Aalto University
School of Science
Degree Programme in Mathematics and Systems Analysis

Nourhan Shafik

On Clustering of Non-smooth Functional Observations.

Master's Thesis
Espoo, November 19, 2018

Supervisor: Assistant Professor Pauliina Ilmonen
Advisors: Adjunct Professor Lauri Viitasaari

Aalto University
School of Science
Degree Programme in Mathematics and Systems Analysis

ABSTRACT OF
MASTER'S THESIS

Author:	Nourhan Shafik		
Title:	On Clustering of Non-smooth Functional Observations.		
Date:	November 19, 2018	Pages:	v + 56
Major:	Systems and Operations Research	Code:	F3008
Supervisor:	Assistant Professor Pauliina Ilmonen		
Advisors:	Adjunct Professor Lauri Viitasaari		

Nowadays, we are able to store large amounts of data. This has led to development of methods for analyzing very high dimensional observations. When we consider observations that are not only high dimensional, but infinite dimensional, we are in the setting of Functional data analysis.

In functional data analysis, the methods are often based on the assumption that the observations are smooth. However, in some applications, the smoothness assumption is not reasonable. For example, the nature of financial data is often non-smooth.

In this thesis, we consider non-smooth functional observations. We introduce a novel method for clustering non-smooth functional data.

First, we review basic concepts related to functional data analysis and clustering. After that, we present our new method. The method relies on Hölder continuity assumption. We map each functional observation to an index that measures the roughness of the observation. After that, we apply k-means clustering to the obtained indices. We consider theoretical properties of the method under the assumption of fractional Brownian motions and we present several simulated examples to assess its performance in practice. Based on these simulated examples, the method works extremely well in clustering fractional Brownian motions with different Hurst indices.

Keywords:	Functional data, clustering, k-means clustering, roughness, Hölder continuity, Hurst index, fractional Brownian motion
Language:	English

Acknowledgements

I would like to thank my whole family, especially, my mother who always gives me great support, my father, my sister, my husband, my brother, my little daughter who are the main sources for encouraging and motivating me, and my grandmother who helps me to be the one who is here. I would like to thank my friends, especially, Heba. I would like to thank my supervisors Pauliina Ilmonen and Lauri Viitasaari for their guidance, assistance, encouragement, and optimism. Thank you!

Espoo, November 19, 2018

Nourhan Shafik

Contents

Chapter 1

Introduction

Clustering aims to search for patterns and groups in the data. For example, clustering can be used for initial analysis of the data. Many clustering methods for different kinds of data have been developed over the last few decades. Hierarchical clustering and K-means algorithm are considered to be the oldest methods. They both are based on calculating geometrical parameters such as distances, while more modern methods are based on the concept such as the probability density function in model-based Gaussian clustering [8].

The advances in technology allow us to store massive amount of data accurately and to take into account other variables such as time variable as well [19], and this phenomenon is visible in many applications such as medicine, climatology, chemistry, and finance. This usually means that we have to consider very high dimensional data. Moreover, in several applications, the used data are not even finite dimensional anymore. This has led to the concept of functional data, where the data is assumed to be observations from some infinite dimensional space. In practice, usually the observations are assumed to be continuous functions. A naive approach is to discretize the data functions. However, this would result in large amount of data, and often it is more feasible to consider the data as functions directly. For example, in speech recognition applications, it is useful to consider the data in function form.

Similarly as in the case of finite dimensional data, clustering of functional data is performed to search for groups of observations having similar pattern. However, in the case of functional observations, the definition of similarity of curves is not obvious. For example, one can consider similarity in location, similarity in shape, or similarity in both. Furthermore, while many of the clustering methods can be applied easily for finite dimensional data set,

extension to functional data is not straightforward. For example, definition of a probability density for this kind of data is absent. Similarly, in infinite dimensional spaces, the choice of the distance metric has a huge effect. Many approaches to cluster functional observations have been presented in the literature. One approach is to project the infinite dimensional data into a smaller finite dimensional space, and perform clustering on the smaller space. Other commonly used approaches are nonparametric methods which aim to define some distances between the curves, and then to perform some typical clustering algorithm, such as k-means or hierarchical clustering, for these distances. Also, there are other methods such as model-based techniques including Gaussian mixture models.

Many of the functional data methods are assuming smoothness of the observations. However, this is not a realistic assumption in some applications. For example, it is known that typically stock indices in finance are not differentiable which violates usual smoothness assumptions. In this thesis, we address this problem by introducing a new metric that measures the non-smoothness of the functional observations. The performance of this new metric is studied by applying it to the clustering of non-smooth functional observations.

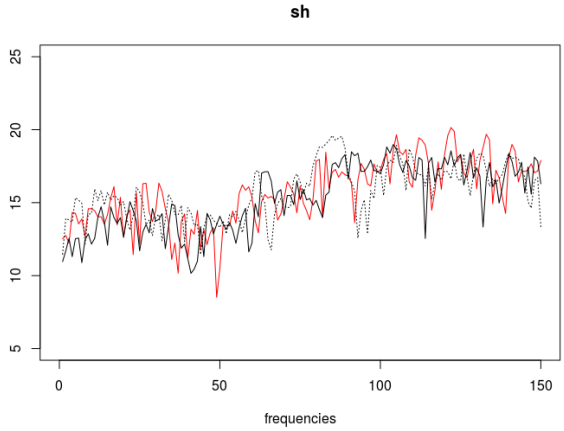
The thesis is organized as follows. In Chapter 2, we present the basic concepts of functional data. In particular, we explain how to deal with functional observations and how to recover functions from discrete observations. In Chapter 3, we discuss commonly used clustering methods. Especially, we introduce k-means clustering algorithm which is applied in this thesis. In Chapter 4, we review some existing clustering methods for functional data which are based on the assumption of smoothness. Specifically, we introduce basic ideas of two-stage methods, nonparametric methods, and model-based methods. Chapter 5 is dedicated to our actual results. We define our new metric which is based on Hölder continuity, and apply it to clustering of simulated functional data.

Chapter 2

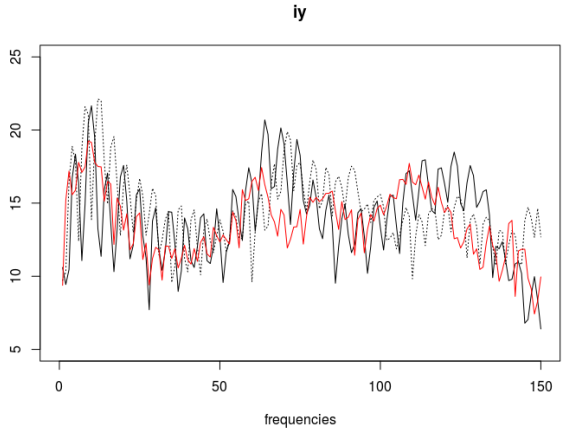
Functional Data Analysis

In this chapter we recall some basic facts and ideas of functional data analysis. For more details, We refer to [28] and [14] which are our main references.

Dealing with very high dimensional data can lead to severe problems such as lack of computational power. However, treating such data as functional observations can be beneficial even if the data is discrete in nature. Indeed, consider the following example from speech recognition which reveals how functional data analysis, FDA for short, plays a role. The used data is taken from [4] and it contains five different classes representing five phonemes:"sh","iy","dcl","aa", and "ao". Each of the five classes has 400 members and each member represents the log-periodogram corresponding to different frequencies (for more details, see [4]). Figure 2.1 represents three members from the first two classes.



(a) "sh" phoneme.



(b) "iy" phoneme.

Figure 2.1: Examples of phoneme data used in speech recognition.

The idea behind the functional data analysis is to look at the data from a new perspective. In other words, it aims to see the internal pattern of the data instead of seeing just a sequence of numbers. For example, the idea is to treat observations shown in Figure 2.1 as functions. In practice, one can not record the data continuously. Instead, one only observe the data discretely. In FDA it is assumed that the observations are pairs $(x(t_j), y_j)$ defined by $y_j = x(t_j) + \epsilon_j$, where x is a function and ϵ represents the noise such as measurement errors. This means that we are supposing that there exists a function that represents the discrete data. Functional data analysis aims to convert these discrete observations into functions or curves to illustrate di□erent characteristics of the data, and to explore more about the shape

and variation of the data. [28] In this case, one important issue is to take the inter-temporal dependence into account [32].

In FDA the objective is to fit a function to the observations. After that, FDA applies the same multivariate analysis techniques such as clustering and classification, but treating the observations as functions, not just a sequence of number. In order to fit a function to the observations, a standard assumption is the smoothness of the underlying function meaning that at least one derivative exists. Smoothness implies that every pair of neighboring points are suitably connected to each other. In addition, by applying first or second derivative, one can also take the shape of the function into account.

Usually it is assumed that the non-smoothness of the observed data could be caused by the noise as a result of the measurement procedure, and in order to transform the discrete observations into functions, it is usual to get rid of the noise term as much as possible. However, as mentioned in the introduction, it might be that the functions themselves are not smooth.

2.1 Commonly Used Basis Functions

In functional data analysis, the discrete data is transformed into (smooth) functions. This is done by defining some basis functions ϕ_k and then constructing a new function as a linear combination of these basis functions:

$$x(t) = \sum_{k=1}^{K} c_k \phi_k(t), \tag{2.1}$$

Where c_k is the corresponding coe□cient for each basis function ϕ_k. [28] Then the data is fitted by estimating the coe□cients c_k of the linear combination. These basis functions should be chosen to be orthogonal (in some suitable space) so that, as a result, any function can be defined as a linear combination of these basis functions. There are two frequently used basis functions: B-splines which are used for non-periodic functions, and Fourier series which are used for periodic functions. For example, The defined basis functions for Fourier are $1, sin(wt), cos(wt), sin(2wt), cost(2wt), ..., sin(kwt), cos(kwt), ...$ We will discuss the B-spline system in more details. However, there are some other beneficial basis systems such as wavelets, exponential and power bases,...etc. For more details about these systems, please see [28].

We begin by defining a spline function. For simplicity, we consider two dimensional space. First, the given interval $[t_L, t_U]$ is divided into p+1 smaller

intervals $[t_k, t_{k+1}], k = 1, 2, \ldots, p$, called sub-intervals, where $t_1 = t_L$ and $t_{p+1} = t_U$. A knot is a point (t_k, y_k) in a two dimensional space for some y_k. Next, we fix an order n for the polynomial, and take any polynomial p_k of order n on $[t_k, t_{k+1}]$ such that $p_k(t_k) = y_k$ and $p_k(t_{k+1}) = y_{k+1}$. In other words, p_k is simply a polynomial of order n on $[t_k, t_{k+1}]$ that hits knots (t_k, y_k) and (t_{k+1}, y_{k+1}). A Spline function is simply constructed by combining these polynomials. The order of the spline function is the number of coe□cients defined in a polynomial for an interval and it is $n+1$ where n is the degree of the polynomial. For example, if the order is four, this will give cubic splines, if the order is three, this will give quadratic splines, and if the order is two, this will give linear splines. We also note that the order should be at least two, since the polynomials should go through the knots.[28]

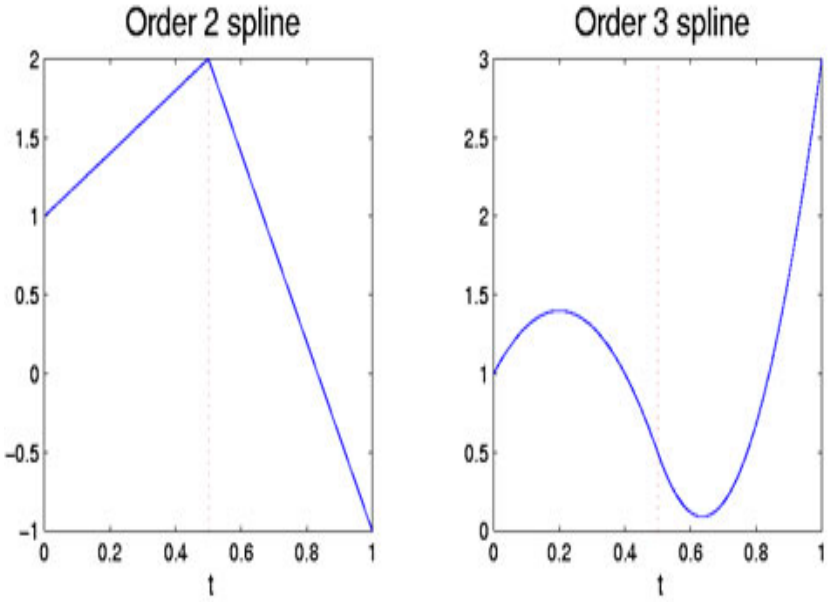


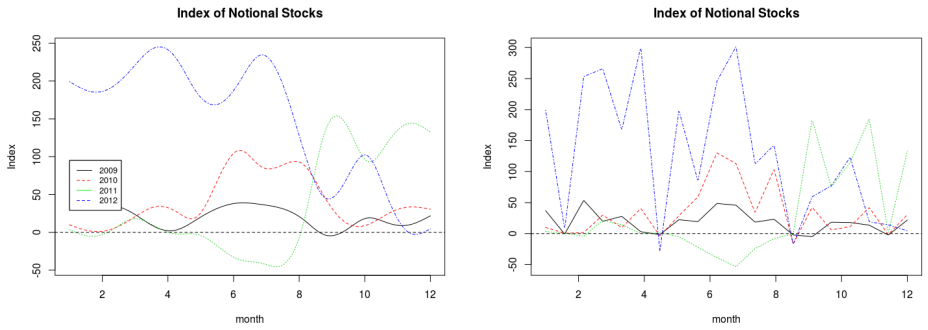
Figure 2.2: Spline functions with order=2 and 3.

In order to define a spline function, we need to determine the polynomial order and the knot points. [28] In practice, it is important to increase the number of knots in the parts where the function has extreme changes in order to have good approximation. Moreover, smoothness assumptions imply higher order splines. For example, the cubic spline is frequently used because this allows to have matched second order derivatives at the knots which creates smooth curves. Similarly, in order to have higher order matched derivatives, you need to increase the order.

The splines have the characteristics that if they are added or subtracted, they will still be spline functions. Thus, every linear combination of spline functions is still a spline function. Therefore, as the polynomials on sub-intervals can be recovered by using arbitrary polynomial basis, fitting a spline into the discrete data simply means fitting a polynomial of fixed order separately on each sub-interval with possible smoothness requirements. For more details about the B-spline system, please see [28].

2.2 Fitting The Data

It is customary to smooth the functional observations. The smoothing process aims to find the curve that represents the data well. In addition, the new curve should not be too fluctuated either. Therefore, we should compromise between having good fit and not producing too fluctuated curves. This can be done by ensuring that, at each point t, the value of the estimated curve mainly depends on the neighbored data around t. An example where the spline system is fitted to financial functional data is given in Figure 2.3. The used data is taken from [2].



(a) Spline order=4. (b) Spline order=2.

Figure 2.3: Index of notional stocks in Finland between 2009 and 2012.

