

Statistical data analysis

Spectral Clustering

Introduction

The aim of this tutorial is to get familiar with spectral clustering. You will use available building blocks and implement algorithm of spectral clustering. You will apply this algorithm on input data and compare the result with known annotation and with the result of classic k-means algorithm. You will perform the comparison for different parametrisations of input data and algorithm of spectral clustering.

1 Input data

In this tutorial, we will work with two synthetic datasets online available at <https://cs.joensuu.fi/sipu/datasets/>. First of all we start with a Spiral dataset (`spiral.txt`) containing three well-separated spirals that is depicted in figure 1. Second one, and more "realistic", is a Jain's toy problem (`jain.txt`) with two clusters. For loading of these datasets, you can use already prepared function `LoadDataset` in `spectral_clustering.R` file.

2 k-means Algorithm

Here, we are dealing with a clustering task with non-compact clusters. Algorithm k-means will probably generate clusters different from the golden standard (application of R function *kmeans* directly on input data - see figure 2). Since k-means algorithm is also the last step of spectral clustering, one of our aims is to find out how transformation performed in spectral clustering before application of k-means algorithm influences its output.

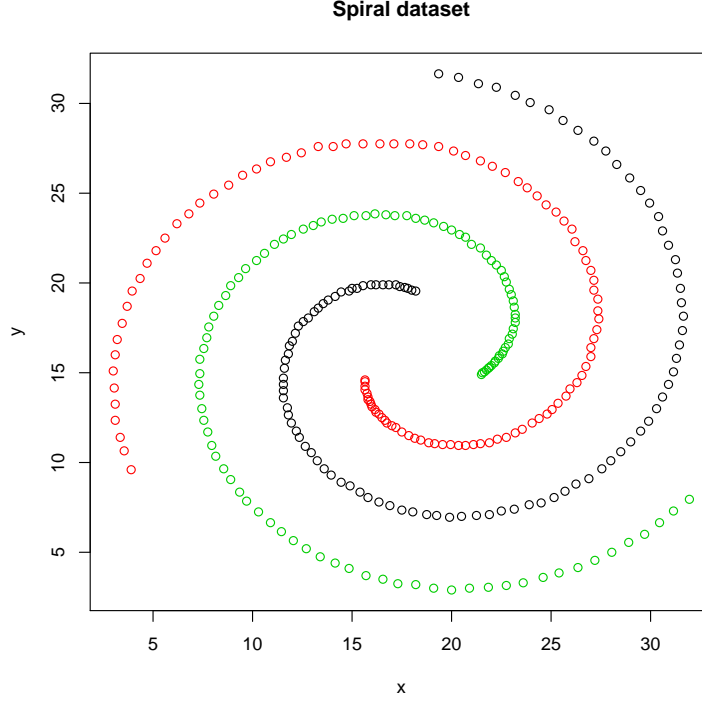


Figure 1: Spiral dataset with original labels.

3 Spectral Clustering

Spectral clustering can be divided into few building blocks. The following list describes these functional blocks.

- **Computation of similarity matrix \mathcal{S} :** `CalcSimMatrix` function calculates Euclidean distances between pairs of points and afterwards applies Gaussian kernel, modification is not required, but it is good to verify the importance of parameter σ (a higher value means greater similarity between more distant points, i.e. similarity is less local).
- **Construction of similarity graph:** there are three popular different similarity graphs, i.e fully connected graph, ϵ -neighborhood graph (`BuildEpsilonGraph` function), and k -nearest neighbor graphs. Fully connected graph, as a most trivial variant, keeps \mathcal{S} without changes; ϵ -neighborhood graph connect all points whose pairwise distance are smaller then ϵ and k -nearest neighbor graph connect vertex v_i with vertex v_j if v_j is among the k -nearest neighbors of v_i . However, this leads

