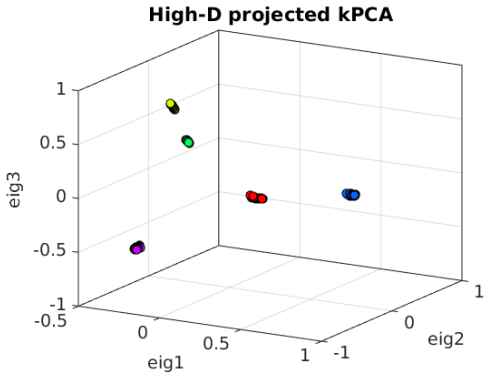
4.1 Using projection techniques to cluster data

We will study how projection techniques can be used to have a simple representation of the data for which it is easy to obtain 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.

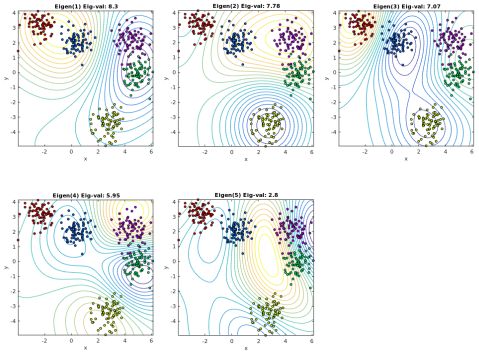
4.2 Tasks

Your task will be to find a good 2D projection technique for the swiss roll example in Figure 5 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 (kernel) k-means algorithm to try clustering the data.

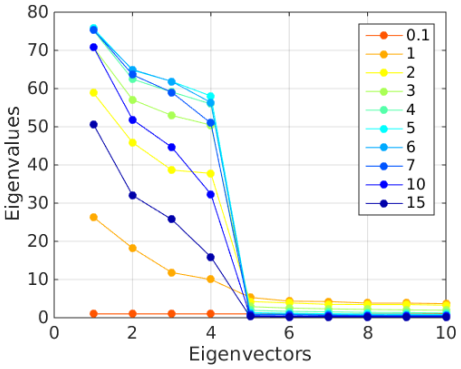
The projection techniques you will use are :



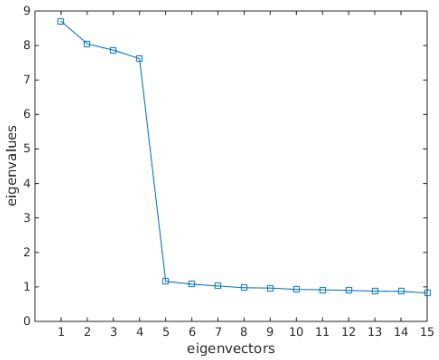
(a)



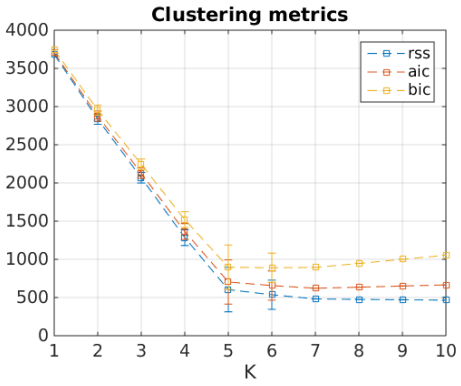
(b)



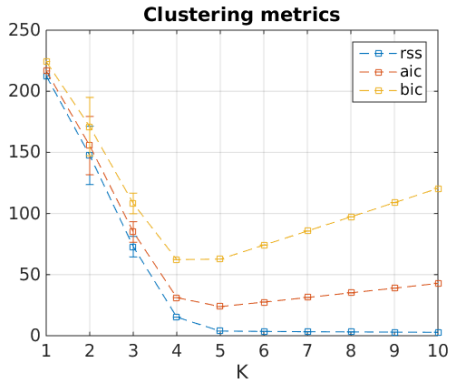
(c)



(d)



(e)



(f)

Figure 4: **Solution** (a) Projected data. (b) Isolines, can see how the eigenvectors cluster the data into groups. (c) search for the correct parameters for the Gaussian kernel function, all are in agreement. (d) same as (c) but for one case. (e) AIC and BIC on the original 10 Dimensional dataset. (f) AIC and BIC on the projected dataset.