There are two diverse approaches for smoothing: least squares methods and roughness penalty. [29]
For both approaches, the relation between the discrete observations and the recovered function is expressed as $y_j = x(t_j) + \epsilon_j$, where y_j are the used discrete data and $x(t)$ is calculated as

$$x(t) = \sum_{k=1}^{K} c_k \phi_k(t), \tag{2.2}$$

where c_k is the corresponding coe□cient for each basis function ϕ_k.
The idea is to minimize the error, measured in some sense, for fitting the data. For this, there are several approaches. One approach is to minimize the residual sum of squares:

$$\sum_{j=1}^{n}[y_j \quad x(t_j)]^2. \tag{2.3}$$

Another approach is to minimize the mean squared error function [28]:

Mean squared error=Bias^2+Sampling Variance.
The first one is called the least squares minimization and the second one is one example of the roughness penalty.

2.2.1 Fitting The Data Using Least Squares

The most common method for fitting the data is done by the least squares minimization. There are two types of least square minimization, weighted and unweighted least squares, although Weighted least squares is often used in practice. However, the unweighted least squares method is usually chosen if the residuals are independent and identically distributed with zero mean and constant variance.

In order to fit the data, a simple optimization problem should be solved to find values of the coe□cients c_k that minimize the error. The error can be expressed as

$$SMSSE(y|c) = \sum_{j=1}^{n}[y_j \quad \sum_{k}^{K} c_k\phi_k(t_j)]^2. \tag{2.4}$$

The previous equation can be expressed in a more simplified vector form:

$$SMSSE(y|c) = (y \quad \phi c)^T(y \quad \phi c) \tag{2.5}$$

To obtain the coe□cient vector c that minimizes the error, we di□erentiate with respect to c. This leads to

$$(2\phi\phi^T c) \quad (2\phi^T y) = 0. \tag{2.6}$$

This equation gives the optimal value for the coe□cient vector

$$c = (\phi^T\phi)\ ^1\phi^T y. \tag{2.7}$$

In practice, the assumption of independent and identically distributed with zero mean and constant variance is not always valid. In these cases, weighted least squares is used. In weighted least squares method, non constant variances are compensated by adding weights to the residuals. [28]

2.2.2 Fitting The Data Using Roughness Penalty

The roughness of a function is defined by integrating the square of the second derivative of the function:

$$PEN_2(x) = \int [D^2x(s)]^2 ds. \tag{2.8}$$

It is obvious that, for more rapidly changing functions, this term will have quite high values.

When the roughness notion is considered, then an important loss function, called penalized sum of squared errors, is used. As we mentioned before, there is clearly a trade o between having quite low errors and getting less fluctuated function. This trade o can be represented in the penalized residual sum of squares which is expressed as

$$PENSSE_\lambda(xjy) = [y \quad x(t)]^T W[y \quad x(t)]^2 + \lambda PEN_2(x). \tag{2.9}$$

The objective becomes now to find the function x, or in other words, to find the coecient vector c, that minimizes the previous expression. The smoothing parameter λ plays a crucial role in the expression. For high values of λ the penalty term is more involved, and non linear functions will lead to higher penalty. As a result, fluctuating curves can not be optimal solutions for this problem. By contrast, for small λ the penalty term plays less role in the whole expression, and consequently the curve can be more changing. [28]

Chapter 3

Clustering

Recently machine learning has become essential for many types of applications. These algorithms can be classified into six categories: supervised learning, unsupervised learning, semi-supervised learning, reinforcement learning, transduction and learning to learn. [7]

Clustering is an unsupervised learning algorithm which means that the data are not labeled. Therefore, the data is clustered without a previous knowledge of the labels. Clustering aims to find homogeneous groups of data. [7] Clustering is used in many applications such as pattern recognition and data segmentation.

There are two various types of clustering: hard clustering and soft clustering. In hard clustering, also called exclusive clustering, the points are belonging to only one cluster. In comparison, in soft clustering, also called fuzzy clustering, the points are associated to all clusters with diverse membership weights. Then, clusters are considered as fuzzy sets and the weight coe□cients are between 0 and 1 such that their sum corresponding to each point must be equal to 1. This leads to a new constraint requiring that the sum of all weights for each point is equal to 1.

There are two distinct types of clustering: hierarchical and partitional clustering. In partitional clustering, the data points are clustered into non-overlapping clusters, which means that each point is attached to only a single cluster. In contrast, hierarchical clustering permits that clusters can include subclusters, in other words, subclusters can be gathered to form some other larger clusters [30].

There exist two kinds of hierarchical clustering methods: agglomerative and divisive. In agglomerative hierarchical clustering, we begin by putting

each point in one cluster. After that, the closest clusters are merged, and then this is repeated until we have only one group. By using this approach, one obtains more information about the relationships among the clusters.

Several distances can be used to merge the groups in agglomerative hierarchical clustering. Single linkage, first introduced in 1957, is based on the minimum distance between two members in two di□erent classes. The minimum distance between groups A and B is

$$d_s(A, B) = min_{i2A,j2B} d(i, j), \tag{3.1}$$

where $d(i, j)$ is the distance between i^{th} member and j^{th} member. Complete linkage, first presented in 1960, based on the maximum distance between two members in two di□erent groups. This means that the distance is computed by

$$d_s(A, B) = max_{i2A,j2B} d(i, j). \tag{3.2}$$

Average linkage, first proposed in 1958, is based on computing the average distance of all the members in a class to all the members in the di□erent class. This distance is expressed as

$$d_A(A, B) = \frac{1}{jAjjBj} \sum_{i2A,j2B} d(i, j), \tag{3.3}$$

where jAj is computed as the number of members in class A. Finally, we mention a novel way introduced by Ward [33]. Ward calculated the total within-cluster sum of squares (SSE) and closed clusters were merged based on this measure. This measure is expressed as

$$SSE = \sum_{i=1}^{K} \sum_{j=1}^{n_i} (y_{ij} \quad \bar{y}_i)^T (y_{ij} \quad \bar{y}_i), \tag{3.4}$$

where y_{ij} is the j^{th} member of the i^{th} class and n_i is the number of members in class i. [15] [30]

3.1 K-means Clustering

K-means clustering is probably the best known partitional clustering method. In this section, we discuss k-means algorithm in details as it has been widely used with functional data. Moreover, it will also be applied for non-smooth functional observations, based on a novel specific distance, in Chapter 5.

K-means clustering algorithm was first introduced in 1967 by MacQueen [23], and it is considered as one of the easiest unsupervised learning algorithms. The algorithm clusters the data into k non-overlapping clusters for some predetermined number k. It is a prototype-based clustering, which is also called center-based clustering. This means that the points in the same cluster are closer to the prototype defined for this cluster than any prototype for other clusters. The prototype is usually defined as the centroid of the cluster, which is often calculated as the mean of all the data belonging to the cluster. [30]

In k-means, the number of centroids plays a crucial role and it is important to choose this number carefully. In other words, these cluster centroids must be defined in a way that facilitates the partitioning of the data, as well as minimizes the within cluster distances. After choosing the number k and placing the initial centroids, distances between each point in the dataset and the defined centroids are calculated. Then, based on these distances, each point is assigned to the cluster with the closest centroid. Subsequently, the new k centroids are calculated based on observation on a particular cluster. This is repeated until the centroids become unchanged. [30]

The algorithm outcome depends in the initial centroids and the chosen distances. The distance is defined by certain proximity measure and this measure can be calculated by various distance metrics such as Manhattan distance, Squared Euclidean, and cosine. The most used metric is the Euclidean distance due to its simplicity.

Let us begin by a simple example for univariate data. We consider a sequence of numbers $x = [1, 2, 4, 4, 15, 20, 13, 68, 77]$ and apply the k-means algorithm with $k = 3$ for this set.

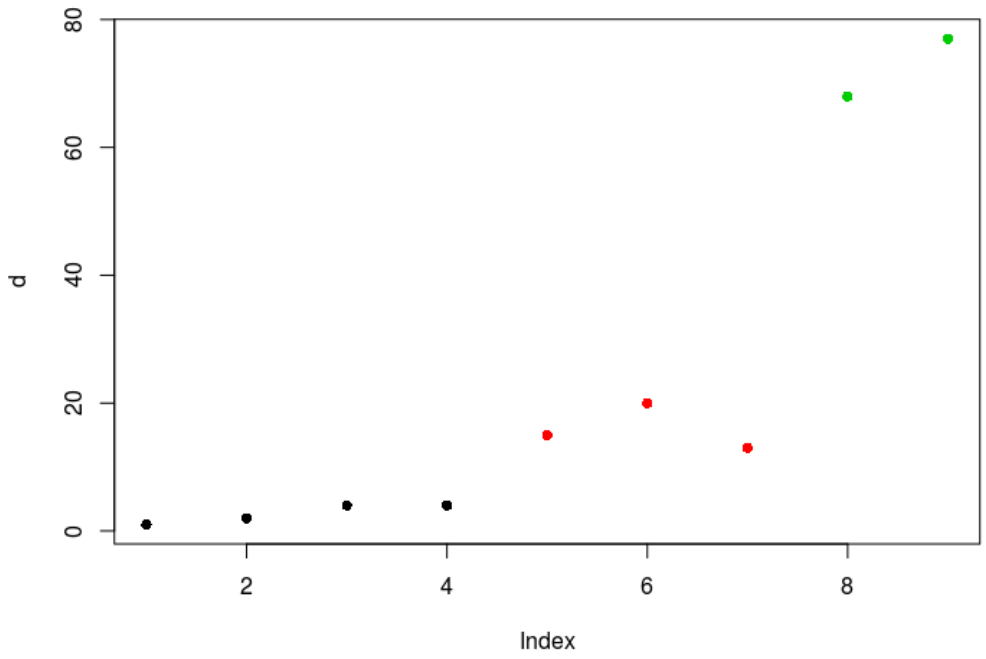


Figure 3.1: Result of k-means for univariate data.

As a result, we have three di□erent clusters which are {1, 2, 4, 4}, {15, 20, 13}, and {68, 77}. These are shown in Figure 3.1.

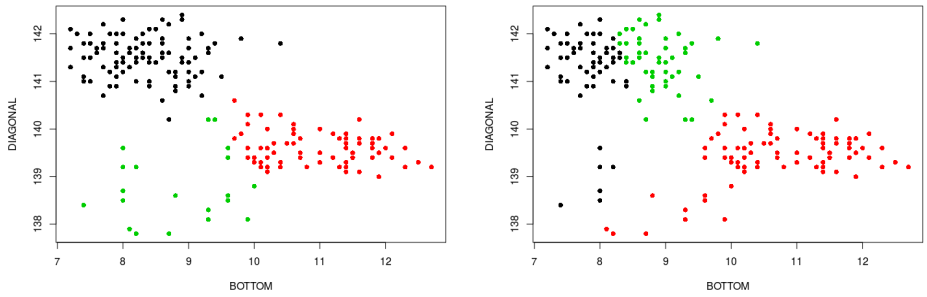
3.2 K-means Problems

It is possible that k-means algorithm do not produce optimal, in some sense, clusters although it usually converges. In this section, we explain three different problems related to k-means algorithm that can cause unwanted behaviour.

3.2.1 Initialization Problem of the centers

The outcome of k-means algorithm depends heavily on the initial centroids. Figure 3.2 illustrates two di□erent outcomes for certain dataset with di□erent initial centroids. Usually, it is essential to execute the algorithm several times as the first k centroids are often chosen randomly [7] [30]. However, executing the algorithm many times does not ensure reasonable outcome.

Consequently, other methods have been developed in order to address the initialization problem.



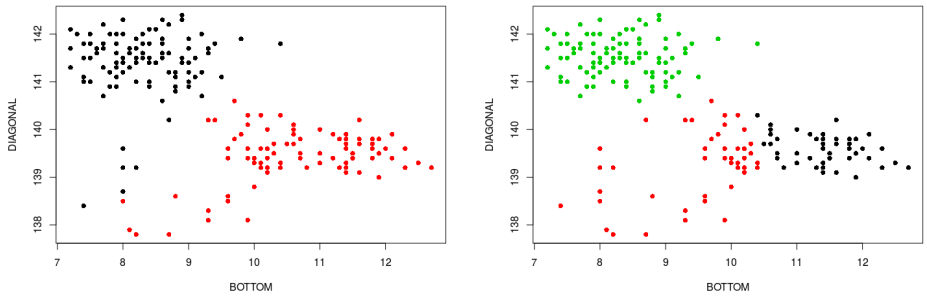
(a) Three clusters by k-means. (b) Three different clusters.

Figure 3.2: K-means initialization problem.

The reason behind is that k-means usually converges to a local minima. This means that it is critical to have a perfect initialization of the centers of the clusters. Many suggestions have been presented in the literature. Wardâs hierarchical method was further developed by Milligan [24], which aimed to have good scheme for the clusters. Another technique to solve this problem was suggested by Fisher [16]. Fisher suggested using hierarchical clustering as a first step to identify the initial centers and then applying k-means. However, this technique should be used merely for small dataset as hierarchical method has relatively high computational cost with large dataset.

3.2.2 Model Selection

Another problem is that the number of clusters should be chosen appropriately. This leads to a model selection problem where we have m models and we are required to choose the best one. An illustrative example is presented in Figure 3.3. Note that it is clear from Figure 3.3 that clustering is better when number of clusters is equal to two. Obviously, this is not true for all the datasets.



(a) Two clusters. (b) Three clusters.

Figure 3.3: K-means model selection problem.