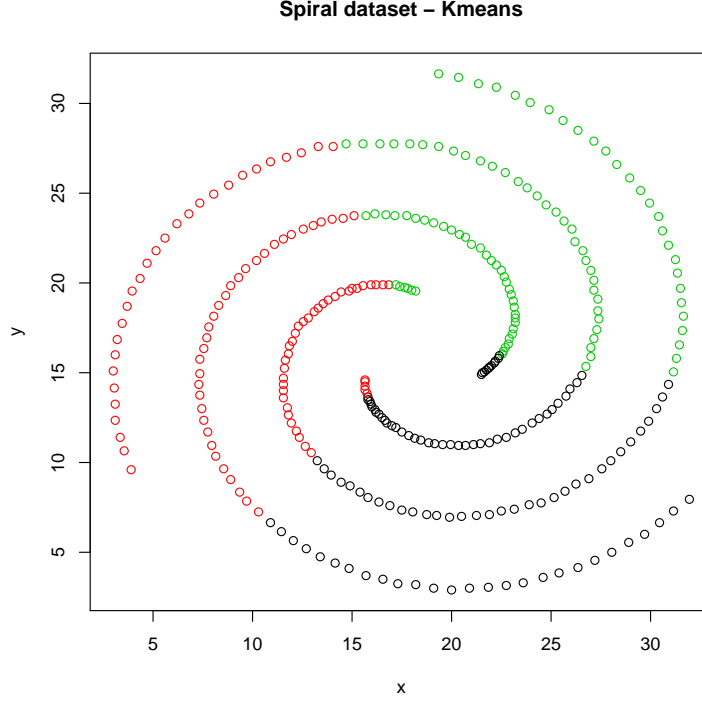


Figure 2: Kmeans clustering with $K = 3$.

to non-symmetric relationships and the similarity graph is **directed** (`BuildDirectedKNNGraph` function). To making similarity graph undirected (`BuildUndirectedKNNGraph` function), we can apply two approaches. First variant connect v_i and v_j if v_i is among the k -nearest neighbors of v_j or if v_j is among the k -nearest neighbors of v_i . Second variant connect vertices v_i and v_j if both v_i is among the k -nearest neighbors of v_j and v_j is among the k -nearest neighbors of v_i . In the second case, the graph is called the **mutual k -nearest neighbor graph** (see [1]).

- **Derivation of Laplace matrices \mathcal{L} with subsequent projection to the space of their k smallest eigenvectors:** `CalcLaplacian` function performs this step for non-normalized Laplacian (see [1]).
- **Application of k-means algorithm on the output of the previous function:** a straightforward application of R function `kmeans`.

4 Step by Step

You should go through the following steps:

1. read `spiral` and `jain` datasets,
2. perform clustering using k-means algorithm, evaluate its success rate visually using function `plot` and numerically using function `Purity`,
3. create the basic variant of the algorithm of spectral clustering from the existing functions,
4. perform clustering using spectral clustering, check connectivity graph by function `PlotConnectedGraph`, check results numerically using function `Purity` and visually by R function `plot`,
5. repeat step 4 with different variants of the spectral clustering algorithm (different variants of similarity graph)
6. summarize your experience from experiments (what option is important, what option does not have influence, etc.).

5 Evaluation

- `BuildEpsilonGraph` function [0.5p],
- `BuildDirectedKNNGraph` function [1p],
- mutual/unmutual version of `BuildUndirectedKNNGraph` function [1p],
- `CalcLaplacian` function [0.5p],
- documentation [1p].

6 Submission Form

Submit task as ZIP file containing all files necessary to run your code (try to test via Rscript `spectral_clustering_v1.2.R`) and write a short report (2-3 pages). The report should contain a definition of the task, interesting implementation details, results of two datasets (`spiral` and `jain`) and your answers for the task 4.6.

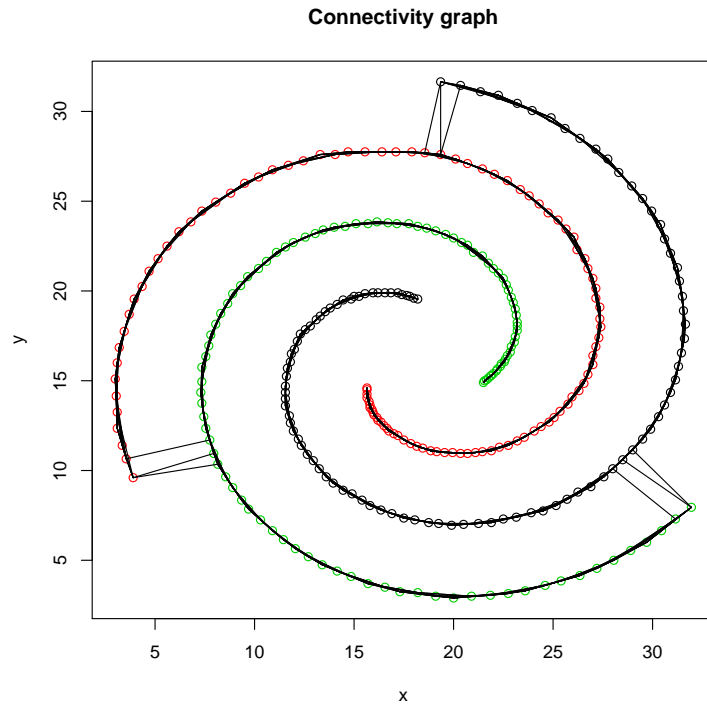


Figure 3: Almost ideal connectivity graph for Spiral dataset.

References

- [1] Luxburg, Ulrike: *A tutorial on spectral clustering*, Statistics and Computing, 17/4, pp. 395–416, 2007.