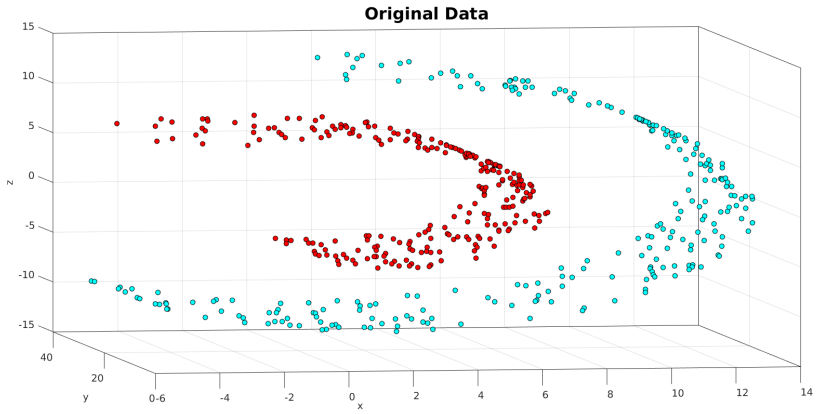


Figure 5: 3D view of the original dataset with two clusters

- PCA
- kPCA
- Isomap
- Laplacian Eigenmaps

4.2.1 Projection

Open **TP2_SwissRollClusters.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 5 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 6. 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 in 2D space ? See Figure 6

PCA and Kernel PCA doesn't work to get have a 2D 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

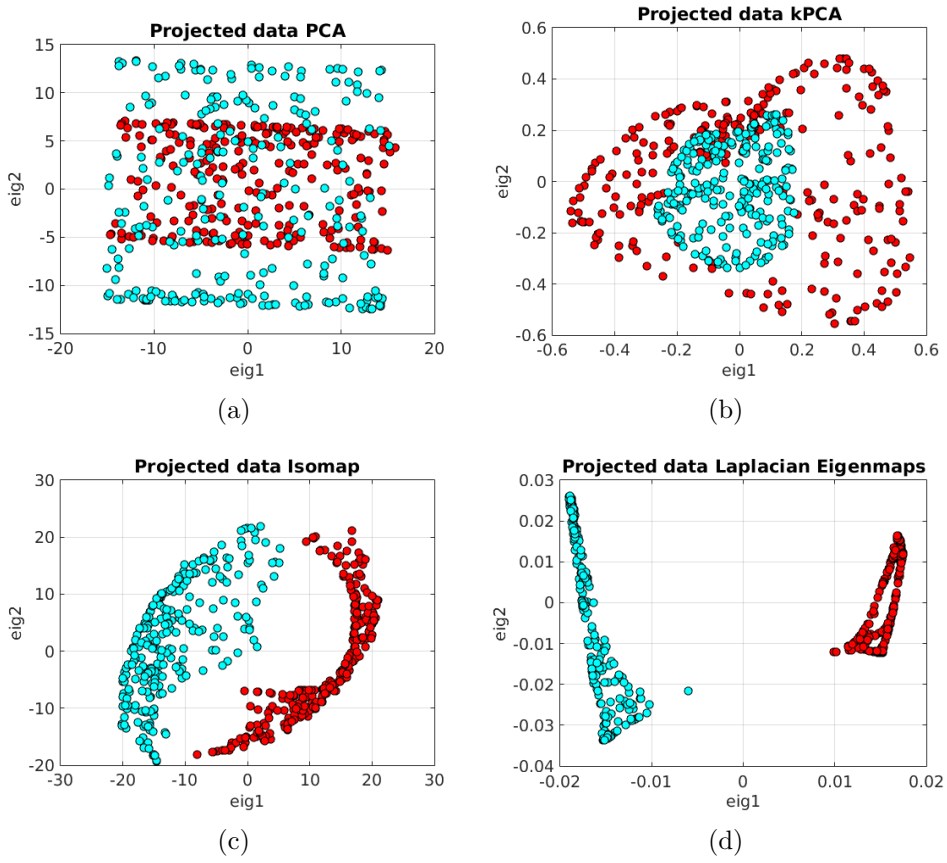


Figure 6: (a) Projected Dataset with PCA (b) Projected Dataset with kernel PCA (c) Projected Dataset with Isomap (d) Projected Dataset with Laplacian Eigenmaps