Many criteria to solve the model selection problem have been proposed. For example, based on Bayes' theorem, the model to be selected is the one with the highest value of the posterior probability. If all the models have the same prior probabilities, then in order to choose the best model, we merely need to consider the likelihood. On the other hand, in order to prevent overfitting we should not select the model having the maximum likelihood. To avoid overfitting, one can introduce penalties to the maximum log likelihood (MLL). One commonly used criterion is Akaike information criterion (AIC)

$$AIC_i = MLL_i \quad d_i, \tag{3.5}$$

where d_i the dimension of model. Another criteria is called Bayes information criterion (BIC) defined by

$$BIC_i = MLL_i \quad \frac{1}{2} d_i \log n. \tag{3.6}$$

In order to choose the best model using these two criteria, one has to choose the model with the highest value of either AIC or BIC. On the other hand, Schwarz has proven that BIC performs properly under some conditions [1]. On the other hand, if the estimated model was not one of the familiar models, this can result in failure of selecting the right size of the clusters. Another criterion called Integrated Complete Likelihood (ICL) has been proposed to avoid this problem. This new method will result in the best clustering scheme for the data. The idea is to choose the model with has the largest value of the integrated likelihood given by the following equation:

$$(\hat{m}, \hat{K}) =_{m,K} f(x\boldsymbol{j}m, K), \tag{3.7}$$

where,

$$f(x|m,K) = \int_{\theta_{m,K}} f(x|m,K,\theta) \prod(\theta|m,K)d\theta \tag{3.8}$$

$$f(x|m,K,\theta) = \prod_{i=1}^{n} f(x_i|m,K,\theta) \tag{3.9}$$

In order to simplify the calculation, this expression can be approximated

$$\log f(x|m,K) = \log f(x|m,K,\hat{\theta}) \quad \frac{\nu_{m,K}}{2}\log n, \tag{3.10}$$

where $\hat{\theta}$ is the point that maximizes the maximum likelihood

$$\hat{\theta} =_{\theta} f(x|m,K,\theta). \tag{3.11}$$

$\nu_{m,K}$ is defined as the number of free parameters in the model m with K components. In order to reduce the approximation error in the expression (3.11), ICL considers an optimization problem

$$(\hat{m},\hat{K}) =_{m,K} f(x,z|m,K), \tag{3.12}$$

where

$$f(x,z|m,K) = \int_{\theta_{m,K}} f(x,z|m,K,\theta) \prod(\theta|m,K)d\theta \tag{3.13}$$

$$f(x,z|m,K,\theta) = \prod_{i=1}^{n} f(x_i,z_i|m,K,\theta) \tag{3.14}$$

$$f(x_i,z_i|m,K,\theta) = \prod_{k=1}^{K} p_k^{z_{ik}}[\phi(x_i|a_k)]^{z_{ik}}. \tag{3.15}$$

The approximated expression will be

$$\log f(x,z|m,K) = \log f(x,z|m,K,\hat{\theta}) \quad \frac{\nu_{m,K}}{2}\log n, \tag{3.16}$$

where $\hat{\theta}$ is the point that maximizes the maximum likelihood

$$\hat{\theta} \ =_{\theta} f(x,z|m,K,\theta). \tag{3.17}$$

The problem here is that the parameter z is not defined. However, for large datasets the undefined z can be expressed also as the maximum posterior. [10]

BIC, AIC and ICL are classical model selection techniques which are usually applied in model-based clustering. However, due to the importance

of this step, several other methods have been proposed to choose the number of clusters perfectly. For example, one proposed method that can compute the number of clusters immediately is the Bayesian model used for functional data hierarchical clustering.[19]

3.2.3 The K-means Non-probabilistic Structure

The last problem related to k-means algorithm we mention is that k-means performs well only if the data is really well-separated. The reason behind is that k-means uses simple distance calculations to define the clusters. One way to improve clustering performance is to use Gaussian mixture models. These models aim to find the best fitted multi-dimensional Gaussian probability distribution for the data.

The joint probability distribution function (pdf) for a multivariate normal random variable $x = (x_1, x_2, ..., x_D)^T$ is given by

$$p(x) = \frac{1}{(2\pi)^{D/2} j \sum j^{1/2}} exp[\quad \frac{1}{2}(x \quad \mu)^T, \sum \quad ^1 (x \quad \mu)] \tag{3.18}$$

where μ is the expectation vector and $\sum$ is the covariance matrix with determinant $j \sum j$. Now a certain scalar random variable x is characterized by the Gaussian-mixture distribution if the pdf of the random variable x is defined as

$$p(x) = \sum_{m=1}^{M} \frac{c_m}{(2\pi)^{1/2} \sigma_m} exp[\quad \frac{1}{2} \frac{(x \quad \mu_m)^2}{\sigma_m}] \tag{3.19}$$

for some number M and parameters σ_m, μ_m, and c_m for $m = 1, \ldots, M$. Similarly, a multivariate random vector has the Gaussian mixture distribution if it has a pdf [13]

$$p(x) = \sum_{m=1}^{M} \frac{c_m}{(2\pi)^{D/2} j \sum_m j^{1/2}} exp[\quad \frac{1}{2}(x \quad \mu_m) \sum_m \quad ^1 (x \quad \mu_m)]. \tag{3.20}$$

The set of parameters to be estimated from the data is the set $\theta = f c_m, \mu_m, \sum_m g$. As the distribution is known, a standard approach is to use maximum likelihood estimation. Specifically, the EM algorithm has been used in this context. The algorithm has two iterative steps: the E-step which calculates the expectation of the log-likelihood using the current estimate for the parameters, and the M-step which ensures that for each new iteration, the likelihood will increase. For more details, see [13].

Chapter 4

Clustering of Functional Data

Functional data clustering is performed to define homogeneous groups of similar curves in order to obtain information about the curves having similar pattern [29]. This is done by identifying some representative curves which are relevant to several shapes of variation [31]. Again, clustering depends on some similarity measures or distances. Nevertheless, a significant problem in clustering functional data is faced due to the high dimensionality of the data itself. There are several approaches for solving this problem. One is to transform the used infinite dimensional data into a finite dimensional space. Nonparametric methods are based on computing some special distances between the recovered curves after which typical clustering algorithm such as k-means clustering can be applied. In addition, there are model-based clustering techniques which depend on some mixture models [19].

Recall that, in practice, the observations are discrete. The simplest approach for functional data clustering is raw-data approach which only considers these discrete points. Thus, there is no reconstruction of the data. However, due to the high dimensionality of the discretized points, clustering algorithms for high-dimensional data should be applied. Usually, the techniques for high-dimensional data are based on the Gaussian mixture models in which each group is identified by a Gaussian probability density.

Other commonly used methods for clustering of functional observations are two-stage methods. The idea of two-stage methods is to perform clustering on finite dimensional space. In order to do this, the infinite dimensional space is projected into a smaller finite dimensional space. After that, classical clustering is applied. The projection step involves a good representation of the functional data by a finite set of basis functions. The most popular used basis is the spline basis as they have some optimal characteristics. Another

well known dimension reduction method is the functional principal component analysis (FPCA). In this method, a good approximation by a finite set of basis is used. With B-splines, the clustering is based on the coe□cient vector of the basis functions while in FPCA, the clustering is based on the first principal component scores.

4.1 Nonparametric Methods

Nonparametric functional clustering methods are separated into two classes. The methods in the first class use normal nonparametric techniques such as k-means clustering based on some special distances whereas the methods in the second class use new heuristics. We discuss the first class in more details. The proximity measure between two curves is defined by

$$d_l(x_i, x_i^{\emptyset}) = (\int_T (x_i^{(l)}(t) \quad x_i^{(l)}(t))^2 dt)^{1/2}, \tag{4.1}$$

where $x^{(l)}$ is the l derivative of x. Based on these metrics, clustering is done using k-means or hierarchical clustering. There are three usual used distances: d_0, d_1, d_2. Some authors have considered only one of these metrics while others have used a combination of these metrics.[19] For example, in [14], Ferraty et Vieu introduced the semi-metric distance which depends on the second derivative to cluster some spectrometric curves. Then they applied hierarchical clustering based on this distance. The semi-metric distance in [14] was defined by

$$d_2^{deriv}(X, Y) = \sqrt{\int_T (X^{(2)}(t) \quad Y^{(2)}(t))^2 dt}. \tag{4.2}$$

That is, the authors in [14] used d_2. Moreover, they also applied hierarchical clustering using the distance d_0 which is simply the L_2 metric. However, they noticed that, in comparison to the L_2 metric, the semi-metric has a good property that it outlines the structure of the curves. In [31], Tarpey and Kinateder proposed the metric d_0 to cluster the data using the k-means algorithm. They only considered Gaussian distributions and their result was based on the fact that for a Gaussian process, its principal points belong to a finite dimensional subspace spanned by FPCA eigenfunctions. This means that the principal points can be written as a linear combination of these eigenfunctions. In [18], the authors applied k-means algorithm with several metrics such as d_0, d_1, and $(d_0+d_1)^{\frac{1}{2}}$. In [32], Tokushige, Yadohisa and Inada applied the k-means algorithm for functional data but they introduced a new

distance which depends on the time variable t and built an objective function based on this distance. In [6], C. Abraham et al. took into consideration the function nature of the data. In order to cluster this data, they proposed to fit the curves using some models and then to cluster the data with the coe□cients of the fitted model. They suggested to use the k-means for the model coe□cients of B-splines. In [27] and [12], clustering was based on computing the distances between the curves using the di□erence between the truncated Karhunen-Loeve expansion coe□cients. In [27], they suggested a similarity measure between curves for sparse data using the conditional expectation of the L_2 distance between the curves. That is, they used

$$D(i,j) = \left\{ \int_T (X_i(t) \quad X_j(t))^2 dt \right\}^{\frac{1}{2}}, \tag{4.3}$$

$$\tilde{D}(i,j) = \left\{ E(D^2(i,j) | Y_i, Y_j) \right\}^2, \tag{4.4}$$

where $\tilde{D}(i,j)$ is the conditional expectation of the L_2 distance between the curves.

The methods in the second class use novel heuristics for clustering. The authors in [17] proposed a new functional clustering technique which aims to cluster a number of functions into k di□erent groups. In addition, they identified each group with some prototype. This technique includes two dynamic programming algorithms. In [34], the used technique was based on k-means principles and aimed to optimally cluster the functional data in a smaller subspace. It is considered to be a subspace clustering method.

4.2 Model-based Approaches

Model-based clustering has been developed to prevent the disadvantages of the dimension reduction [11]. In model-based clustering, some mixture models are usually applied and clustering is based on maximizing the posterior probability. Model-based clustering has a great benefit in creating a special framework to properly select the number of clusters. It aims to estimate a probability density function for the functional random variable. However, this is not an easy task because the concept of probability density function of a functional data does not exist. In order to solve this problem, dimension reduction is commonly used to extract the coe□cient vector of the basis functions or the first principal component scores that represent the curve characteristics. Then a probability density function of the parameters is studied. In model-based approaches, the dimension reduction step and

the clustering step are done at the same time. In comparison, in two stage methods, extracting the parameters is executed before the clustering step [10].

As a particular example, Gaussian mixture models are commonly used. For example, these are widely applied in speech recognition applications. In functional Gaussian mixture model, the parameters such as principal component scores are assumed to follow a finite mixture of probability distributions.

4.2.1 Funclust Algorithm

In this section, we discuss a clustering algorithm, specific for functional data, called Funclust [20]. Funclust is a method which depends on defining the concept of the probability densities for functional data. This can be done by using the Karhunen-Loeve expansion. As such, Funclust is an example of model-based techniques in which the clustering is based on a finite number of coe□cients in the Karhunen-Loeve expansion.

We apply the algorithm for two functional data sets. The first dataset is taken rom [3] and it describes the population rate of four various countries: Egypt, Finland, Tunisia, and Sweden. These four countries are chosen because there are some similarities between Egypt and Tunisia as well as between Finland and Sweden. This means that we should find two clusters: one cluster for Egypt and Tunisia and another cluster for Finland and Sweden. The functional data was recovered by using b-spline basis with order two and order three. Simulations are done with R.

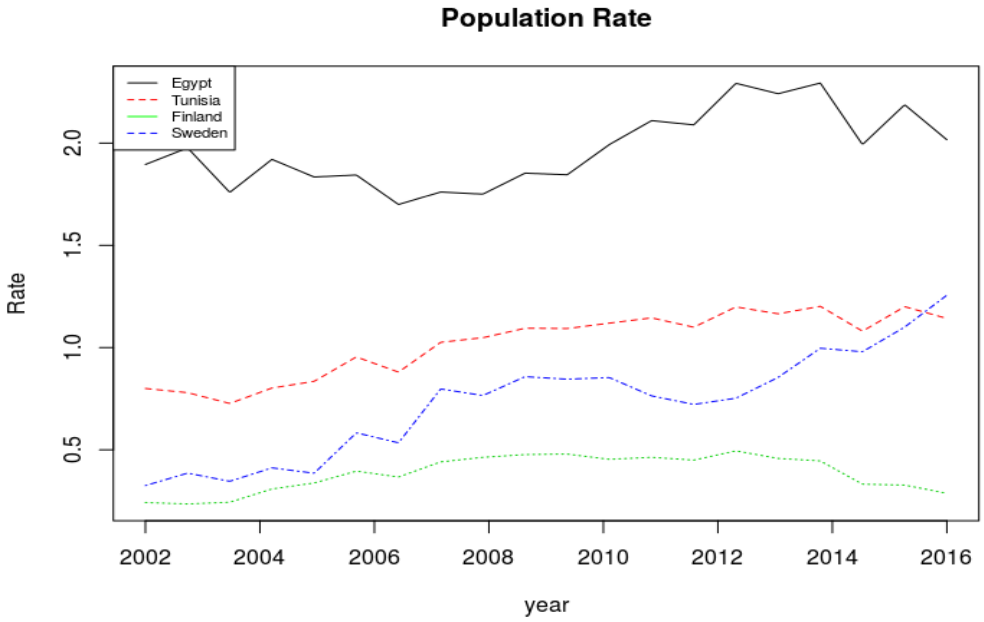


Figure 4.1: Population curves by using splines of order two.

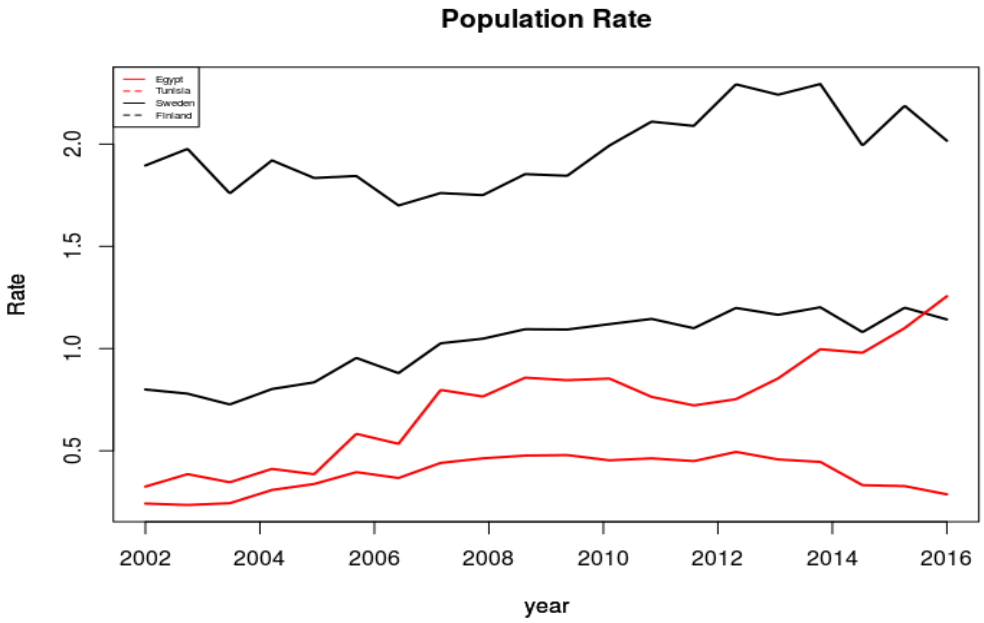


Figure 4.2: Results of population clustering. Spline order two.

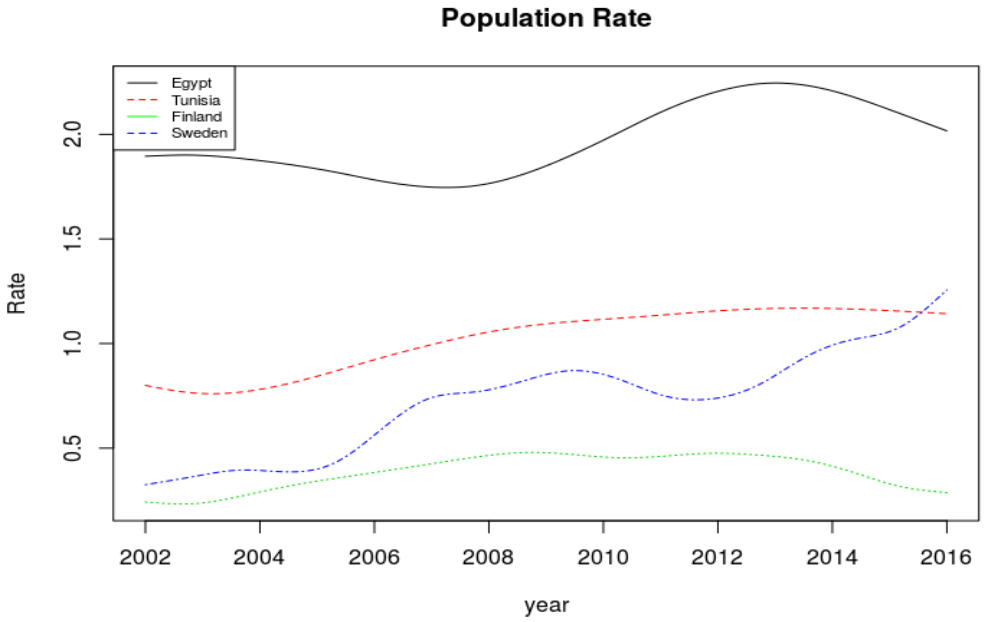


Figure 4.3: Population curves by using splines of order three.

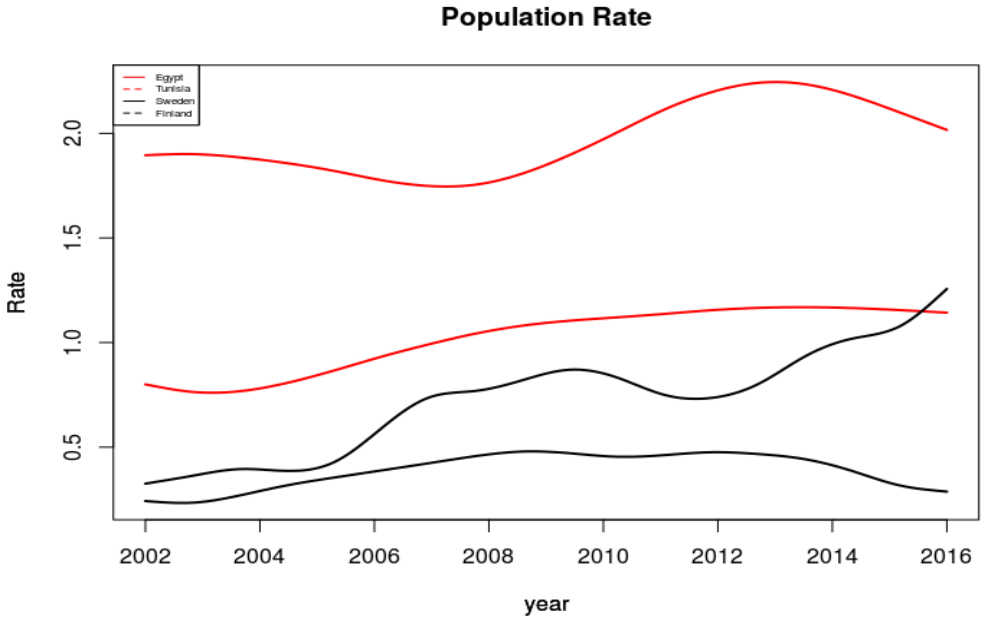


Figure 4.4: Results of population clustering. Spline order three.

It can be seen from Figures 4.1, 4.2, 4.3, and 4.4 that the order of splines has no effect on the clustering. Moreover, clustering resulted into two expected clusters: one for Egypt and Tunisia, and one for Finland and Sweden. This means that population rates for Egypt and Tunisia are similar to each other, as well as the population rates for Finland and Sweden.

As another example we consider youth unemployment rate for the same four countries. The data can be found from [5] and the curves were recovered using B-splines of order two and of order four.

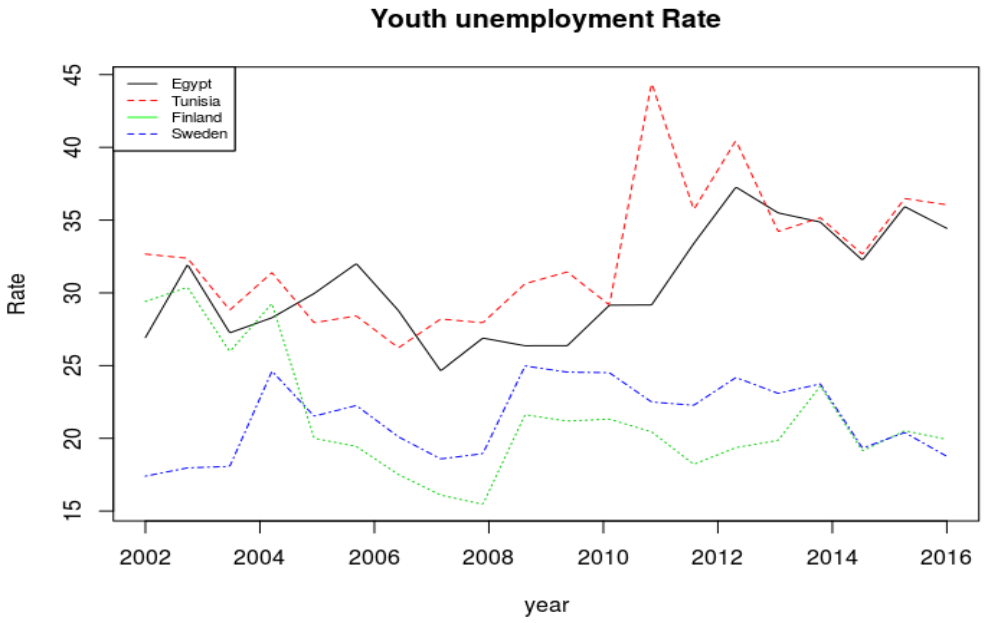


Figure 4.5: Unemployment rate curves. Spline order two.

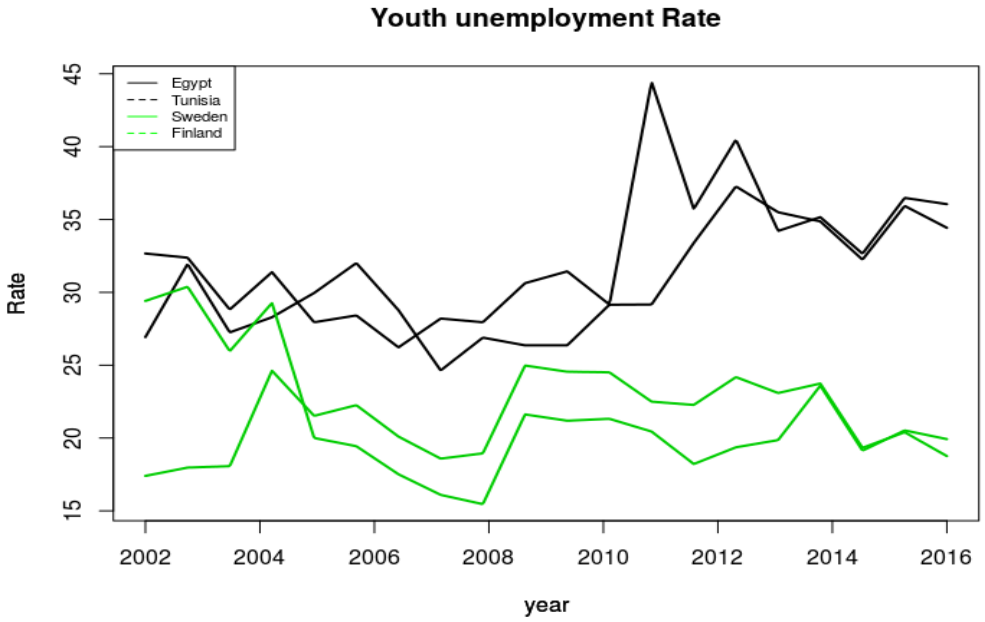


Figure 4.6: Results of unemployment clustering. Spline order two.

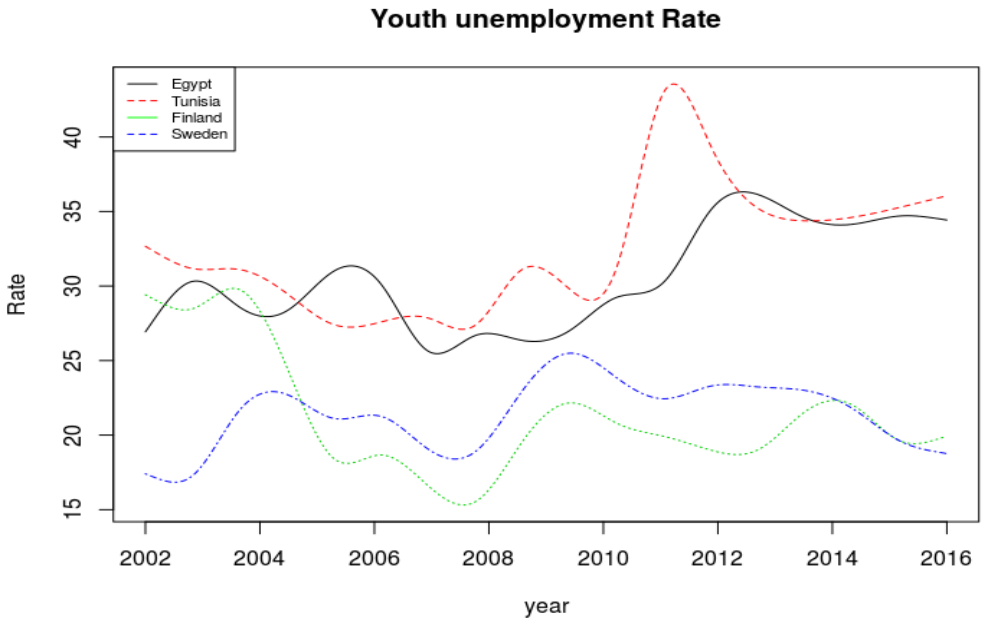


Figure 4.7: Unemployment rate curves. Spline order four.

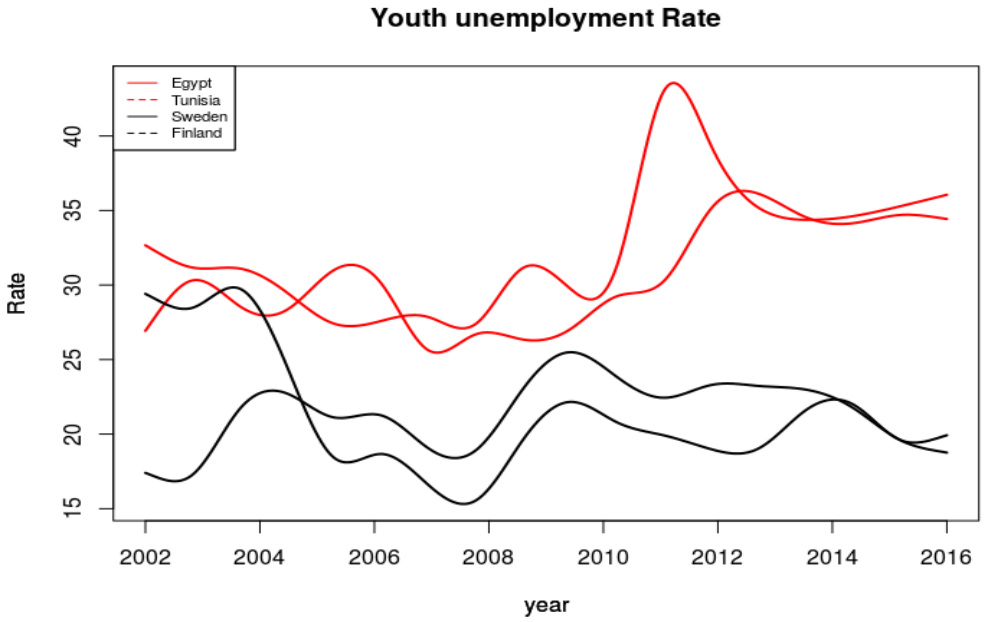


Figure 4.8: Results of unemployment clustering. Spline order four.

Similarly as in the other example, it can be seen from Figures 4.5, 4.6, 4.7, and 4.8 that the order of splines has no influence on the clustering. Moreover, as expected, clustering resulted in two clusters: one containing Egypt and Tunisia and one containing Finland and Sweden. This means that also youth unemployment rates are similar in Egypt and Tunisia, as well as in Finland and Sweden.

4.3 Benefits and Drawbacks of Functional Clustering Methods

Functional clustering methods are divided into four categories that were described briefly above. The first category is based only on the discrete points and does not take into consideration the functional nature of the data. The second category is based on a filtering step in which the functions are projected into a finite dimensional space in which the clustering is applied. The third category is based on performing the clustering step and the dimension reduction step simultaneously. Finally, the fourth category contains nonparametric methods that are called distance-based techniques. They are based on some functional distances that are used for clustering.

There are some benefits and drawbacks associated with each functional clustering method. First, raw-data clustering is a quite simple method. However, neglecting the functional nature of the data causes some problems. Second, while performing better compared to the raw-data clustering, two-stage methods have a drawback since the filtering step that is performed before the clustering can cause unreliable results. Third, model-based techniques can be optimal as long as the model fits the data well. However, assuming wrong model can cause severe problems. Finally, non-parametric methods cannot create complex cluster schemes. [21] In addition, all of the mentioned methods usually relies on the smoothness assumptions. In the next chapter, we present a non-parametric measure for the smoothness of the data itself.

Chapter 5

Clustering of non smooth observations

As discussed in previous chapters, it is usual in FDA that the curves are assumed to be smooth meaning that at least one derivative exists. However, in some cases, the data can be non-smooth. For example, this is usually the case for data consisting of financial stocks indices or heart's beating curves. As a result, assuming smoothness could corrupt the data and may result into poor analysis. Therefore, when dealing with such cases, one should apply di□erent methods or, at least, check the validity of the smoothness assumptions. However, to the best of our knowledge, the methods that measure smoothness of functional observations, are not widely acknowledged in the literature. In this thesis, we present one such measure that, for example, can be used as a preliminary step to verify smoothness assumptions. We illustrate the applicability of this measure by performing clustering to functional observations that are realizations of the fractional Brownian motions.