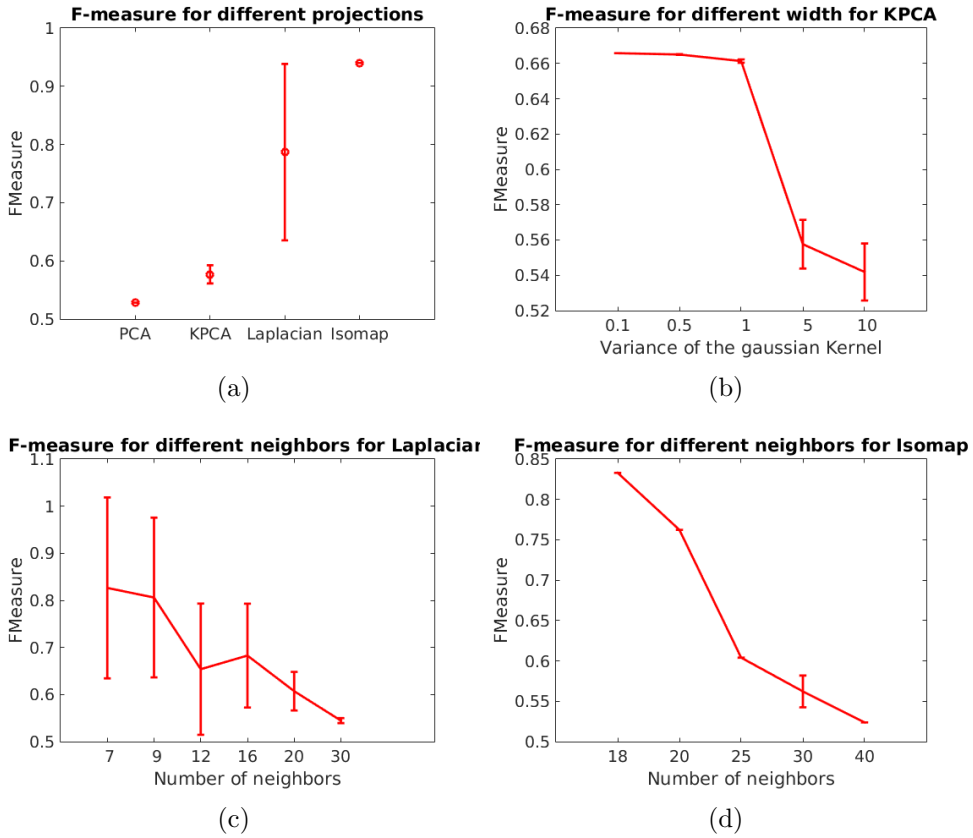


Figure 7: (a) F-measure for diferent projection techniques (b) F-measure for different width of the gaussian kernel (KPCA) (c) F-measure for different neighborhood (Laplacian Eigenmaps) (d) F-measure for different neighborhood (Isomap)

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 7

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 precision is not very high in mean. It can be very good or very bad depending on the initialization.
See Figure 8