5.1 Fractional Brownian motion

In this section, we briefly recall the definitions and basic properties of the fractional Brownian motion. This process was first introduced by Kolmogorov [22], and nowadays it is one of the best studied and most widely applied process. Fractional Brownian motion depends on one parameter, the Hurst index. This name comes from Hurst who applied fractional Brownian motion in hydrology. Hurst studies specifically the River Nile.[25]

In this thesis we consider stochastic processes $(X_t)_{t2[0,1]}$ on an interval $[0, 1]$.

Definition 5.1 (Gaussian Process). A stochastic process $(X_t)_{t2[0,1]}$ is a Gaussian process if for all $0 \quad t_1 \quad t_2 ..... \quad t_n$, the random vector $(X_{t_1}, ..., X_{t_n})$ on R^n is normally distributed.

It is known that a Gaussian process is completely characterized by its mean and covariance functions. The covariance function of a Gaussian process is defined as:

$$cov(X_t, X_s) = E((X_t \quad E(X_t))(X_s \quad E(X_s)), \tag{5.1}$$

where $E(X_t)$ is the mean function of the Gaussian process.
A process is called centered if $E(X_t) = 0$ for all t.

Definition 5.2 (Fractional Brownian motion). Fractional Brownian motion $B^H = fB_t^H, t \quad 0g$ with Hurst index H 2 $(0, 1)$ is a centered Gaussian process with covariance function

$$R_H(s, t) = \frac{1}{2}(t^{2H} + s^{2H} \quad jt \quad sj^{2H}). \tag{5.2}$$

Fractional Brownian motion has several interesting properties that makes it a suitable process for di□erent applications. First of all, it is H self-similar. Recall that a process is self-similar if the two processes $fa \ ^H B_{at}^H, t \quad 0g$ and $fB_t^H, t \quad 0g$ have the same finite dimensional probability distributions. In the Gaussian case, as the distribution is completely characterized by the mean and covariance functions, self-similarity is equivalent to the homogeneity of the covariance function defined by (5.2). In addition, Fractional Brownian motion has stationary increments. That is, the increments $X_k = B_{k+1}^H \quad B_k^H$ forms a stationary process meaning that the finite dimensional distributions of X are invariant under time shifts. Another property of Fractional Brownian motion is the long range dependence property for H 2 $(\frac{1}{2}, 1)$. This means that the covariance function [26]

$$r(n) = E[B^H(1)(B^H(n+1) \quad B^H(n)] \tag{5.3}$$

of the increments satisfies

$$\sum_{n=1}^{1} r(n) = 1 . \tag{5.4}$$

Some pictures for di□erent Fractional Brownian motions with various indices are shown below.

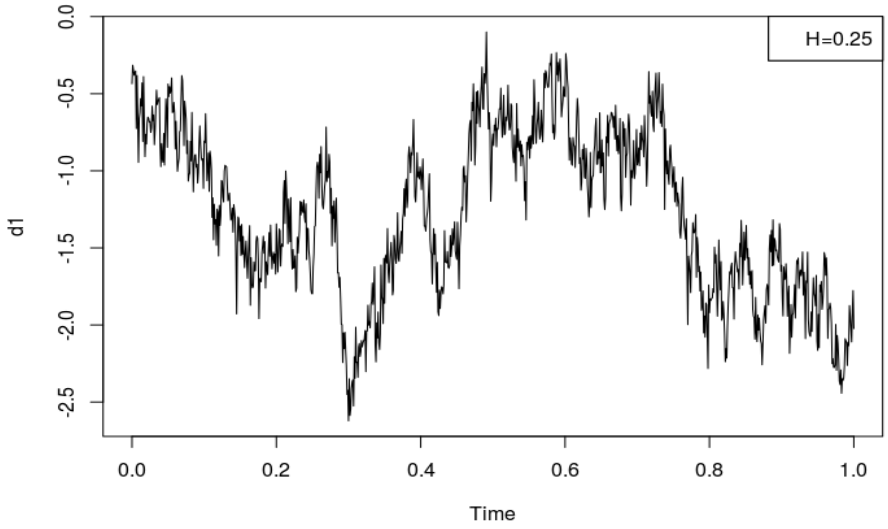


Figure 5.1: Fbm with Hurst index $H = 0.25$.

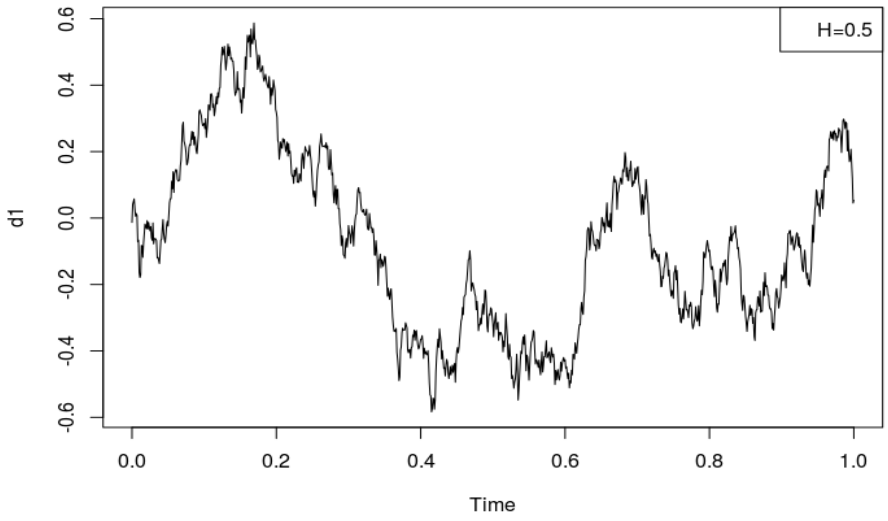


Figure 5.2: Fbm with Hurst index $H = 0.5$.

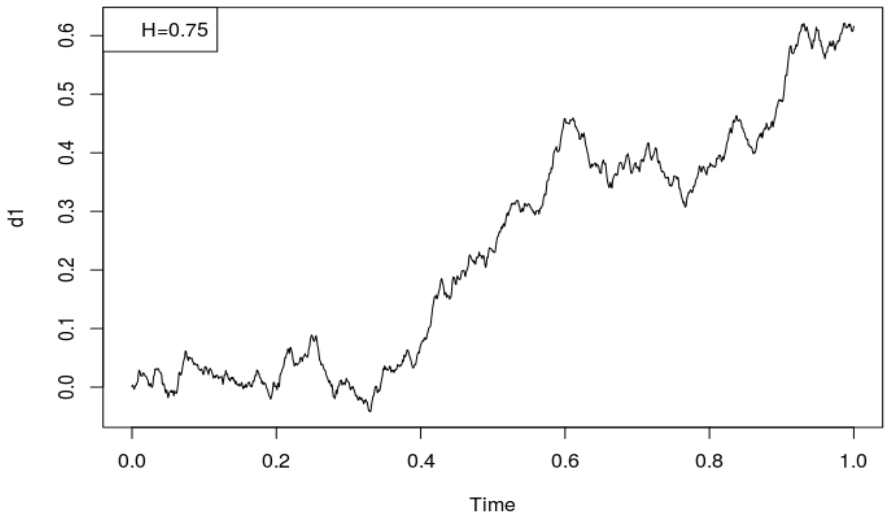


Figure 5.3: Fbm with Hurst index $H = 0.75$.

It can be seen from Figures 5.1, 5.2, and 5.3 that the paths of the fractional Brownian motions are not smooth. On the other hand, they behave very differently for different Hurst indices. Suppose we aim to distinguish the paths of two fractional Brownian motions with different but close Hurst indices. The paths are illustrated in Figure 5.4.

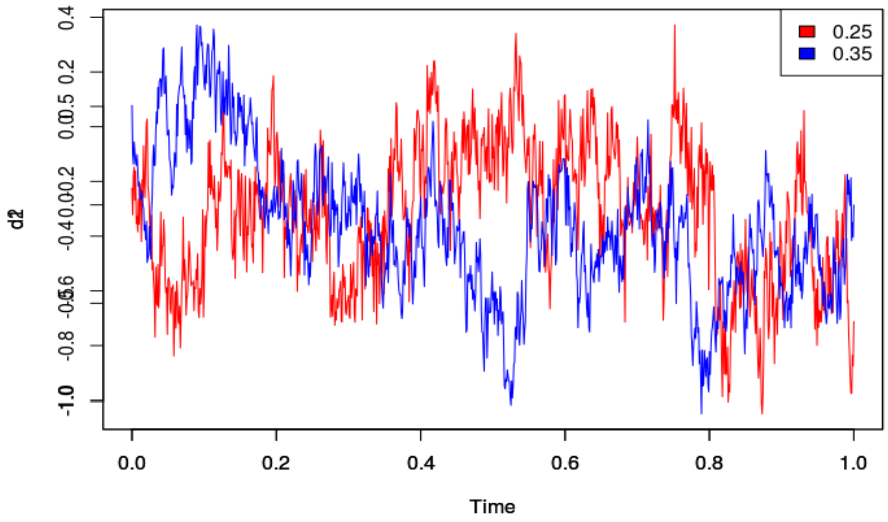


Figure 5.4: Two Fbms with indices=0.25 and 0.35.

The key concept is the Hölder continuity of the paths.

Definition 5.3 (Hölder Continuity). Let α *2* $(0, 1]$. We say that a function f is α Hölder continuous if there exists a constant C such that, for all x, y, we have

$$jf(x) \quad f(y)j \quad Cjx \quad yj^{\alpha}. \tag{5.5}$$

Note that $\alpha = 1$ corresponds to the smooth case.

From the covariance function defined in Equation (5.2), it can be seen that

$$E((B_t^H \quad B_s^H)^2) = jt \quad sj^{2H}. \tag{5.6}$$

Since B^H is Gaussian, it follows that

$$E(jB_t^H \quad B_s^H j^k) = C_k jt \quad sj^{kH}. \tag{5.7}$$

Thus Kolmogorov's continuity theorem implies that the paths of the fractional Brownian motion are $H \quad \epsilon$ Hölder continuous for any $\epsilon > 0$.

5.2 Measure For Roughness

Let us now introduce our metric for smoothness. We assume that each observation x_i is Hölder continuous of some order α_i. The idea is to estimate

α_i. Note however that if x_i is Hölder continuous of some order α_i, then it is also Hölder continuous of any order $\gamma < \alpha_i$. Thus, we try to identify largest possible value α_i such that

$$\frac{jx_i(t) \quad x_i(s)j}{jt \quad sj^{\alpha_i}} \quad C < 1 . \tag{5.8}$$

After such α_i is identified for each observation, we can perform clustering based on the estimated values. In the best, we obtain cluster consisting of smooth curves with value $\alpha_i = 1$ and other clusters containing the non-smooth observations.

In order to identify the largest possible value of α_i, suppose that

$$\frac{jX_t \quad X_sj}{jt \quad sj^H} \quad C \tag{5.9}$$

for some H. Note that usually the problematic case is when t, s are close to each other. Let $jt \quad sj < 1$. We take logarithm on both sides and get

$$\log jX_t \quad X_sj \quad \log jt \quad sj^H \quad \log C.$$

Using $\log jt \quad sj^H = H \log jt \quad sj$, dividing with $\log jt \quad sj$ together with $jt \quad sj < 1$ yields

$$\frac{\log jX_t \quad X_sj}{\log jt \quad sj} \quad H \quad \frac{\log C}{\log jt \quad sj},$$

where $t \neq s$ and $jt \quad sj < 1$. This leads to the following definition.

Definition 5.4. Let $t_k = \frac{k}{n}$ be a uniform partition of $[0, 1]$. We define the estimator $\hat{H}$ for an index H by

$$\hat{H} = \min_k \frac{\log jX_{t_k} \quad X_{t_k \ 1}j}{\log jt_k \quad t_k \ 1j}. \tag{5.10}$$

The following proposition and examples should convince the reader that the defined estimator $\hat{H}$ is reasonable.

Proposition 5.5. *Let X be H Hölder continuous. Then*

$$\liminf_n \hat{H} \quad H$$

Proof. Recall that now

$$\frac{\log jX_t \quad X_s j}{\log jt \quad sj} \quad H \quad \frac{\log C}{\log jt \quad sj}.$$

Plugging $t = t_k$ and $s = t_{k\ 1}$ into the equation gives

$$\hat{H} \quad H \quad \frac{\log C}{\log jt_k \quad t_{k\ 1} j}.$$

Passing to the limit proves the claim. □

Example 5.6. Let $X_t = t^H$, H 2 $(0, 1)$. A picture of such function is presented in Figure 5.5.

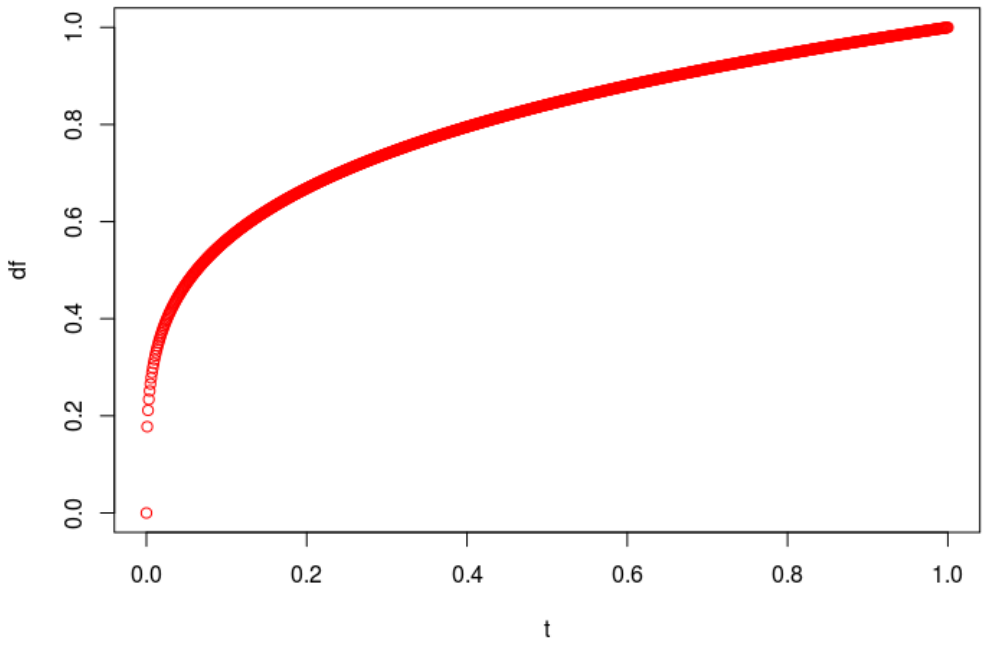


Figure 5.5: Fractional power function with index=0.25.

Now

$$\begin{aligned}\hat{H} &= \frac{\log[(\frac{k}{n})^H \quad (\frac{k\ 1}{n})^H]}{\log\frac{1}{n}} \\ &= \min_k \frac{\log[(\frac{1}{n})^H[k^H \quad (k \quad 1)^H]]}{\log\frac{1}{n}} \\ &\quad \frac{\log(\frac{1}{n})^H}{\log\frac{1}{n}} \\ &\quad H.\end{aligned}$$

On the other hand, by Proposition 5.5 we have

$$\liminf_n \hat{H} \quad H.$$

Thus $\lim \hat{H} = H$.