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

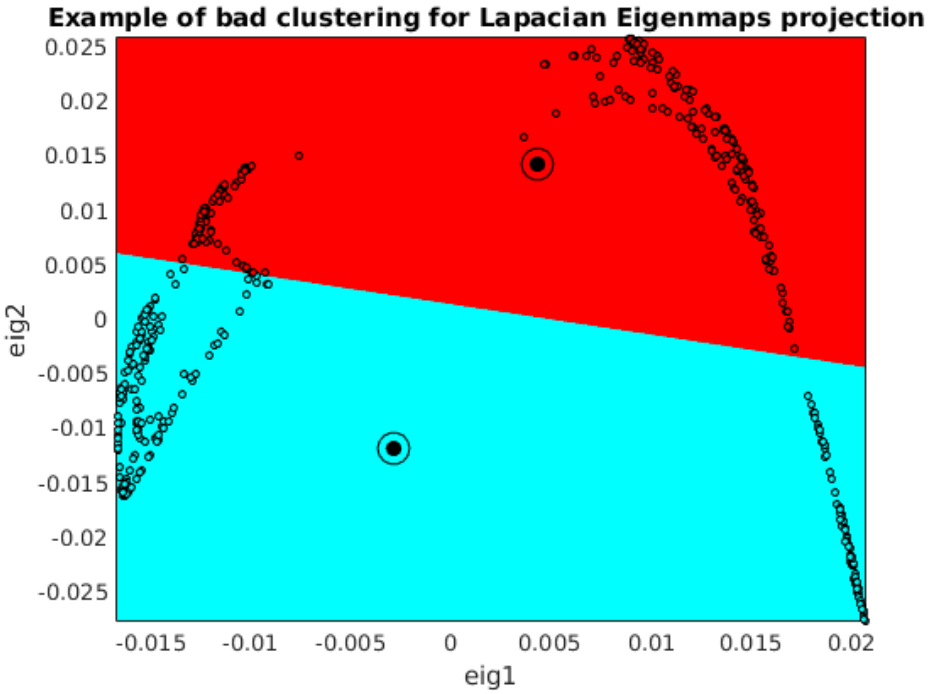


Figure 8: Example of a clustering not representative of the data

Is it possible to avoid the effect of selecting randomly the initial centers ? 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.

5 Appendix

5.1 kPCA and PCA properties

Properties of (Kernel) PCA. If we use a kernel that satisfies the conditions given in section 3, we know that we are in fact doing a standard PCA in F. Consequently, all mathematical and statistical properties of PCA (see, e.g., Jolliffe, 1986; Diamantaras & Kung, 1996) carry over to kernel-based PCA, with the modifications that they become statements concerning F rather than $\mathbf{R}^N$

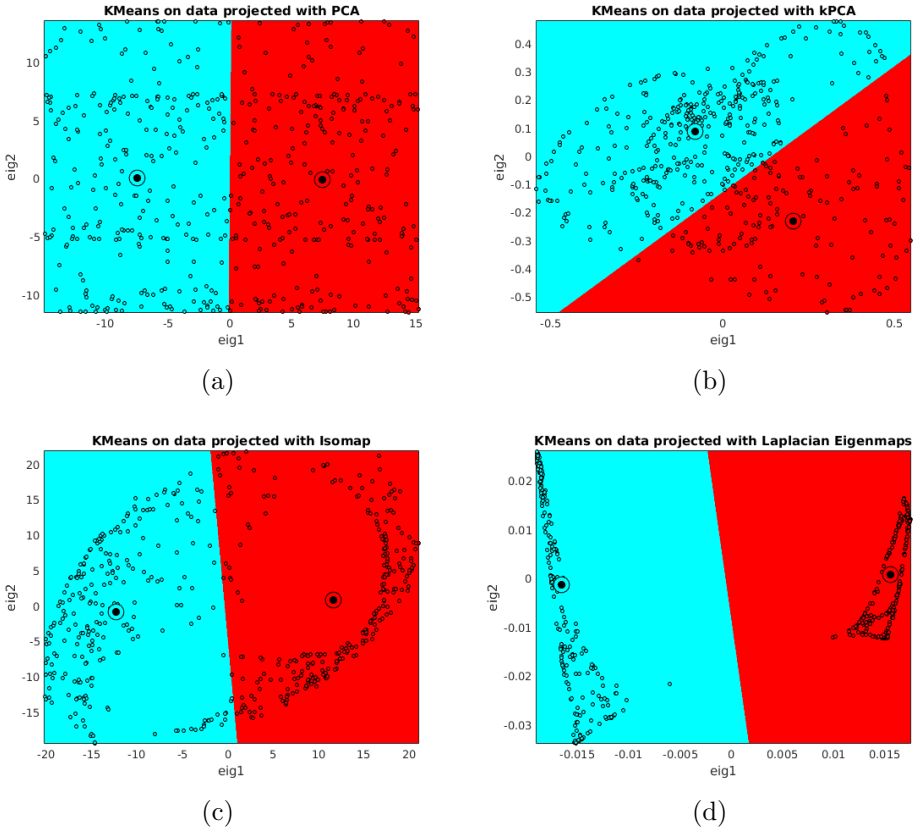


Figure 9: Examples of KMeans applied to each of the projected data