Proposition 5.7. *Let X be continuously di erentiable with non-vanishing derivative* X^0. *Then* $\hat{H}$! 1.

Proof. By the mean value theorem,

$$jX_{t_k} \quad X_{t_k\ 1}j = jX^0_{\epsilon_k}jjt_k \quad t_k\ 1j \tag{5.11}$$

for some

$$\epsilon_k\ 2\ [t_k\ 1, t_k]$$

. Therefore,

$$\begin{aligned}\min \frac{\log jX_{t_k} \quad X_{t_k\ 1}j}{\log jt_k \quad t_k\ 1j} &= \min_k \left[1 \quad \frac{\log jX^0_{\epsilon_k}j}{\log n}\right] \\ &= 1 \quad \max_k \frac{\log jX^0_{\epsilon_k}j}{\log n} \\ &= 1 \quad \frac{\log \max_k jX^0_{\epsilon_k}j}{\log n}\end{aligned}$$

where we have used $\min \quad f(x) = \quad \max f(x)$ and the fact that logarithm is an increasing function. Now $\max_k jX^0_{\epsilon_k}j$! $\sup_{s2[0,1]} jX^{0j>0}_s$ from which the claim follows. □

Next we show that $\hat{H}$ defined in (5.10) is consistent also in the case of Fractional Brownian Motion. For this, we need the following result taken from [9].

Theorem 5.8. *Let ϵ_k be a stationary Gaussian sequence with covariance $r(k)$. If $r(n)\log n \to 0$, then*

$$\frac{\max_{1 \le k \le n} |\epsilon_k|}{\sqrt{2\log n}} \xrightarrow{p} 1$$

.

Theorem 5.9. *Let X be a fractional Brownian motion with $H \in (0,1)$ and let $\hat{H}$ be defined by* (5.10). *Then $\hat{H} \to H$ in probability.*

Proof. By self-similarity property for fractional Brownian motion, we have

$$|X_{\frac{k}{n}} - X_{\frac{k-1}{n}}| \overset{Law}{=} n^{-H} |X_k - X_{k-1}|$$

Moreover, since the increments of the fractional Brownian motion are stationary, the sequence $\epsilon_k = X_k - X_{k-1}$ is a stationary Gaussian sequence. In addition, it is known that the covariance $r(k)$ of ϵ_k satisfies $r(k) = E\{\epsilon_{k+v}\epsilon_v\} \sim k^{2H-2}$ as $k \to \infty$. Hence by Theorem 5.8, we have

$$\frac{\max |\epsilon_k|}{\sqrt{2\log n}} \xrightarrow{p} 1$$

in probability. This leads to

$$\begin{aligned}
\hat{H} &= \min_k \frac{\log |X_{t_k} - X_{t_{k-1}}|}{\log |t_k - t_{k-1}|} \\
&\overset{Law}{=} \min_k \frac{\log n^{-H} |X_k - X_{k-1}|}{\log \frac{1}{n}} \\
&= H - \frac{\log \max_k |\epsilon_k|}{\log n} \\
&= H - \frac{\log \frac{\max_k |\epsilon_k|}{\sqrt{2\log n}}}{\log n} - \frac{\log \sqrt{2\log n}}{\log n}.
\end{aligned}$$

By Theorem 5.8, we have

$$\frac{\log \frac{\max_k |\epsilon_k|}{\sqrt{2\log n}}}{\log n} \xrightarrow{p} 0.$$

Clearly, we also have

$$\frac{\log \sqrt{2\log n}}{\log n} \to 0.$$

This concludes the proof. □

Note that although $\hat{H}$ is consistent in the case of the fractional Brownian motion, the rate of convergence is very poor. Indeed, this follows from the fact that $\frac{\log\sqrt{2\log n}}{\log n}$ converges to 0 very slowly. Indeed, to obtain 0.01 accuracy, we need that $\frac{\log\sqrt{2\log n}}{\log n} \quad 0.01$. Therefore, $\frac{1}{\log n} \quad 0.01$ which leads to $n \quad \exp 100$. Thus we introduce a modified estimator by

$$\tilde{H} = \hat{H} + \frac{\log\sqrt{2\log n}}{\log n}. \tag{5.12}$$

It can be proved that the rate of convergence for $\tilde{H}$ is $O(\frac{1}{(\log n)^2})$ while for $\hat{H}$, the rate of convergence is $O(\frac{\log\log n}{\log n})$. Note also that in order to achieve 0.01 accuracy for $\tilde{H}$, we need

$$\frac{1}{(\log n)^2} \quad 1$$

leading to $n \quad \exp 10$. Therefore n must be greater than 22000 which is still computationally feasible.

5.3 Clustering Based on Roughness Distance

In this section, we perform clustering based on the distance introduced above. We illustrate our approach by clustering simulated paths of fractional Brownian motions. For this reason, we use the modified distance (5.12) with better rate of convergence. Throughout this section, k-means clustering algorithm is used. In most of the cases, we set the number n of evaluation points to be 1000, 10000, or 100000. That is, the time interval $[0, 1]$ is divided into n equidistant sub-intervals.

5.3.1 Clustering of two groups of fractional Brownian motions with two indices that are relatively far from each other

Clustering is performed for 20 fractional Brownian motions (fBms) with indices 0.25 and 0.5. That is, we cluster 10 paths of fractional Brownian motions with index 0.25 with 10 paths of fractional Brownian motions with index 0.5. After the paths are simulated, the estimated index is computed for each path based on 1000, 10000, or 100000 evaluation points.

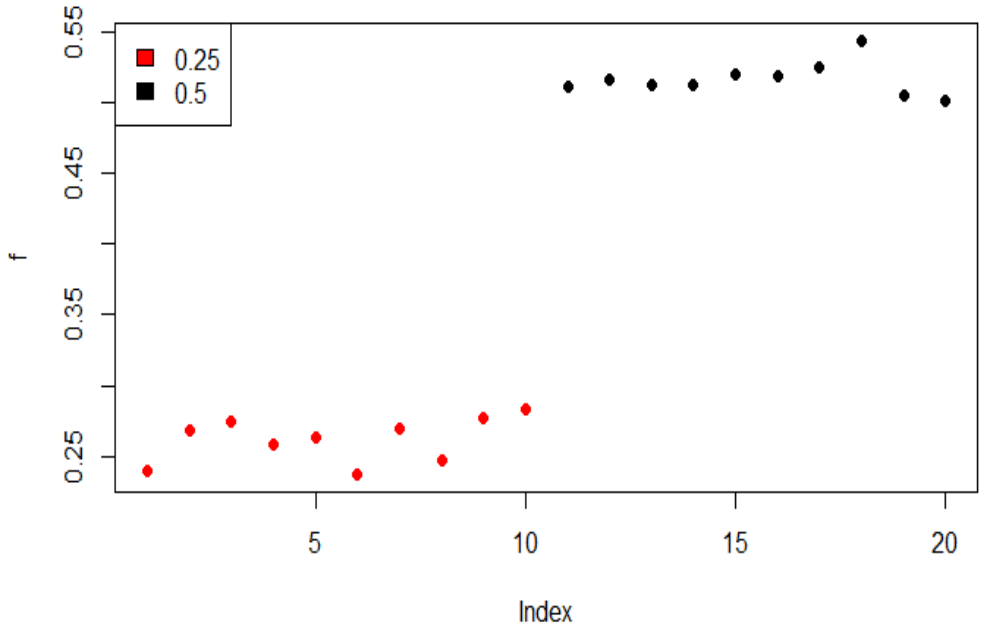


Figure 5.6: Clustering of 20 paths based on 1000 evaluation points. True indices are 0.25 and 0.5.

	index=0.25	index=0.5
min	0.2375336	0.5010077
max	0.2837742	0.5437011
mean	0.2620984	0.5167019

Table 5.1: Values for the group statistics. True indices are 0.25 and 0.5.

Figure 5.6 represents the results based on 1000 evaluation points. Clustering success is obvious. The 20 functions are split into two classes: one with index=0.25 and another one with index=0.5. However, the index is slightly overestimated by 0.012 although this is rather small error. The estimated index for 0.25 is about 0.262 whereas the estimated one for 0.5 is approximately 0.5167. Table 5.1 illustrates the statistics of each class.

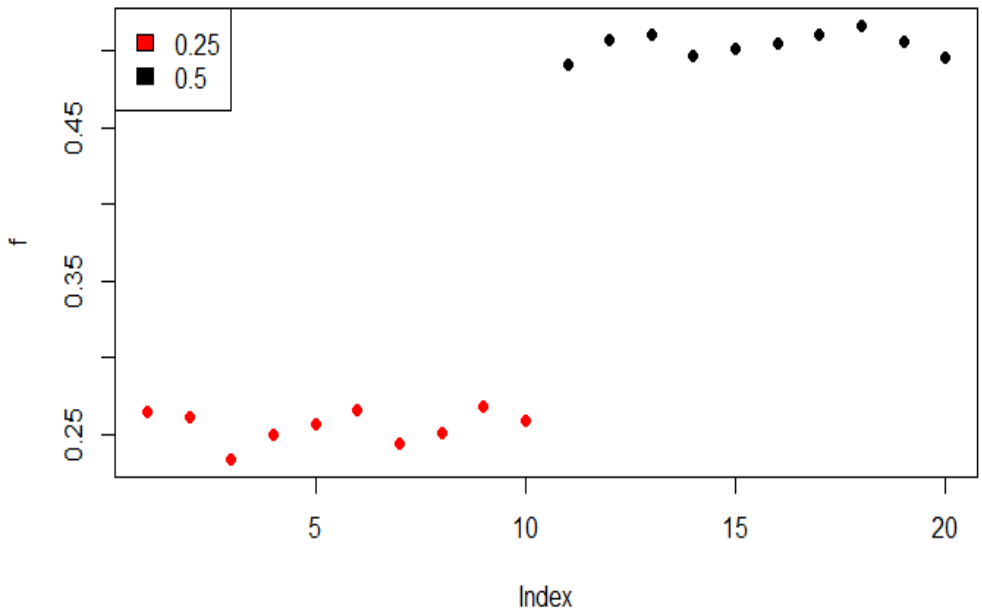


Figure 5.7: Clustering of 20 paths based on 10000 evaluation points. True indices are 0.25 and 0.5.

	index=0.25	index=0.5
min	0.2342302	0.4909899
max	0.2683832	0.5161249
mean	0.2556594	0.5038768

Table 5.2: Values for the group statistics. True indices are 0.25 and 0.5.

Clustering results based on 10000 evaluation points is shown in Figure 5.7. Again, the 20 functions are classified correctly. Moreover, the values of the estimated indices are improved. For example, the estimated index for 0.25 is approximately 0.2556 while the estimated one for 0.5 is approximately 0.5039. This means that increasing the number of evaluation points to 10000 points slightly improves the results as it decreases the error proportional to $\frac{1}{(\log n)^2}$. Table 5.2 represents the statistics of each class of functions.

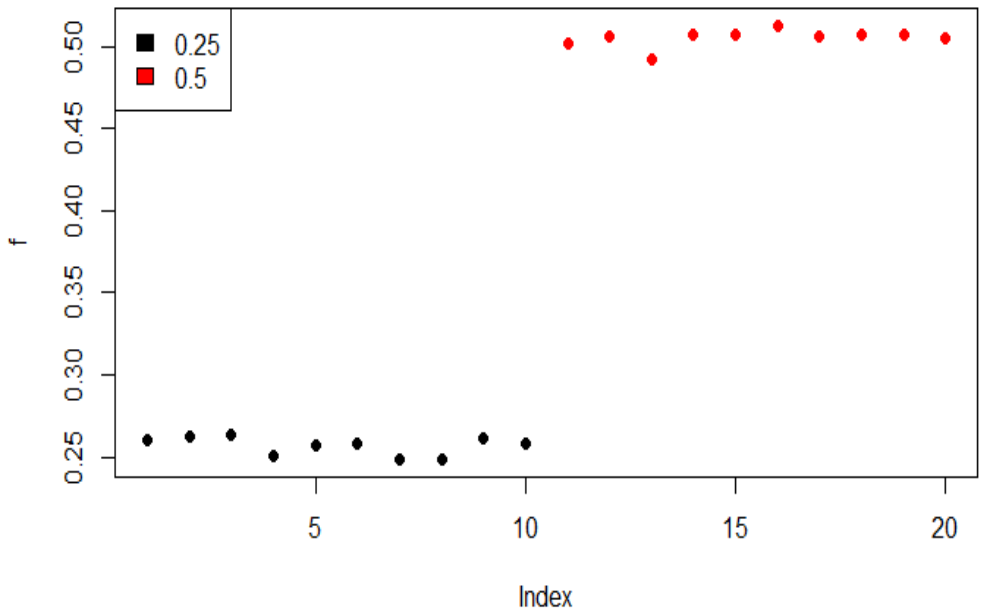


Figure 5.8: Clustering of 20 paths based on 100000 evaluation points. True indices are 0.25 and 0.5.

	index=0.25	index=0.5
min	0.2487222	0.4924642
max	0.2634939	0.5125276
mean	0.2567637	0.5052803

Table 5.3: Values for the group statistics. True indices are 0.25 and 0.5.

Finally, changing the number of evaluation points from 10000 to 100000 improves gradually the estimation results and, as expected, there is no classification error. For example, the estimated index for 0.25 is approximately 0.25676 while the estimated index for 0.5 is approximately 0.5052803. The results are presented in Figure 5.8 and Table 5.3.

5.3.2 Clustering of two groups of fractional Brownian motions with two indices that are relatively close to each other

In previous subsection, the di□erence between the two indices was relatively large. As a result, we did not obtain classification errors even with small number of evaluation points. In this subsection, we study the e□ect of having two indices relatively close to each other. We select two indices 0.25 and 0.3 and compare the classification errors with the number of evaluation points equal to 1000, 10000, and 100000. The results are illustrated in Figures 5.9, ??, and 5.11. The summary statistics are presented in Tables , , and .

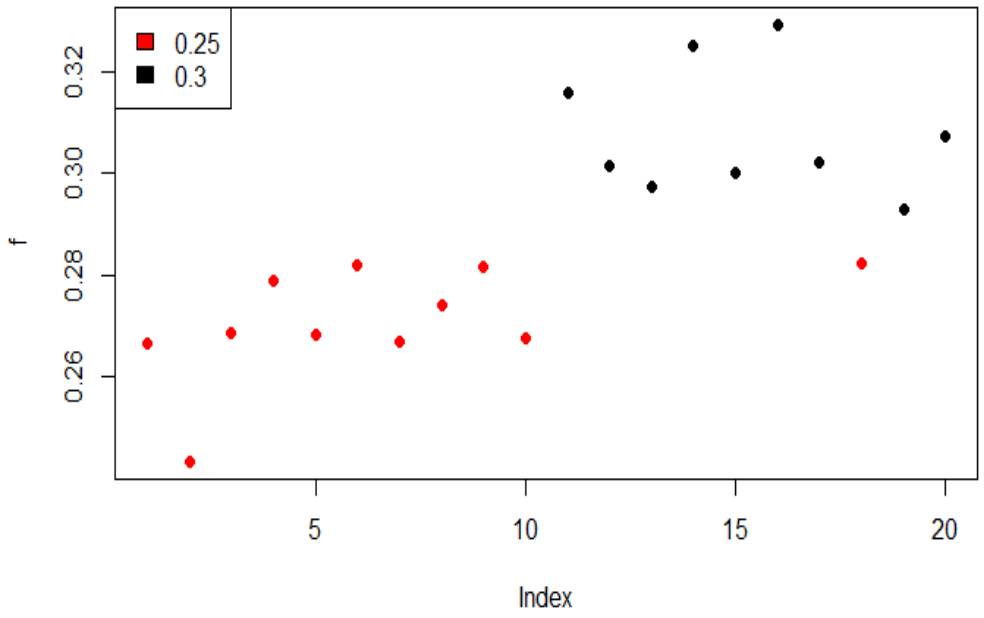


Figure 5.9: Clustering of 20 paths based on 1000 evaluation points. True indices are 0.25 and 0.3.

	index=0.25	index=0.3
min	0.2435931	0.2821857
max	0.2819484	0.3290261
mean	0.2698911	0.3053362

Table 5.4: Values for the group statistics. True indices are 0.25 and 0.3.

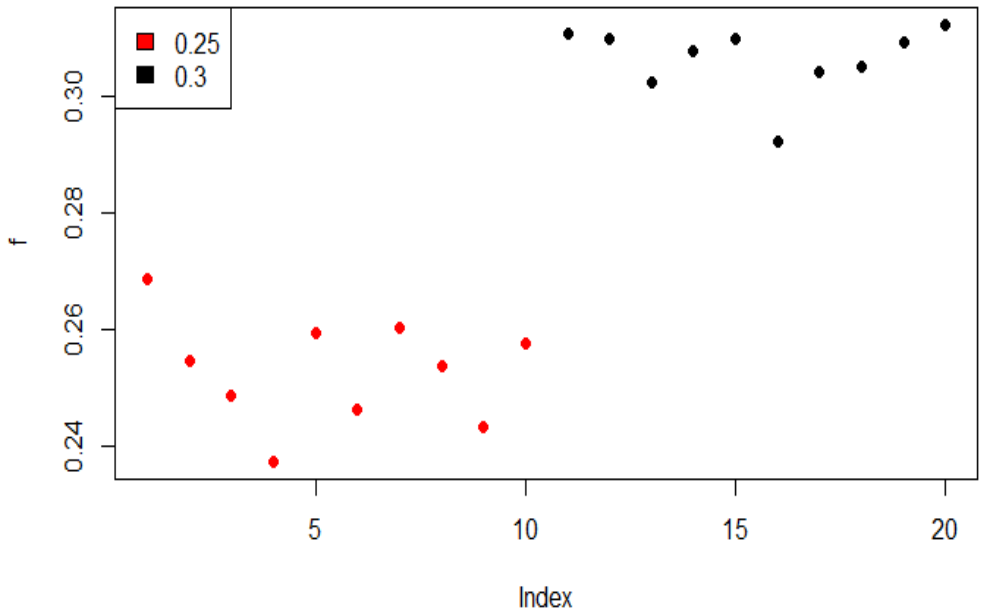


Figure 5.10: Clustering of 20 paths based on 10000 evaluation points. True indices are 0.25 and 0.3.

	index=0.25	index=0.3
min	0.2374151	0.2922893
max	0.2687104	0.3120891
mean	0.2529816	0.3062863

Table 5.5: Values for the group statistics. True indices are 0.25 and 0.3.

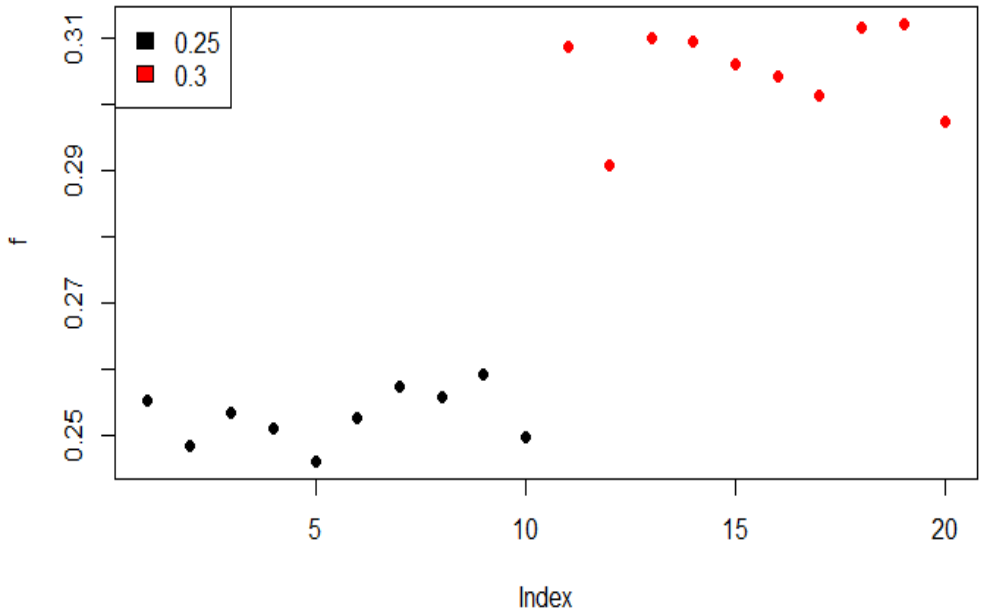


Figure 5.11: Clustering of 20 paths based on 100000 evaluation points. True indices are 0.25 and 0.3.

	index=0.25	index=0.3
min	0.2461483	0.290716
max	0.2591794	0.3119534
mean	0.2528554	0.3050574

Table 5.6: Values for the group statistics. True indices are 0.25 and 0.3.

With 1000 evaluation points, we obtain one misclassified function (see Figure 5.9) which corresponds to the classification error rate of 5 %. This is due to the closeness between the two true indices and the dispersion within each class with 1000 evaluation points. However, the clustering becomes perfect when the number of evaluation points is increased to 10000 or 100000.

5.3.3 E□ect of number of clusters

In this subsection we study the e□ect that the number of natural clusters has to the clustering results. In addition, we study the e□ect of the chosen

number k. The clustering results of 20 paths of fractional Brownian motions into three classes are presented in Figures 5.12 and 5.13 and Table 5.7. The true indices are 0.25 and 0.35 and the used number of evaluation points is equal to 1000 or 10000.

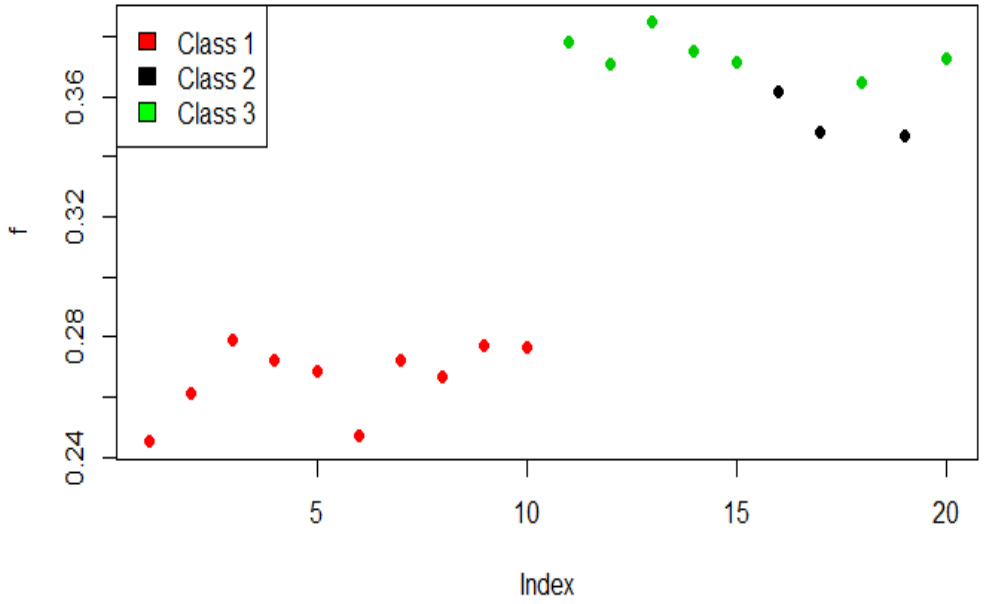


Figure 5.12: Clustering of 20 paths into three groups based on 1000 evaluation points. True indices are 0.25 and 0.35.

	Class 1	Class 2	Class 3
min	0.2451027	0.3465991	0.3647249
max	0.278715	0.3613890	0.3845858
mean	0.2665197	0.3520404	0.3737819

Table 5.7: Values for the group statistics when clustering into three classes. True indices are 0.25 and 0.35.

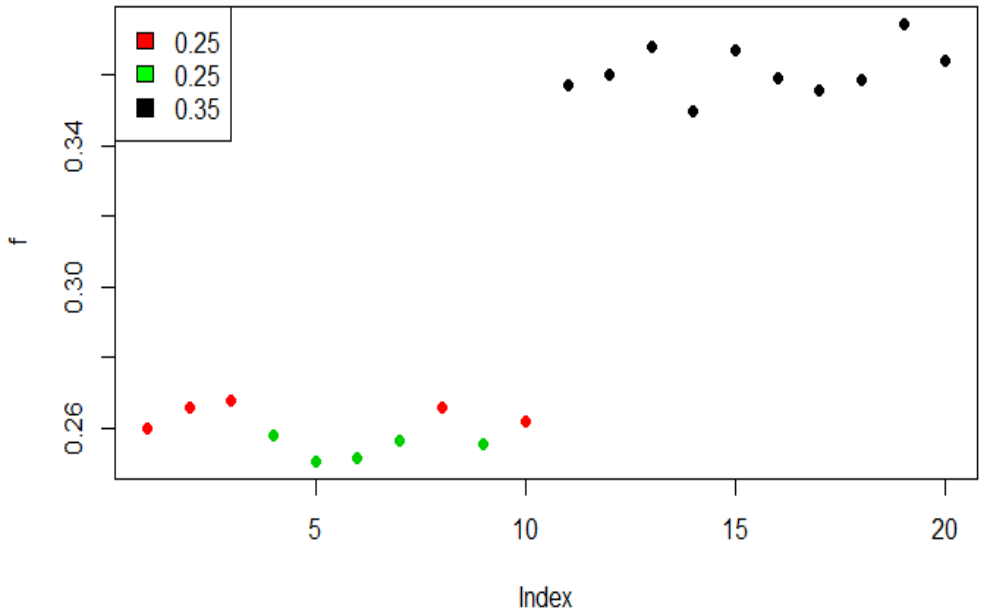


Figure 5.13: Clustering of 20 paths into three groups based on 10000 evaluation points. True indices are 0.25 and 0.35.

Clustering (erroneously) of two natural groups into three classes only results in dividing one natural group into two classes.

Next, we study the case of having three natural groups that are clustered only into two classes. The (correct) clustering results into three classes are presented in Figure 5.14 and Table 5.8 while the clustering results into two classes are shown in Figure 5.15 and Table 5.9. True indices are 0.25, 0.35, and 0.45 and the used number of evaluation points is 100000.

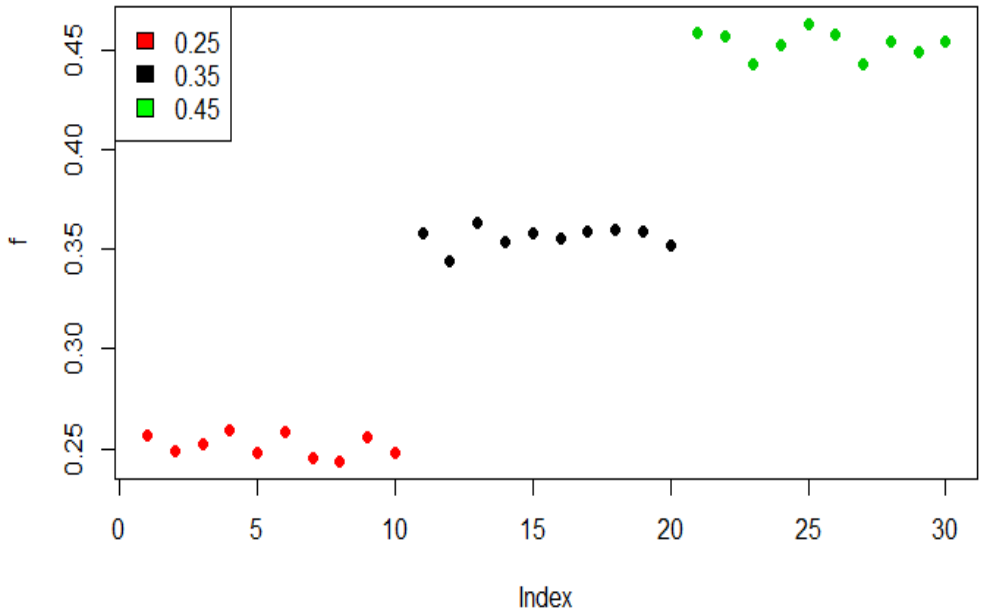


Figure 5.14: Clustering of 30 paths into three classes based on 100000 evaluation points. True indices are 0.25, 0.35, and 0.45.

	index=0.25	index=0.35	index=0.45
min	0.2438431	0.344028	0.4428062
max	0.2592672	0.3629713	0.4629843
mean	0.251538	0.356235	0.4534503

Table 5.8: Values for the group statistics when clustering into three classes. True indices are 0.25, 0.35, and 0.45.

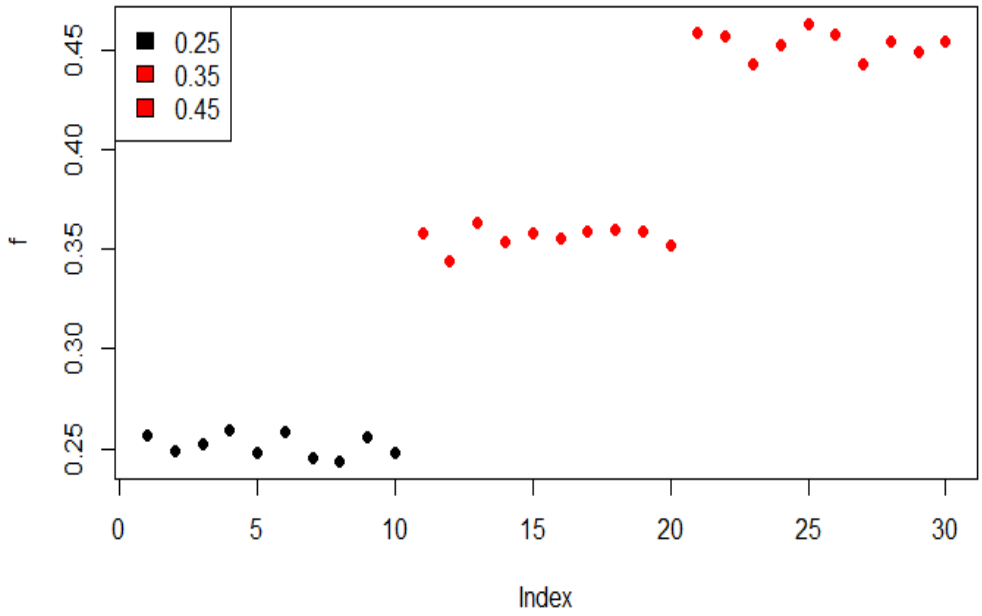


Figure 5.15: Clustering of 30 paths into two classes based on 100000 evaluation points. True indices are 0.25, 0.35, and 0.45

	Class 1	Class 2
min	0.2438431	0.344028
max	0.2592672	0.4629843
mean	0.251538	0.4048426

Table 5.9: Values for the group statistics when clustering into two classes. True indices are 0.25, 0.35, and 0.45.

In this case, two of the natural clusters are merged. This leads to higher deviation in the corresponding cluster. The average index is 0.4048426 that is between the true indices 0.35 and 0.45. However, there are no estimated values that are close to this average. Together with high deviation, this hints that the number of used clusters is selected wrongly.

5.3.4 Results with power function

We end this chapter by studying clustering results in a case when fractional power function is added to the dataset consisting of 20 paths of fractional Brownian motions. In this case, it does not matter to the clustering results whether one adds the correction term or not. However, it has a crucial effect on the group statistics. The reason is that when the correction term is added, the index for fractional power function is overestimated. On the other hand, without the correction term, the estimated indices for fractional Brownian motions are underestimated. As, a priori, one can not say from a given data which functions are Gaussian (that need the correction term) and which are not, this is a source of possible problem in practice.

We compare the clustering results between adding the correction term and without. We consider a data consisting of one power function with index 0.25, and 20 fractional Brownian motions with indices 0.25 and 0.35. The data is clustered into two groups corresponding to the number of natural clusters. We use only 1000 evaluation points to highlight the above mentioned problem.

The results of clustering with the correction term are presented in Figure 5.16 and Tables 5.10 and 5.11. The index of the power function is clearly overestimated (0.4400612) and consequently, it is assigned to the wrong cluster.

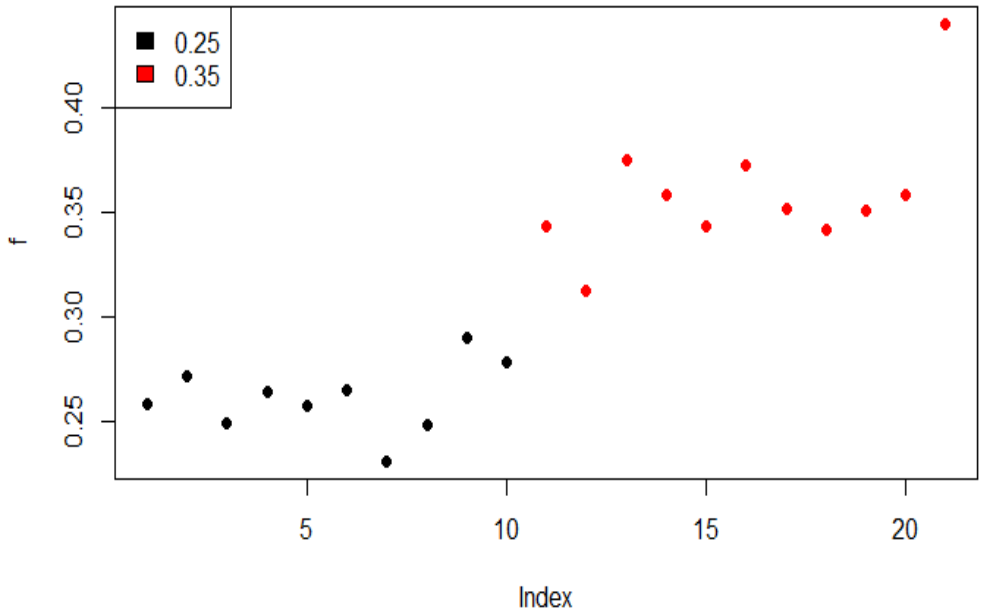


Figure 5.16: Clustering of 20 paths and one fractional power function with adding the correction term based on 1000 evaluation points.

	index=0.25	index=0.35
min	0.2312538	0.3129045
max	0.2901134	0.3750166
mean	0.261529	0.3507946

Table 5.10: Values for the group statistics of the 20 paths of fractional Brownian motions with adding the correction term. True indices are 0.25 and 0.35.

	index=0.25	index=0.35
min	0.2312538	0.3129045
max	0.2901134	0.4400612
mean	0.261529	0.3589097

Table 5.11: Values for the group statistics when clustering the 20 paths and one fractional power function with adding the correction term.

The results of clustering without adding the correction term are shown in Figure 5.17 and Tables 5.12 and 5.13. Again, the power function is assigned to the wrong cluster. In this case, while the estimated index of the power function is equal to the correct one, the indices of the fractional Brownian motions are highly underestimated.

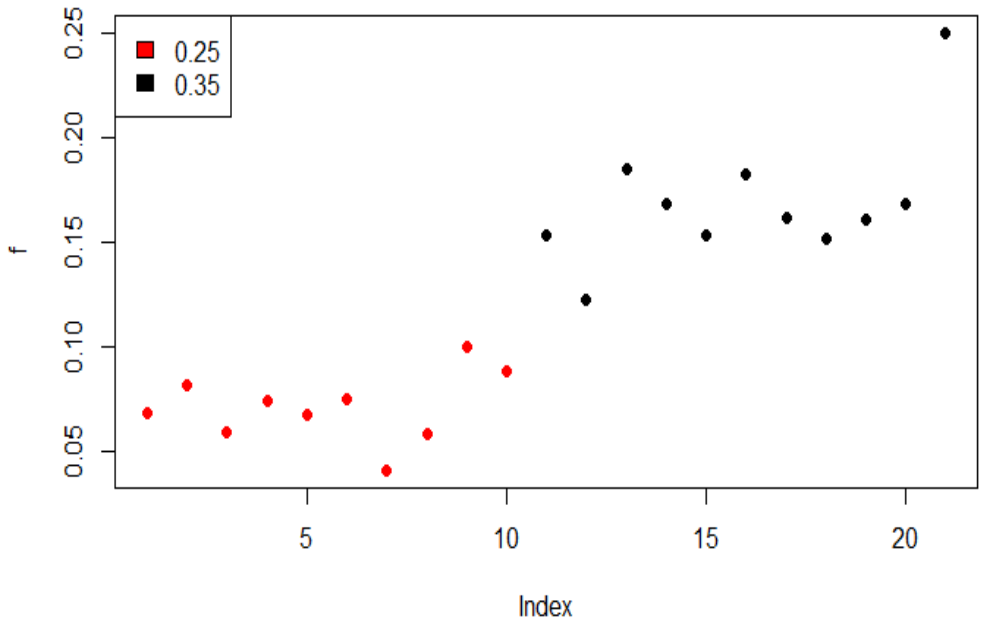


Figure 5.17: Clustering of 20 paths and one fractional power function without the correction term based on 1000 evaluation points.

	index=0.25	index=0.35
min	0.0411926	0.1228433
max	0.1000523	0.1849554
mean	0.07146786	0.1607334

Table 5.12: Values for the group statistics of the 20 paths of fractional Brownian motions without the correction term.

	index=0.25	index=0.35
min	0.0411926	0.1228433
max	0.1000523	0.25
mean	0.07146786	0.1688485

Table 5.13: Values for the group statistics when clustering of 20 paths and one fractional power function without adding the correction term.

In practice, one can not know whether correction term should be added or not. Thus, one should either use very high number of evaluation points (which is unfeasible due to high computational cost), or perform some preliminary analysis of the data such as testing for Gaussianity.

Chapter 6

Conclusion

In this thesis, we introduce a novel method for clustering functional observations. The method is based on measuring the roughness of these observations. The metric is based on the assumption of Hölder continuity. The method is applied to cluster two types of functions: fractional Brownian motions and power fractional functions.

We discuss functional data analysis, basis functions, and data fitting. Moreover, various functional clustering methods such as raw-data clustering, two-stage methods, nonparametric methods, and model-based clustering methods are reviewed. In our novel clustering approach, we apply the k-means algorithm. Thus, we explain the basic idea of k-means clustering. In addition, we consider problems in k-means clustering and solutions for these problems.

We consider theoretical properties of the proposed roughness measure. The most important result we obtain is that, under fractional Brownian motions, our roughness index converges to the corresponding Hurst index of this process. This explains the excellent performance of our clustering approach.

In addition to theoretical properties, we assess the performance of the approach in practice. We present several simulated examples. The estimated index combined with k-means clustering works successfully in the case of having the same type of functions. For example, we illustrate that it has the ability to cluster three groups of fractional Brownian motions with three different indices into three classes extremely well. However, if the Hurst indices of the processes are very close to each other, we might have some classification errors. If the number of evaluation points is large enough, the method works extremely well also in that case. In general, in order to obtain

excellent clustering, the method requires large number of evaluation points. Indeed, it is advisable to select the number of evaluation points to be between $[10000, 100000]$. In case the dataset consists of di□erent types of functions for example, fractional Brownian motions and power fractional function, our new approach does not provide excellent classification results.

6.1 Future Work

In this thesis, we apply our method to two types of functions: fractional Brownian motions and power fractional function. We could apply the method to cluster other types of functions or use real data such as financial stock indices. We could also try Hierarchical clustering and compare its results to the results obtained by applying k-means algorithm. Moreover, we aim to develop other new methods for analyzing rough functional observations. For example, in addition to clustering, the roughness index could be applied in classification.

Bibliography

[1] The Bayes information criterion (BIC). http://www-math.mit.edu/~rmd/650/bic.pdf. Accessed: 2018-06-15.

[2] Index of notional stocks. http://www.economagic.com/em-cgi/data.exe/ecb/117.bsi.m.fi.n.a.a20.a.i.u6.1000.z01.a-m. Accessed: 2018-06-15.

[3] Population growth rate. http://www.economagic.com/em-cgi/data.exe/fedstl/sppopgrowegy. Accessed: 2018-06-11.

[4] Various functional datasets. https://www.math.univ-toulouse.fr/~ferraty/SOFTWARES/NPFDA/npfda-datasets.html. Accessed: 2018-07-25.

[5] Youth unemployment rate. http://www.economagic.com/em-cgi/data.exe/fedstl/sluem1524zsswe. Accessed: 2018-06-11.

[6] C. Abraham, P. Cornillon, E. Matzner-Lober, and N. Molinari. Unsupervised curve clustering using b-splines. *Scandinavian Journal of Statistics*, 30:1–15, 2003.

[7] T. Ayodele. *New Advances in Machine Learning*. InTech, 2010.

[8] J. Banfield and A. Raftery. Model-based Gaussian and non-Gaussian clustering. *Biometrics*, 49:803–821, 1993.

[9] S. Berman. Limit theorems for the maximum term in stationary sequences. *The Annals of Mathematical Statistics*, 35:502–516, 1964.

[10] C. Biernacki, G. Celeux, and G. Govaert. Assessing a mixture model for clustering with the integrated completed likelihood. *IEEE Transactions on pattern analysis and machine intelligence*, 22:1–16, 2000.

[11] C. Bouveyron and C. Brunet-Saumard. Model-based clustering of high-dimensional data: A review. *Computational Statistics Data Analysis*, 71:52–78, 2014.

[12] J. Chiou and P. Li. Functional clustering and identifying substructures of longitudinal data. *The Royal Statistical Society*, 69:679–699, 2007.

[13] L. Deng and D. Yu. *Automatic Speech Recognition – A Deep Learning Approach*. Springer, 2014.

[14] F. Ferraty and P. Vieu. *Nonparametric Functional Data Analysis*. Springer, 2006.

[15] L. Ferreira and D. Hitchcock. A comparison of hierarchical methods for clustering functional data. *Communications in Statistics - Simulation and Computation*, 38:1925–1949, 2009.

[16] D. Fisher. Iterative optimization and simplification of hierarchical clusterings. *Artificial Intelligence Research*, 4:147–178, 1996.

[17] G. Hébrail, B. Hugueney, Y. Lechevallier, and F. Rossi. Exploratory analysis of functional data via clustering and optimal segmentation. *Neurocomputing*, 73:1125–1141, 2010.

[18] F. Ieva, A. Paganoni, D. Pigoli, and V. Vitelli. Multi-variate functional clustering for the analysis of ECG curves morphology. *The Royal Statistical Society*, 62:401–418, 2012.

[19] J. Jacques and C. Preda. Clustering multivariate functional data. In *COMPSTAT 2012*, pages 353–366, 2012.

[20] J. Jacques and C. Preda. Funclust: A curves clustering method using functional random variables density approximation. In *Neurocomputing*, pages 164–171, 2013.

[21] J. Jacques and C. Preda. Functional data clustering: a survey. *Advances in Data Analysis and Classification*, 8(3):24, 2014.

[22] A. Kolmogorov. Wienersche spiralen und einige andere interessante kurven im hilbertschen raum. c.r. (doklady). *Acad. Sci. URSS (N.S.)*, 26:115–118, 1940.

[23] J. MacQueen. Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley Symposium on Mathematical Statistics and Probability*, pages 281–297, 1967.

[24] G. Milligan. An examination of the e□ect of six types of error perturbation on fifteen clustering algorithms. *Psychometrika*, 45:325–342, 1980.

[25] Y. Mishura. *Stochastic calculus for fractional Brownian motion and related processes*. Springer, Berlin, 2008.

[26] D. Nualart. Fractional Brownian motion: stochastic calculus and applications. In *Proceedings of the International Congress of Mathematicians*, pages 1541–1562, 2006.

[27] J. Peng and H. Müller. Distance-based clustering of sparsely observed stochastic processes, with applications to online auctions. *The Annals of Applied Statistics*, 2 (3):1056–1077, 2008.

[28] J. Ramsay and B. Silverman. *Functional Data Analysis*. Springer, 2005.

[29] J. Strandberg. Cluster analysis for functional data. Master's thesis, Umea University, Sweden, 2013.

[30] P. Tan, M. Steinbach, A. Karpatne, and V. Kumar. *Introduction to Data Mining*. Pearson, 2018.

[31] T. Tarpey and K. Kinateder. Clustering functional data. *Classification*, 20:93–114, 2003.

[32] S. Tokushige, H. Yadohisa, and K. Inada. Crisp and fuzzy k-means clustering algorithms for multivariate functional data. *Computational Statistics*, 22:1–16, 2007.

[33] J. Ward. Hierarchical grouping to optimize an objective function. *Journal of the American Statistical Association*, 58:236–244, 1963.

[34] M. Yamamoto. Clustering of functional data in a low-dimensional subspace. *Advances in Data Analysis and Classification*, 6:219–247, 2012.